Universidad de las Ciencias Informáticas
FACULTAD 6

Título: Componente para la representación espacial de dispositivos de seguridad.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Yunier Leyva Quintanilla

Tutor: Ing. Rayner Pupo Gómez

La Habana, Junio del 2013

"Año 55 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Yunier Leyva Quintanilla

_____
Firma del Autor

Ing. Rayner Pupo Gómez

_____
Firma del Tutor

DATOS DEL CONTACTO

Datos del Tutor:

Nombre y apellidos: Ing. Rayner Pupo Gómez

Correo electrónico: rpgomez@uci.cu

Categoría docente: Recién Graduado en Adiestramiento.

Año de graduación: 2011

Profesión: Ingeniero en Ciencias Informáticas.

Breve descripción: Graduado en la Universidad de las Ciencias Informáticas. Actualmente se desempeña como desarrollador del Módulo Visor en el proyecto Video Vigilancia perteneciente al Departamento de Señales Digitales del Centro de Desarrollo “Geoinformática y Señales Digitales” de la facultad 6.

AGRADECIMIENTOS

A todas las personas que me ayudaron durante estos cinco años de vida como universitario y que de una forma u otra contribuyeron a mi formación como profesional. Entre ellos quiero agradecer especialmente:

A mi familia por apoyarme en todo momento y por depositar en mí toda la confianza del mundo, principalmente a mis padres que bajo esfuerzo y sacrificio siempre dieron lo mejor para mí.

A mi prima Loydita y a mi tía Nora que me acogieron como a un hijo desde que llegué a La Habana para cumplir con el servicio militar.

A una persona tan especial para mí como Olivia, por ser un ejemplo a seguir, por ayudarme en lo que he necesitado y por sacar lo mejor de mí.

A todos mis amigos por compartir junto a ellos experiencias inolvidables y por estar ahí siempre presentes para mí.

A mi tutor por guiarme durante todo este curso y por motivarme a seguir superándome como programador.

A los profesores Carlos Luis Serrano, Olga María Rivera y Reinier Pupo Ruiz por aclararme las dudas que se me presentaban en cada momento.

A todas estas personas quiero darle las gracias de corazón, por estar ahí siempre para mí, cuando más y menos los necesitaba, la verdad es que sin ustedes hubiera sido imposible llegar a la meta.

DEDICATORIA

De forma muy especial y con mucho cariño a mis padres Valentina y Fidel por darme todo el amor del mundo y por luchar en esta vida para verme crecer como hombre.

A mi hermanito Yadier que a pesar de hacerme travesuras me quiere con el alma.

A mis abuelos Isabel, Rosa y Victor que de seguro deben estar muy orgullosos de mí.

A mis tíos, principalmente a Osmany y Varlenia por ambos quererme y preocuparse tanto por mí.

A mis primos, especialmente a José Luis que ha sido como un hermano para mí y me ha apoyado en todo momento.

A mis amistades del apto y a las del pikete de la RedBull.

A todas las personas que confiaron en mí, que me apoyaron y me incitaron a transitar por este largo camino que emprendí con voluntad y dedicación para realizar mi sueño de ser ingeniero.

RESUMEN

Los sistemas de video vigilancia tienen cada vez mayor demanda en la actualidad, especialmente para garantizar la seguridad de interiores y alrededores de las empresas e instituciones. La presencia de estos sistemas en cualquier entorno urbano es un hecho, particularmente en Cuba, donde pueden ser utilizados en lugares como embajadas, bancos, aeropuertos, tiendas comerciales y aduanas. En la Universidad de las Ciencias Informáticas (UCI) se desarrolla el sistema de video vigilancia Suria, y la vinculación de mapas o planos a dicho sistema es la expectativa propuesta durante la presente investigación. La misma tiene como objetivo el desarrollo de un componente para la representación espacial de dispositivos de seguridad que ayude a disminuir el tiempo de respuesta ante la ocurrencia de eventos como disturbios, actos vandálicos u otras situaciones de emergencia.

El componente desarrollado es una aplicación de escritorio multiplataforma que se encarga de representar sobre un plano arquitectónico los dispositivos de seguridad registrados por el módulo Visor. También el mismo es capaz de generar un fichero donde se almacena toda la información referente a la representación realizada. Mediante la representación resultante se derivan las siguientes utilidades: proporciona una clara visión de cómo pueden estar desplegados los dispositivos, en qué áreas pueden tener mayor utilidad y cómo asegurar la mayor cantidad de áreas posibles según la disponibilidad de estos. Partiendo de las ventajas ofrecidas y de los problemas que ayuda a solucionar el componente, se destaca que el mismo constituye una ampliación en las funcionalidades del sistema Suria.

Palabras Claves

Dispositivos de seguridad, mapas, planos, representación espacial, video vigilancia.

Índice de Figuras:

Índice de Tablas:

Índice:

INTRODUCCIÓN

Los mapas confeccionados antiguamente con el propósito de representar y conocer al mundo, han evolucionado con el transcurso del tiempo gracias al perfeccionamiento de las técnicas para su elaboración, apoyado sobre la base del desarrollo científico-técnico alcanzado en materia geográfica. Estas representaciones gráficas constituyen en la actualidad una fuente importante de información, en donde gran parte de la actividad humana mantiene una estrecha relación con la cartografía (PCS, 1998). Los mapas se utilizan principalmente para la representación gráfica y métrica de un determinado territorio, pueden contener elementos importantes como las referencias, que permiten identificar características específicas del mapa para una mejor comprensión del mismo, y, las representaciones en escala, que permiten saber en qué medida se ha reducido la realidad para graficar. Además, a partir de la escala se pueden realizar cálculos de distancias, ángulos o superficies para arribar a un resultado tan exacto como sea posible (Definición.de, 2008).

Relacionado con los mapas se encuentra el plano, como representación de una superficie geométrica bidimensional y, a diferencia de los mapas, este tiene la particularidad que para elaborarlo no es necesario realizar una proyección sobre el mismo. Este procedimiento matemático se emplea para representar una superficie curva en una plana. Además es válido destacar que los planos suelen realizarse para representar espacios relativamente pequeños o delimitados (Bembibre, 2007).

La aplicación de mapas y planos resulta importante en diferentes esferas de la sociedad, debido a que no solo permiten representar territorios y obras arquitectónicas realizadas por el hombre, sino que también sus utilidades pueden ser vinculadas a sistemas informatizados. Dentro de estos sistemas se encuentran los basados en video vigilancia que utilizan cámaras IP[1] y una serie de dispositivos de seguridad para cumplir con la prestación de servicios. Estos sistemas se desarrollan con el objetivo de garantizar la seguridad de interiores y alrededores de empresas e instituciones, proporcionando la capacidad de monitorizar, seguir y reconocer indicios visuales que ayuden a prevenir robos o identificar algún tipo de evento.

En el mundo se evidencia una creciente demanda por los sistemas basados en video vigilancia debido a la necesidad de proteger lugares de importancia relevante. Particularmente en Cuba, donde no sólo se deben asegurar los recursos con los que se cuentan, sino también diferentes zonas de interés como,

[1] Protocolo de Internet (*Internet Protocol*)

lugares turísticos, centros comerciales, aeropuertos y embajadas por solo citar algunos ejemplos. La mayoría de estos sistemas presentan licencias propietarias y hacer uso de los servicios de soporte resultaría costoso, entrando además en desacuerdo con el proceso de migración a software libre llevado a cabo por el país para garantizar la soberanía tecnológica. Producto a la utilización del sistema operativo Linux como plataforma de código abierto quedarán en desuso y prácticamente obsoletas las aplicaciones informáticas que necesiten la plataforma Windows para ser funcionales. La Universidad de las Ciencias Informáticas (UCI), no ha estado ajena a este proceso de migración tecnológica por lo que en el centro de desarrollo Geoinformática y Señales Digitales (GEYSED) específicamente en el proyecto de Video Vigilancia Inteligente (VVI), se desarrolla bajo tecnologías libres el sistema de video vigilancia Suria, el cual utiliza cámaras IP para realizar un control monitorizado. El sistema cuenta con cinco módulos, entre los que se encuentra el módulo Visor, encargado de visualizar los flujos de video emitidos por las cámaras IP y con el cual interactúa el operador para hacer uso de sus funciones. Producto a que el usuario del módulo desconoce la localización o ubicación exacta de los dispositivos de seguridad, surge como problemática que, la velocidad de respuesta se vea disminuida ante la ocurrencia de eventos adversos como actos vandálicos, disturbios, robos u otras situaciones de emergencia.

Atendiendo a la situación problemática existente se define como **problema a resolver:** ¿Cómo facilitar la representación espacial de dispositivos de seguridad sobre un plano?

Como consecuencia se determina que el **objeto de estudio** son los procesos para la representación espacial de dispositivos de seguridad, enmarcándose como **campo de acción** la informatización de los procesos para la representación espacial de dispositivos de seguridad en sistemas de video vigilancia.

Teniendo en cuenta la situación problemática descrita, el problema planteado, el objeto de estudio, así como el campo de acción delimitado, la presente tesis se estructura y desarrolla en función del siguiente **objetivo general**: Desarrollar un componente para la representación espacial de dispositivos de seguridad.

A partir de lo anteriormente planteado se propone como **idea a defender:** Con el desarrollo del componente para la representación espacial de dispositivos de seguridad se logrará que el módulo Visor cuente con una herramienta capaz de facilitar la representación espacial de dispositivos de seguridad sobre un plano, lo que constituye una ampliación en las funcionalidades del sistema Suria.

Para el cumplimiento del **objetivo general** fueron definidas las siguientes **tareas para la investigación**:

1. Caracterizar los procesos relacionados con el funcionamiento de los distintos tipos de dispositivos de seguridad.
2. Caracterizar las soluciones existentes con características similares al componente que se necesita desarrollar.
3. Caracterizar las metodologías de desarrollo de software, las herramientas y tecnologías seleccionadas para la implementación y desarrollo del componente.
4. Realizar el diseño del componente para la representación espacial de dispositivos de seguridad.
5. Implementar el componente para la representación espacial de dispositivos de seguridad.
6. Realizar el diseño y la implementación de pruebas al componente.

Durante el desarrollo de la investigación se utilizan un grupo de **métodos científicos** entre los que se encuentran:

- Los **métodos teóricos**:

  - **Analítico-Sintético:** Se utiliza en el estudio de las bibliografías para realizar la fundamentación teórica de la investigación y sintetizar los conceptos fundamentales que sean necesarios para la solución del problema.
  - **Análisis Histórico-Lógico:** Se emplea en el análisis de los procesos de representación en el espacio de dispositivos de seguridad en sistemas basados en video vigilancia existentes en el mundo y en el país.
  - **Modelación:** Se utiliza para modelar a partir de diagramas ingenieriles los procesos de representación espacial de dispositivos de seguridad facilitando una mayor comprensión de la solución a implementar.

El contenido del presente trabajo de diploma se divide en cuatro capítulos:

Capítulo 1: Fundamentación teórica de los procesos para la representación espacial de dispositivos de seguridad. En este capítulo se describe lo referido a los fundamentos teóricos de la investigación, se hace énfasis en la descripción del objeto de estudio para caracterizar los procesos de representación e identificar la problemática existente. Además se realiza un estudio del estado del arte partiendo del análisis de soluciones existentes.

Capítulo 2: Tecnologías, herramientas y metodología. En este capítulo se describen las tecnologías y herramientas en las cuales se apoya la solución, las características del marco de trabajo a utilizar, se

selecciona una biblioteca para el procesamiento de imágenes digitales, así como la metodología de desarrollo de software para guiar todo el proceso de desarrollo del componente.

Capítulo 3: Características del componente para la representación espacial de dispositivos de seguridad. En este capítulo se plantea la conceptualización de un entorno mediante un modelo de dominio. Se definen los requisitos funcionales y no funcionales propios del sistema. Se especifican los actores y los casos de uso que tendrá el sistema con sus respectivas descripciones. Se selecciona el patrón arquitectónico bajo el cual se va a estructurar el componente y se describen los patrones de diseño involucrados.

Capítulo 4: Construcción y Validación de la solución propuesta. Se desarrollan los artefactos del Diseño e Implementación, dígase modelo de diseño, despliegue e implementación. Para finalizar, en el capítulo se documentan las pruebas realizadas al software.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA DE LOS PROCESOS PARA LA REPRESENTACIÓN ESPACIAL DE DISPOSITIVOS DE SEGURIDAD.

1.1 Introducción

Para lograr una mejor comprensión acerca de los procesos de representación espacial es necesario conocer los principales conceptos que intervienen en el dominio del problema. Estas definiciones hacen alusión a elementos que forman parte del entorno donde se debe desarrollar el componente para la representación espacial de dispositivos de seguridad. Se destaca que los elementos descritos se involucran directamente con la representación espacial debido a que forman parte de dicho proceso. Los procesos de representación espacial constituyen el peso fundamental de la investigación y es importante saber cómo van a estar relacionados con el propósito de la misma teniendo en cuenta la problemática descrita. Además se debe realizar un análisis de las soluciones existentes tanto en Cuba como en el resto del mundo con características similares al componente a desarrollar. En el contenido del capítulo se explican con claridad los aspectos mencionados.

1.2 Conceptos asociados al dominio del problema

La palabra **mapa** proviene del término latino mappa. Se trata de una representación geográfica de la Tierra o parte de ella en una superficie plana (RAE, 2010). Fundamentalmente los mapas son empleados para representar un determinado territorio y sus particularidades, a partir de ellos se derivan un número de utilidades como lo son el análisis de comportamiento de fenómenos meteorológicos, el trazo de rutas aéreas, marítimas y terrestres, etc. Relacionados con los mapas están los planos, que gráficamente indican o marcan ubicaciones, desde una ciudad hasta una casa. Teniendo en cuenta que un **plano** es una representación gráfica de una superficie geométrica bidimensional (Dictionary, 2012).

Los planos son utilizados en disímiles funciones gracias a la información que aportan, entre ellas se encuentra su aplicación en sistemas basados en **video vigilancia**, siendo esta la vigilancia realizada a través de un sistema de cámaras, fijas o móviles (RAE, 2010). Atendiendo a la definición de video vigilancia un **sistema de video vigilancia** está compuesto por un conjunto de cámaras IP u otros dispositivos de seguridad que se encuentran interconectados mediante cableado o inalámbricamente a una estación central o servidor para la administración del sistema y el almacenamiento de datos. La información transmitida por estos dispositivos es procesada en el servidor y, la misma podría ser, el video capturado por las cámaras, los movimientos detectados, así como la notificación ante eventos. Los **dispositivos de seguridad** son componentes autónomos que se integran al sistema de video vigilancia

(Carrasco, 2005). Entre ellos se encuentran las **cámaras IP** las cuales posibilitan la captura de video y generalmente cuentan con un servidor web de video incorporado, lo que les permite transmitir su imagen a través de redes IP como en redes LAN[2], WAN[3] e Internet (AE, 2005).

1.3 Descripción general del objeto de estudio de la investigación

Las representaciones espaciales pueden ser utilizadas en diferentes materias como la matemática, la geografía y la informática. Resultando provechoso su empleo, debido a que las mismas ofrecen un acercamiento a la realidad de lo representado, de forma orientadora e ilustrativa. Las mismas pueden ser útiles según el propósito o la finalidad con que sean empleadas.

Para comprender el objeto de estudio de la investigación fue necesario realizar un estudio acerca de las representaciones espaciales sobre mapas o planos utilizados por sistemas de video vigilancia. Desarrollando como idea principal la representación espacial de dispositivos de seguridad (cámaras IP, detectores de intrusos, alarmas contra incendios, detectores de humo) que pertenecen al sistema Suria.

Existen representaciones del espacio geográfico como los mapas y planos donde es posible ubicar elementos que se encuentren en correspondencia con la realidad. Para realizar estos procesos de representación resulta importante conocer el lugar o territorio que se refleja mediante dichas representaciones gráficas. Además se deben tener en cuenta las características que poseen en el mundo real los elementos a representar. La posición es otro de los aspectos a destacar cuando se quiere representar un objeto que brinde determinada información. El comportamiento de dichos elementos dentro del espacio puede ser variable y debe estar en correspondencia con el entorno representado.

Partiendo de lo planteado es válido aclarar que antes de realizar los procesos de representación es importante conocer dónde se encuentra posicionada la información contenida por la imagen digital del mapa o plano. Para garantizar el dominio de tal información y facilitar dichos procesos es necesario aplicar algoritmos para el procesamiento de imágenes digitales que se encarguen de realizar las transformaciones pertinentes a la imagen para luego poder identificar con claridad el contenido. A partir del resultado derivado por este proceso se pueden utilizar algoritmos especializados para la detección de líneas y objetos que ayuden a considerar las posiciones correspondientes a los elementos contenidos en la imagen. Al finalizar la etapa del procesamiento de la misma se puede apreciar que los procesos de representación sobre un mapa o plano pueden resultar con mayor facilidad, partiendo de que se puede

[2] Red de Área Local (*Local Area Network*)
[3] Red de Área Amplia (*Wide Area Network*)

conocer el contenido del mismo. Lo que garantiza que la integración de mapas o planos al sistema Suria sea factible para la realización de procesos de representación espacial.

Para comprender la estructura del sistema Suria y conocer las responsabilidades de sus módulos, se hizo necesario reflejar la composición del sistema, debido a que el componente a desarrollar durante la presente investigación formará parte del módulo Visor.

El sistema Suria cuenta con cinco módulos que le permiten realizar una serie de funcionalidades durante la prestación de sus servicios. Entre estos módulos se encuentran:

- **Modulo Gestor:** encargado de gestionar la interacción entre los demás módulos del sistema.
- **Módulo Analítico:** permite el procesamiento de flujos de video mediante video sensores.
- **Módulo Grabador:** gestiona y ejecuta las grabaciones en el sistema.
- **Módulo Recuperador:** subsistema para la búsqueda y visualización de grabaciones.
- **Módulo Visor:** permite la visualización y gestión de las cámaras IP.

Este último módulo es el encargado de visualizar los flujos de video emitidos por las cámaras IP y de registrar los dispositivos de seguridad que forman parte de Suria, donde se cataloga como operador a la persona que interactúa con el mismo. Partiendo de la relación establecida entre el operador y el módulo se derivan un conjunto de procesos como la personalización de estilos de visualización, la gestión de cámaras IP de distintos fabricantes, la visualización de flujos de video, por solo citar algunos. Luego de caracterizar los procesos realizados en el módulo, se identificaron dificultades que presenta el operador para conocer la localización o ubicación exacta de los dispositivos de seguridad del sistema, lo que trae consigo que se vea afectada la velocidad de respuesta ante situaciones de emergencia. Por lo que se evidencia la necesidad de representar sobre un mapa o plano los dispositivos de seguridad registrados por el módulo Visor.

Teniendo en cuenta que las utilidades de dichas representaciones gráficas pueden ser aplicadas a las funciones del módulo Visor, se propone la informatización de procesos de representación relacionados con planos. Se seleccionaron los planos debido a que contienen la información necesaria correspondiente al lugar u obra arquitectónica donde se encuentre desplegado el sistema. Dentro de dicha información se pueden encontrar líneas que representen paredes, puertas o ventanas, y otros objetos que sean posibles obstáculos. Partiendo del vínculo establecido se ofrecerán una serie de ventajas para Suria, aumentando sus prestaciones con un valor añadido. Entre las utilidades ofrecidas mediante la representación resultante se encuentran: proporcionarle al operador un amplio dominio sobre la forma en que está

desplegado el sistema, facilitar un entorno configurable para la representación espacial de dispositivos de seguridad y mejorar la capacidad de respuesta ante cualquier eventualidad.

1.4 Análisis de soluciones existentes

Para la argumentación de este epígrafe se investigaron algunos sistemas de video vigilancia existentes a nivel nacional e internacional que utilizan cámaras IP y que emplean mapas o planos para realizar funciones durante la prestación de servicios. Entre estos sistemas se encuentran:

(DATYS, 2010)

Figura 1 Xyma Safe Vision

XYMA SAFE VISION

El producto XYMA SAFE VISION desarrollado por la empresa cubana DATYS, es un software de video vigilancia profesional basado en tecnología IP, con un alto grado de modularidad, adaptable a una gran cantidad de entornos, flexible, escalable y con el uso de tecnologías analógicas. Permite mantener la seguridad monitorizando y controlando en tiempo real y de forma histórica cada uno de los movimientos que ocurren en las áreas sensibles que se identifiquen. Las principales características de este sistema son (DATYS, 2010):

- Herramientas para la administración y configuración del sistema.
- Grabaciones de forma manual, programada, continua y por detección de movimiento.
- Monitoreo y grabación de video y audio desde cada cámara.
- Visualización y manipulación de las grabaciones.
- Realización de acciones y emisión de mensajes ante alarmas.
- Compatibilidad con una amplia gama de productos AXIS, VIVOTEK, PANASONIC, ACTI, SONY y PLANET.
- Capacidad ilimitada de manejo de usuarios y cámaras.

- Control PTZ[4] remoto con alternativas interactivas.
- Manejo de múltiples pantallas en el monitor.
- Mapas y planos para apoyar la administración y utilización del sistema.

(CORADIR, 2012)

Figura 2 Coradir S.A

Sistema de Video Vigilancia Coradir S.A.

Para constituir un sistema de video vigilancia eficiente es necesaria la presencia combinada de sistemas de video abiertos, imágenes digitales, transporte sobre redes IP, vinculadas con la presencia de cámaras de alta gama y sistemas de visualización inteligentes. CORADIR S.A. es una empresa argentina que ha logrado una síntesis eficiente de los elementos antes señalados, lo cual le permite contar con un sistema de altas prestaciones, pero con costos sensiblemente bajos. Este sistema cuenta las siguientes características técnicas (CORADIR, 2012):

- Servidores de hasta 80 cámaras.
- Vista matricial configurable.
- Función splash[5] en visualización.
- Control de cámaras PTZ.
- Registros de detección de movimiento.
- Grabación por tarea programada en bloques horarios, fecha, día, hora.
- Envío de mail, SMS[6]. Ante eventos, o desconexión de cámaras.
- Cámaras virtuales, proxy para grandes instalaciones.
- Integración con mapas para proyectos de vigilancia urbana.
- JPEG, MPEG, MPEG4, H264.
- Integración de google maps con equipos GPS[7].
- Asociación de equipos GPS con cámaras móviles.

[4] Panorámica-Inclinación-Acercamiento (*Pan-Tilt-Zoom*)
[5] Efectos de color sobre la imagen o video
[6] Sistema de Mensajes Cortos (*Short Message System*)
[7] Sistema de Posicionamiento Global (*Global Positioning System*)

(BOSCH, 2012)

Figura 3 BOSCH

Video vigilancia BOSCH

Video Management System permite la completa administración de todos los elementos de video vigilancia. Las cámaras CCTV[8] de alta resolución pueden realizar un seguimiento continuo de los grupos a medida que se acercan y entran al estadio, así como de las gradas a fin de identificar a los infractores. Se pueden emplear micrófonos direccionales para grabar los eventos.

Sistema de gestión de edificios

El sistema de integración de edificios (BIS) de Bosch ofrece un punto de control en línea unificado para la supervisión de los sistemas de gestión de edificios, seguridad y protección. Entre sus características se encuentran la gestión de alarmas, la rápida identificación de alarmas mediante mapas de localización o los procedimientos de seguimiento. Integra a la perfección los sistemas de detección de intrusión, evacuación, control de acceso, CCTV y automatización de edificios en una sola plataforma.

Mediante el BIS, un único operador puede supervisar y controlar todos los sistemas de seguridad, al tiempo que se acelera el tiempo de respuesta frente a emergencias y se mejora el nivel general de eficacia (BOSCH, 2012).

(CITELUM, 2012)

Figura 4 CITELUM

CITELUM

Entrega soluciones de video vigilancia en zonas urbanas con beneficios para toda la ciudadanía en seguridad pública y disuasión de delito. Con el profundo conocimiento de las plataformas que ofrece el

[8] Circuito Cerrado de Televisión

mercado de la seguridad electrónica, hoy en día CITELUM ofrece integración de plataformas sólidas para servicio a urbes modernas con tecnología de punta utilizando cámaras IP que permiten la transmisión de video en redes inteligentes Ethernet. Este sistema de video vigilancia propone las siguientes características (CITELUM, 2012):

- Hasta 10,000 cámaras en una misma ciudad.
- Almacenamiento de hasta 72 horas para todas ellas.
- Análisis de contenido:
  - Objetos abandonados.
  - Líneas de seguridad.
  - Detección temprana de amenazas.
  - Conteo de Personas.
- Alarma de robo en video.
- Zoom digital sobre video grabado con cámaras mega pixel.
- Visualización de video sobre cartografía o mapas digitales.
- Redes de video combinadas inalámbricas, fibra óptica y Ethernet bajo una misma plataforma.

Todos estos sistemas de video vigilancia utilizan mapas o planos para realizar funciones como, la vinculación de google maps con equipos GPS para realizar procesos de localización, como herramientas de apoyo para la administración y utilización del sistema, así como la visualización de video sobre mapas digitales. Las soluciones analizadas se caracterizan por ser modernas, contando con una amplia capacidad funcional, pero la mayoría se encuentran bajo licencias propietarias lo que implicaría para el país la realización de una inversión. Aunque integran a sus sistemas mapas o planos sus funcionalidades no se ajustan a la necesidad real de la investigación, basada en el uso de los mismos para realizar procesos de representación espacial de dispositivos de seguridad.

1.5 Conclusiones

La descripción de conceptos asociados al problema posibilitó reflejar con claridad el significado de los principales conceptos a tratar durante la investigación. La profundización acerca del objeto de estudio garantizó precisar aspectos importantes a tener en cuenta para realizar los procesos de representación como, la posición y el comportamiento del elemento a representar. El análisis de soluciones existentes evidenció la utilidad de dichas representaciones gráficas y posibilitó extraer características comunes en estos sistemas. Partiendo de la dificultad presentada por el operador para notificar alertas ante la ocurrencia de eventos adversos y teniendo en cuenta la existencia del sistema Suria, surge la necesidad de desarrollar un componente para la representación espacial de dispositivos de seguridad que solucione los problemas identificados.

CAPÍTULO 2: TECNOLOGÍAS, HERRAMIENTAS Y METODOLOGÍA.

2.1 Introducción

Para la construcción del componente es necesario caracterizar y seleccionar las tecnologías a utilizar, una metodología de software capaz de guiar el proceso de desarrollo, así como definir las herramientas que apoyen dicho proceso. Antes de realizar estas actividades se deben tener en cuenta las tecnologías y herramientas propuestas por el departamento de Señales Digitales para el desarrollo de aplicaciones de escritorio, debido a que el componente a desarrollar formará parte de Suria. Durante el desarrollo del capítulo se describen las bases tecnológicas que dan soporte a la realización del componente.

2.2 Metodología de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas que permiten guiar el proceso de desarrollo del software derivando toda la documentación asociada. Las mismas indican las actividades a realizar y los roles que deben desempeñar las personas que intervienen en dichas tareas durante cada etapa del proceso (Barzanallana, 2006).

RUP[9] es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto. RUP utiliza el UML[10] para realizar los esquemas de un sistema de software. Pero el peso de la metodología de desarrollo de software RUP recae sobre tres aspectos fundamentales que rigen todo el proceso de desarrollo, estas características son (Jacobson, 2000):

- *Dirigido por Casos de Uso:* los casos de uso constituyen un medio para determinar los requisitos correctos y utilizarlos para conducir el proceso de desarrollo.
- *Centrado en la arquitectura:* la arquitectura establece lo que se tiene que hacer; esquematiza los niveles significativos de la organización del software y se centra en la armazón del sistema.
- *Iterativo e Incremental:* propone adoptar una aproximación iterativa e incremental durante el desarrollo del software. Las iteraciones constituyen pasos en el flujo de trabajo, reflejándose como resultado de estos un incremento del producto.

RUP es una metodología con ciclo de vida comprendido por cuatro fases (inicio, elaboración, construcción y transición), en estas fases se realizan diferentes actividades definidas para cada una de las nueve

[9] Proceso Unificado de Rational (*Rational Unified Process*)
[10] Lenguaje Unificado de Modelado (*Unified Modeling Language*)

disciplinas de trabajo, seis ingenieriles y tres de soporte. Cada vez que se termina un ciclo, de este se deriva una nueva versión del producto (Jacobson, 2000).

El proyecto VVI es un proyecto en donde se genera un gran volumen de documentación y otros artefactos importantes como el código fuente de los módulos del sistema Suria, por lo que se necesita de un tipo de metodología robusta o tradicional como RUP. Esta metodología de desarrollo de software ha demostrado ser efectiva en proyectos de gran tamaño respecto a tiempo y recursos, propone una planificación bien definida para la realización de actividades durante todo el proceso de desarrollo. Se seleccionó esta metodología porque el componente para la representación espacial de dispositivos de seguridad forma parte del sistema Suria desarrollado bajo RUP, definido en el documento de arquitectura del proyecto. Además reúne elementos de todos los modelos de procesos genéricos, toma en cuenta las mejores prácticas en el modelo de desarrollo de software, la documentación generada en cada fase del proceso permite que el software sea entendible para cualquier desarrollador durante la realización de versiones posteriores, y su objetivo es producir un software de alta calidad.

2.3 Lenguaje de Modelado

Para el desarrollo del componente se necesita seleccionar un lenguaje de modelado que proporcione una visión constructiva del software y que se encuentre en correspondencia con la metodología de desarrollo a utilizar.

UML es un lenguaje de modelado visual que se utiliza para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se emplea para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos.

Se seleccionó el UML debido a que la metodología RUP lo propone como lenguaje de modelado para el proceso de desarrollo del software. Este lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar (Booch, 2000).

2.4 Lenguaje de Programación

Un lenguaje de programación es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que este pueda comunicarse con los elementos de hardware y software existentes (RAE, 2010).

Lenguaje de Programación C++

C++ es un lenguaje imperativo orientado a objetos derivado del C, en realidad un superconjunto de C, que nació para añadirle cualidades y características de las que carecía. El resultado es que como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se la han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción (Stroustrup, 2000).

Este lenguaje de programación fue seleccionado por ser un lenguaje estandarizado, eficiente en cuanto a tiempo de ejecución, con muy buen soporte de bibliotecas y el código fuente expresado en el mismo puede ser compilado en diversas plataformas (Americati, 2006).

2.5 Marco de trabajo

Un marco de trabajo es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que permite resolver nuevos problemas. En el desarrollo de software constituye una estructura conceptual y tecnológica de soporte definida, normalmente cuenta con artefactos o módulos de software concretos, empleados fundamentalmente para desarrollar código fuente (Quesada, 2008).

Qt 4.8

Marco de trabajo multiplataforma de código abierto que se utiliza generalmente para desarrollar aplicaciones de software haciendo uso de una interfaz gráfica GUI[11], cuenta con soporte para la programación multihilo, la comunicación con bases de datos, etc. Qt utiliza el C++ como lenguaje de programación nativo haciendo un uso extensivo de un generador de código llamado MOC[12], junto con varias macros para enriquecer el lenguaje (ZonaQt, 2010).

Este marco de trabajo fue seleccionado debido a que cumple con las políticas definidas para el proyecto VVI en el documento de arquitectura, haciendo uso de las ventajas que ofrecen las tecnologías libres.

[11] Interfaz Gráfica de Usuario (*Graphical User Interface*)
[12] Compilador de Meta Objetos (*Meta Object Compiler*)

2.6 Entorno de Desarrollo Integrado

Un IDE[13] es un programa compuesto por una serie de herramientas que utilizan los programadores para desarrollar código. Esta herramienta puede estar pensada para su utilización con un único lenguaje de programación o bien puede dar cabida a varios de estos (Pes, 2006).

Qt Creator 2.4

Qt Creator es un IDE multi-plataforma, puede ser ejecutado en Windows, Linux/X11 o Mac OS X, posibilita a los desarrolladores crear aplicaciones de escritorio y permite desplegar aplicaciones sobre múltiples plataformas de dispositivos móviles.

Cuenta con las siguientes características principales (Digia, 2012):

- Editor avanzado para C++.
- Diseñador de formularios (GUI) integrado.
- Herramientas para la administración y construcción de proyectos.
- Completado automático.
- Depurador visual.

Para la implementación del componente se seleccionó como IDE el Qt Creator en su versión 2.4, debido a que el mismo es multiplataforma, de libre distribución y utiliza las potencialidades del lenguaje de programación C++.

2.7 Bibliotecas utilizadas

Las bibliotecas son un conjunto de subprogramas utilizados para desarrollar software. Las mismas contienen código y datos que proporcionan servicios a programas independientes pasando a formar parte de estos.

OpenCV 2.3.1

OpenCV[14] es una biblioteca de funciones en C y C++ que contiene un conjunto de utilidades de procesamiento de imágenes, visión artificial, captura de video y visualización de imágenes. Es de código abierto, gratuita, multiplataforma (disponible para entornos MS Windows, Mac OS y Linux), rápida, de fácil uso, la misma se encuentra en continuo desarrollo. OpenCV fue desarrollada originalmente por Intel.

[13] Entorno de Desarrollo Integrado (*Integrated Development Environment*)
[14] (*Open Source Computer Vision Library*)

Actualmente es un proyecto libre publicado bajo licencia BSD[15], lo que permite su uso tanto para aplicaciones comerciales como no comerciales (Mateos, 2010).

OpenCV 2.3.1 es una biblioteca que se seleccionó por su capacidad en el procesamiento y visualización de imágenes para el desarrollo del componente. Mediante las diferentes transformaciones y detecciones realizadas a la imagen del plano es posible conocer la posición que presentan las líneas u objetos contenidos dentro del mismo, aspecto importante a tener en cuenta durante los procesos de representación.

2.8 Herramienta CASE

Las herramientas CASE[16] ayudan a los gestores y practicantes de la ingeniería del software en todas las actividades asociadas a los procesos de software. Automatizan las actividades de gestión de proyectos, gestionan los productos de los trabajos elaborados a través del proceso, y ayudan a los ingenieros en el trabajo de análisis, diseño y codificación (Pressman, 2002).

Visual Paradigm 8.0

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML constituye una ayuda para la construcción de aplicaciones, con calidad y a un menor coste. Posibilita realizar diferentes tipos de diagramas, código inverso, generar código a partir de diagramas y generar documentación. La herramienta CASE también proporciona tutoriales de UML, demostraciones interactivas de UML y proyectos UML (Soft, 2010).

Esta herramienta se seleccionó para la realización de diagramas ingenieriles utilizando UML. Además se destaca que la misma forma parte de las tecnologías y herramientas propuestas por el departamento de Señales Digitales para el desarrollo de aplicaciones de escritorio.

[15] *Berkeley Software Distribution*

[16] Ingeniería de Software Asistida por Computador (*Computer Aided Software Engineering*)

2.9 Conclusiones

Al finalizar las actividades documentadas en este capítulo se concluye que la caracterización de tecnologías y herramientas posibilitó realizar una selección de estas teniendo en cuenta los principios de soberanía tecnológica que defiende el país. La inclusión de la metodología RUP asegura una guía para regir todo el proceso de desarrollo del componente y propone para cada flujo de trabajo las actividades a realizar para generar un conjunto de artefactos ingenieriles. Además es válido agregar que la calidad del software dependerá en gran medida del cumplimiento de estas tareas y de la combinación de herramientas a utilizar. Partiendo de los conocimientos adquiridos sobre el sistema Suria, la caracterización de los procesos de representación descritos en el capítulo 1 y las tecnologías, herramientas y metodología a utilizar, se orienta el trabajo al desarrollo del componente.

CAPÍTULO 3: CARACTERÍSTICAS DEL COMPONENTE PARA LA REPRESENTACIÓN ESPACIAL DE DISPOSITIVOS DE SEGURIDAD.

3.1 Introducción

Para darle continuidad al proceso de desarrollo del componente es necesario definir sus características partiendo del propósito para el cual es desarrollado y las necesidades que debe satisfacer. Asumiendo que dentro de estas se pueden encontrar propiedades y cualidades funcionales que sirven de antesala para definir una arquitectura. Estas particularidades se engloban dentro del flujo de trabajo Requisitos propuesto por la metodología RUP durante el proceso de desarrollo de software.

Se destaca también que es importante conocer cómo va interactuar el componente con el módulo Visor y los principales elementos que se involucran en dicho entorno. Una vez comprendidas las características y la lógica de funcionamiento de la solución propuesta, se debe seleccionar una arquitectura que sea capaz de estructurar el sistema. La arquitectura define un diseño y asegura la interacción entre los elementos del software para cubrir todos los objetivos y restricciones del mismo. A continuación en este capítulo se enuncian elementos importantes referentes a los aspectos planteados para lograr un mejor entendimiento de cómo tiene que ser y qué debe hacer el componente.

3.2 Modelo de dominio

Se plantea la conceptualización de un entorno mediante un modelo de dominio para comprender el contexto del sistema. Esto se debe a que durante la investigación no se definen procesos de negocio que estén involucrados con el dominio del problema. Este modelo es una representación visual que permite agrupar los principales conceptos del dominio y las relaciones que se establecen entre éstos, proporcionando un vocabulario común para usuarios y desarrolladores (Jacobson, 2000).

3.2.1 Descripción de clases del dominio

Sistema Suria: sistema basado en video vigilancia que utiliza tecnología IP, del cual forman parte los módulos: Gestor, Análisis, Grabador, Recuperador y Visor. Suria es desarrollado con el objetivo de gestionar los procesos de video vigilancia realizados por cámaras IP u otros dispositivos de seguridad. El mismo se encarga de garantizar la vigilancia elevando el control monitorizado en lugares o instituciones donde sean requeridos estos servicios.

Módulo Visor: módulo encargado de gestionar las cámaras IP y visualizar los flujos de videos generados por las mismas. Proporciona un entorno configurable y organizativo de los dispositivos de seguridad registrados en el sistema y propone diferentes estilos de visualización.

Componente para la Representación Espacial de Dispositivos de Seguridad: forma parte del módulo Visor y facilita la representación espacial de dispositivos de seguridad del sistema mediante la utilización de planos. Dispone de una interfaz visual encaminada a orientar al usuario durante la realización de los procesos de representación.

Dispositivos de seguridad: son componentes que forman parte del sistema de video vigilancia Suria, encontrándose involucrados en los procesos de video vigilancia. Estos dispositivos emiten flujos de video, notificaciones y todo tipo de información relevante que deba ser procesada por el sistema. Los mismos pueden ser cámaras IP fija o manipulable, alarmas de incendio, detectores de humo y detectores de intruso o de movimiento.

Plano: representación gráfica del lugar o territorio donde se necesite representar los dispositivos de seguridad del sistema Suria, de forma distribuida y organizada.

Representación: se deriva de la relación establecida entre los dispositivos de seguridad y la representación de sus particularidades sobre un plano, teniendo en cuenta la información contenida en el mismo.

Archivo de Representación de Dispositivos de Seguridad: es un archivo generado por el componente luego de finalizar los procesos de representación sobre el plano. Una vez guardado este archivo puede ser cargado nuevamente para ser editado.

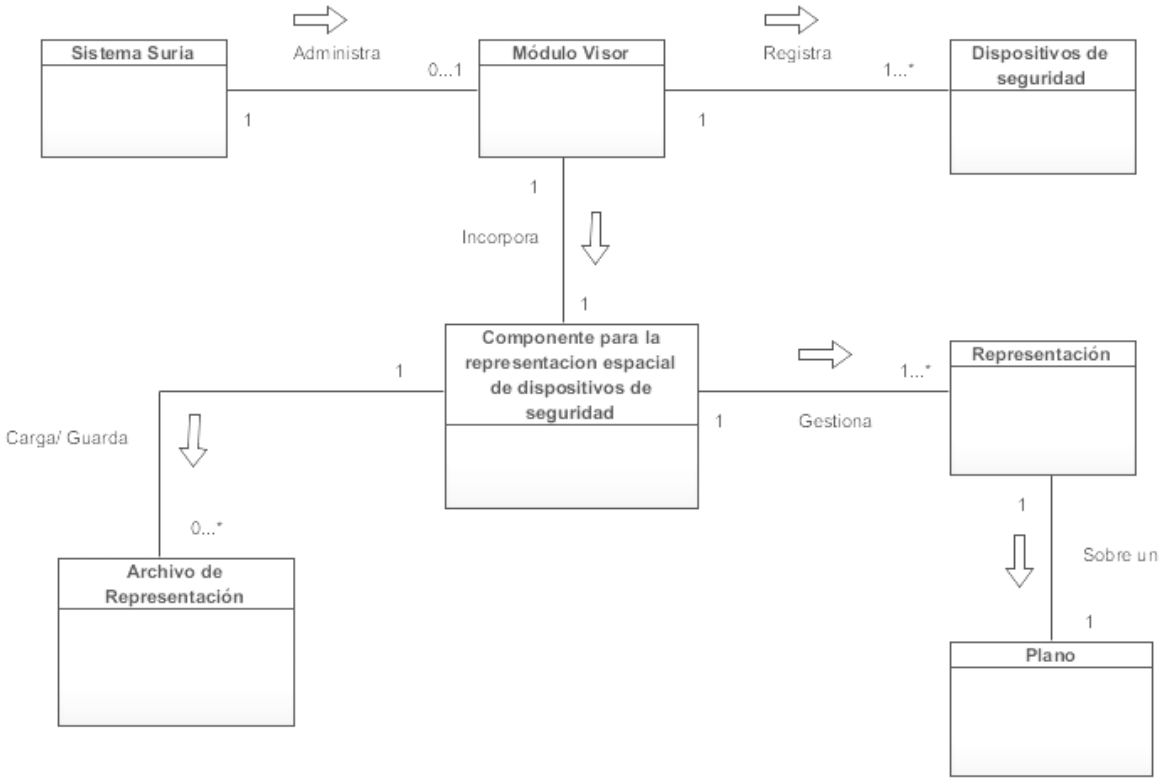


Figura 5 Modelo de dominio

3.2.2 Descripción del flujo del diagrama del modelo de dominio

El sistema Suria está compuesto por cinco módulos que procesan la información obtenida para cumplir con la prestación de servicios durante la video vigilancia. Los módulos que conforman al sistema son: Gestor, Análisis, Grabador, Recuperador y Visor. Este último se encarga de visualizar los flujos de video emitidos por las cámaras IP y además puede registrar otros dispositivos de seguridad que conformen al sistema. El módulo Visor incorpora un componente capaz de gestionar la representación espacial de los dispositivos de seguridad sobre un plano, aprovechando las utilidades que aporta dicha representación gráfica. Luego de finalizar el proceso de representación el componente genera un archivo con la información referente a los procesos de representación realizados sobre el plano. Una vez guardado este archivo, el mismo puede ser cargado nuevamente para ser editado.

3.3 Especificaciones de requisitos

3.3.1 Requisitos Funcionales (RF)

Los requisitos funcionales son capacidades con las que debe contar el componente a desarrollar. Los mismos especifican el cómo debe reaccionar el componente ante las entradas de información y cómo se debe comportar en situaciones particulares (Sommerville, 2005).

RF1: Cargar imagen del plano: esta funcionalidad debe permitir cargar una imagen de un plano arquitectónico compatible con los formatos relacionados a las extensiones (.jpg, .png, .jpeg, .bmp) a partir de una dirección especificada por el usuario.

RF2: Cargar archivo de representación: esta funcionalidad debe permitir cargar un archivo con extensión (.drf) que contendrá la representación de los dispositivos de seguridad en un plano a partir de una dirección especificada.

RF3: Redimensionar la imagen del plano: esta funcionalidad debe permitir redimensionar la imagen del plano a distintas escalas para facilitar los procesos de representación.

RF4: Ubicar dispositivos de seguridad en el plano: esta funcionalidad debe permitir ubicar en el plano los dispositivos de seguridad que han sido registrados en el módulo Visor teniendo en cuenta la información contenida en el plano. Los mismos se agrupan en una estructura jerárquica en forma de árbol donde se encuentran organizados por zonas.

RF5: Eliminar dispositivos de seguridad del plano: esta funcionalidad debe permitir eliminar cualquier dispositivo de seguridad que haya sido ubicado en el plano así como toda la información de representación asociada al mismo.

RF6: Definir un ángulo de orientación para las cámaras IP fijas: esta funcionalidad debe permitir asignarle a las cámaras IP un ángulo de orientación. Posibilita representar la dirección o sentido donde se encuentran apuntando las cámaras.

RF7: Mostrar información del dispositivo de seguridad: esta funcionalidad debe permitir mostrar la información del dispositivo de seguridad que haya sido seleccionado.

RF8: Guardar archivo de representación: esta funcionalidad debe permitir guardar un archivo de representación con extensión (.drf) que contenga toda la información referente a la representación de los dispositivos de seguridad en el plano.

RF9: Exportar a formatos: esta funcionalidad debe permitir exportar a otros formatos como (.pdf, .jpg) luego de realizar la representación sobre una imagen de un plano o haber cargado un archivo de representación.

3.3.2 Requisitos No Funcionales (RNF)

Los requisitos no funcionales expresan una propiedad o cualidad que el sistema debe presentar y pueden contener restricciones físicas sobre los requisitos funcionales, como restricciones de tiempo, acerca del proceso de desarrollo y estándares. Los requerimientos no funcionales generalmente son aplicados al sistema en su totalidad (Sommerville, 2005).

Requisitos de usabilidad

RNF1: El componente deberá poder ser usado por cualquier persona que tenga experiencia utilizando sistemas de video vigilancia o software de representación geográfica.

RNF2: El componente para la representación espacial de dispositivos de seguridad deberá ser una aplicación de escritorio.

Requisitos de confiabilidad

RNF3: Para hacer uso del componente se accederá a través de la autenticación convencional: usuario y contraseña ingresando al módulo Visor.

Restricciones de diseño

RNF4: El componente deberá tener un diseño gráfico sencillo que se identifique con el logo de Suria y los colores del sistema, limitándose al área que ocupa el visualizador del módulo Visor.

RNF5: El lenguaje de programación a utilizar en la implementación del componente será C++, con el framework de desarrollo Qt 4.8.

Requisitos de interfaz

RNF6: El componente deberá tener indicadores que permitan conocer al usuario las acciones que puede realizar, a partir de íconos sugerentes, alternativa textual o algún otro elemento que le facilite la orientación al usuario para el trabajo con el mismo.

RNF7: Los mensajes de error deben mostrarse en color rojo al lado del campo erróneo con un texto de ayuda.

Requisitos de hardware

RNF8: Hardware: 512Mb de memoria RAM, pero es recomendado 1GB de memoria RAM; además de un CPU de dos núcleos a 2.4GHz.

Requisitos de software

RNF9: El componente deberá ser multiplataforma para que pueda ser utilizado en los sistemas operativos: Microsoft Windows 2000/NT o superior, distribución de GNU/Linux, Unix o Mac OS X.

Requisitos Legales, de Derecho de Autor y otros.

RNF10: Como producto, se distribuirá amparado bajo las normativas legales establecidas en el registro comercial emitido por las entidades jurídicas de la Universidad de las Ciencias Informáticas.

3.4 Modelo del sistema

3.4.1 Modelo de Casos de Uso

Un modelo de casos de uso es un modelo del sistema que contiene actores, casos de uso y las relaciones que se derivan entre estos. Permite presentar el modelo en diagramas que muestran a los actores y a los casos de uso desde diferentes puntos de vista y con diferentes propósitos (Jacobson, 2000).

Descripción de los actores

Luego de haberse definido los requisitos tanto funcionales como no funcionales que el sistema debe cumplir, corresponde definir qué actores intervienen en el sistema.

Actores del sistema

Se encargan de inicializar los casos de uso e intercambian directamente con el sistema. Cada actor del sistema puede asumir un conjunto coherente de papeles, en donde un mismo actor puede desempeñar diferentes roles (Jacobson, 2000).

Actor Operador: Es la persona que interactúa con el módulo Visor haciendo uso de las funciones brindadas por el sistema.

Casos de Uso del Sistema

Un CUS[17] es un fragmento de una funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores, especificando una secuencia de acciones o un determinado comportamiento que el

[17] Caso de Uso del Sistema

sistema puede llevar a cabo durante la interacción con los actores, incluyendo alternativas dentro de la propia secuencia (Jacobson, 2000).

3.4.2 Diagrama de Casos de Uso del Sistema

Un DCUS[18] representa gráficamente a los procesos y sus interacciones con los actores. La Figura 6 muestra el diagrama de casos de uso del componente para la representación espacial de dispositivos de seguridad.

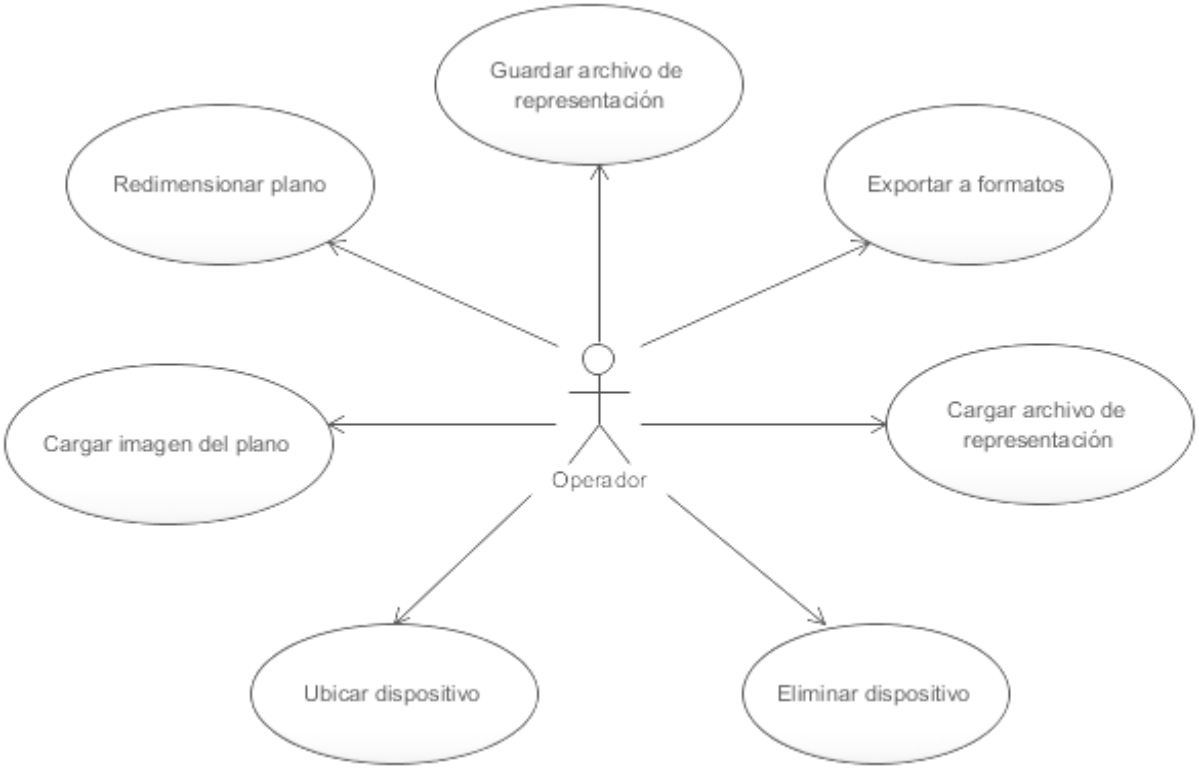


Figura 6 Diagrama de Casos de Uso del Sistema

[18] Diagrama de Caso de Uso del Sistema

3.4.3 Descripción textual de los casos de uso del sistema

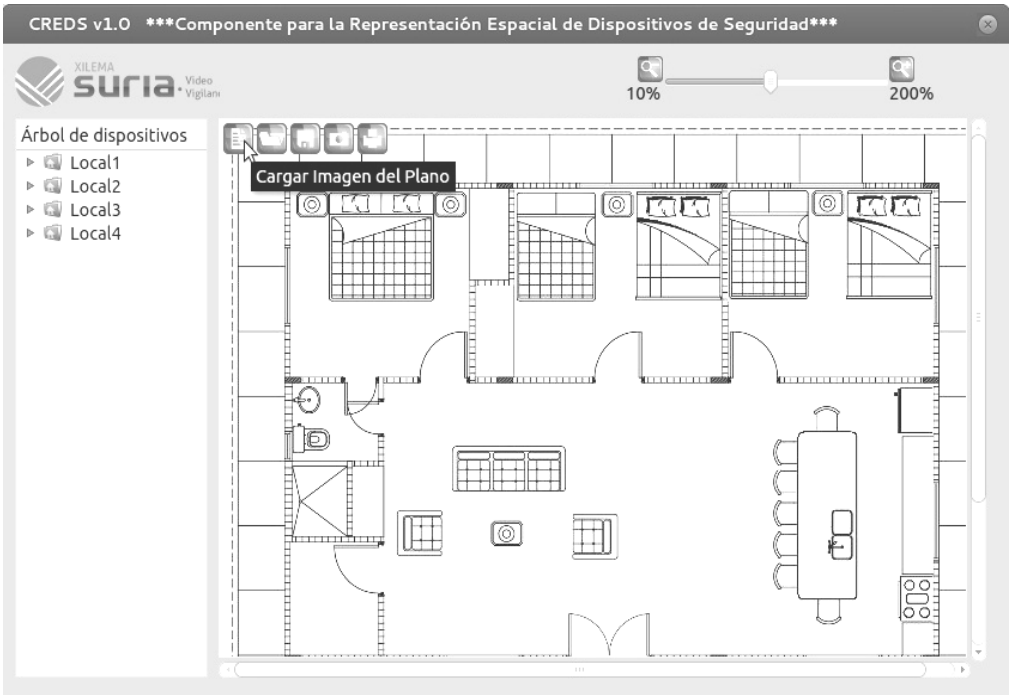
CUS "Cargar imagen del plano"

Tabla 1 CUS "Cargar imagen del plano"

Objetivo	Cargar la imagen del plano donde luego se ubicarán los dispositivos de seguridad para realizar los procesos de representación.	
Actores	Operador (Inicia)	
Resumen	El caso de uso inicia cuando el operador del visor accede a la opción "Cargar imagen del plano" contenida en la barra de herramientas del editor. El mismo debe seleccionar la imagen del plano a partir de una ventana para cargar archivos. El sistema carga la imagen del plano a partir de la dirección especificada y la visualiza en el editor con su tamaño de origen finalizando el caso de uso.	
Complejidad	Media	
Prioridad	Media	
Precondiciones	La imagen deberá cumplir con alguna de las siguientes extensiones (.jpg, .png, .jpeg, .bmp) y contener un plano arquitectónico.	
Postcondiciones	El sistema carga y visualiza en el editor la imagen del plano para comenzar los procesos de representación.	
Flujo de eventos		
Flujo normal "Cargar imagen del plano"		
Actor		**Sistema**
1.	Accede a la opción "Cargar imagen del plano" que pertenece a la barra de herramientas del editor.	Muestra una ventana para cargar archivos con un filtro especificado para imágenes.
2.	Encuentra la imagen del plano y la selecciona.	El sistema ofrece las opciones: • Abrir • Cancelar

3.	Si el operador selecciona la opción “Abrir”	El sistema carga y visualiza la imagen en su tamaño original en el editor a partir de la dirección especificada. Luego de esta acción se cierra la ventana de cargar archivos. ***(ver Interfaz 1)***

Interfaz 1



Flujo alterno “Cargar imagen del plano”

Actor		**Sistema**
3.	Si el operador selecciona la opción “Cancelar”	El sistema cancela el proceso de cargar la imagen del plano y cierra la ventana de cargar archivos.
Relaciones	**CU Incluidos**	No tiene.
	CU Extendidos	No tiene.

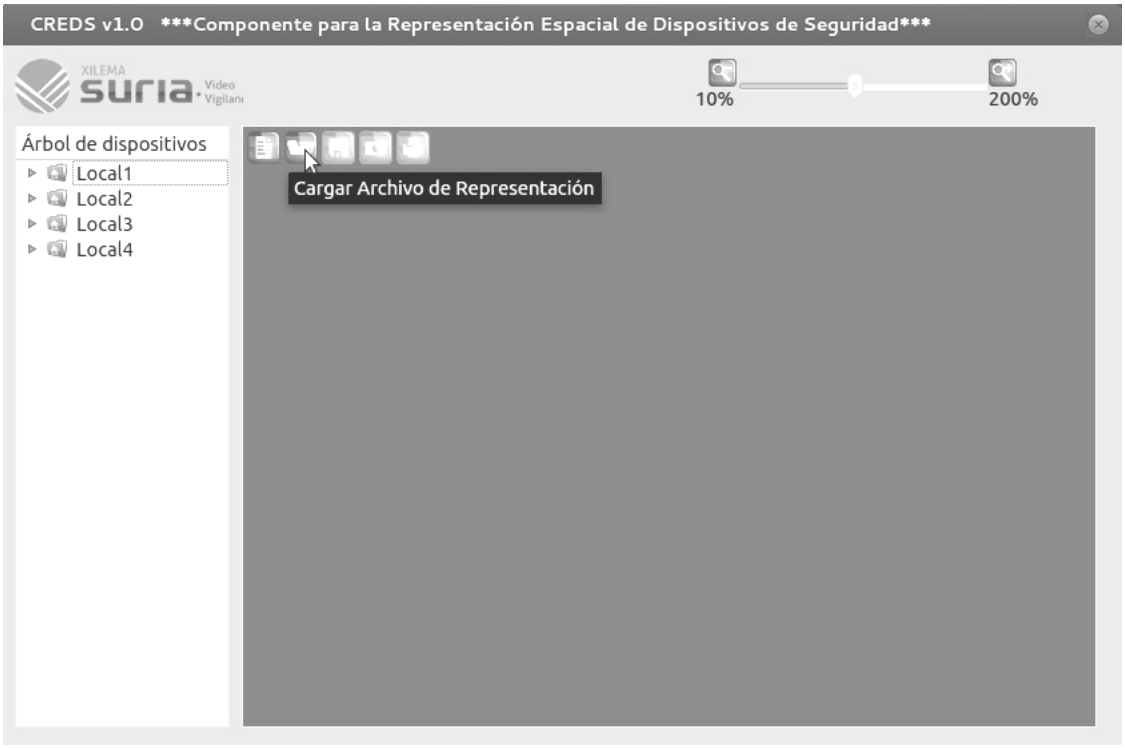
CUS “Cargar archivo de representación”

Tabla 2 CUS “Cargar archivo de representación”

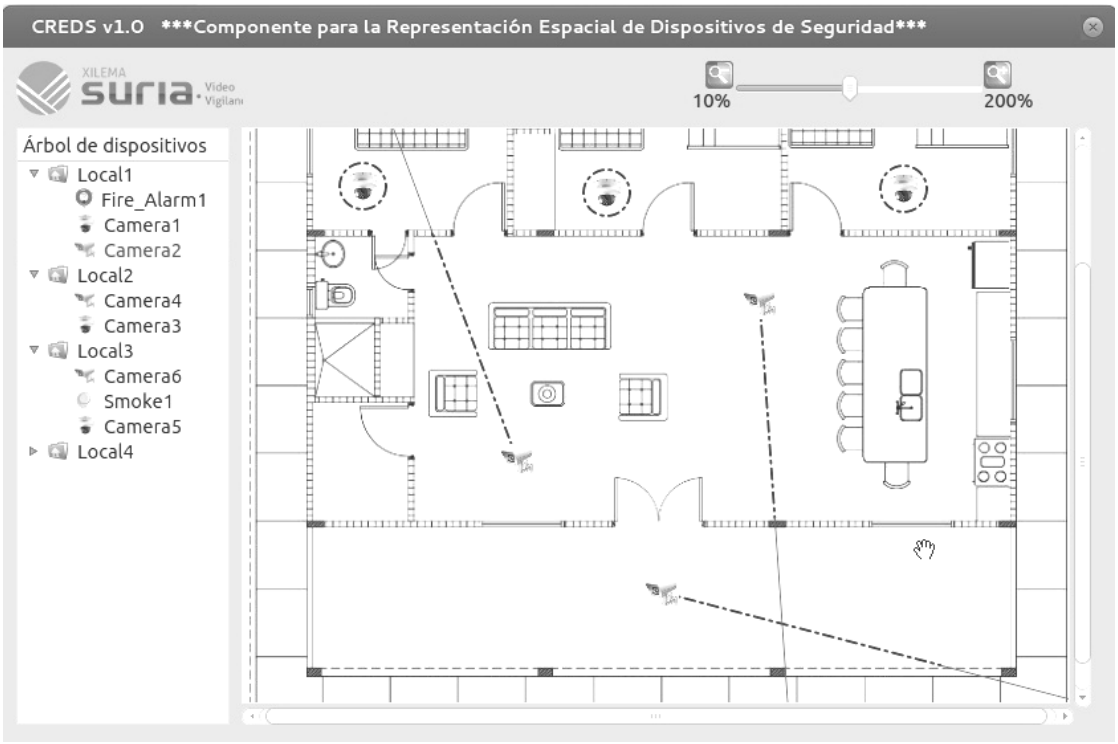
Objetivo	Cargar el archivo de representación con toda la información de representación almacenada, brindando la posibilidad de editar el mismo nuevamente.	
Actores	Operador (Inicia)	
Resumen	El caso de uso inicia cuando el operador del visor accede a la opción “Cargar Archivo de Representación” perteneciente a la barra de herramientas del editor. El mismo debe seleccionar dicho archivo a partir de una ventana para cargar archivos. El sistema carga el archivo de representación a partir de la dirección especificada y visualiza en el editor la representación contenida en el mismo. Luego de esto finaliza el caso de uso.	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones	El archivo de representación deberá cumplir con la siguiente extensión (.drf).	
Postcondiciones	El sistema carga el archivo de representación brindando la posibilidad de ser editado nuevamente.	
Flujo de eventos		
Flujo normal “Cargar archivo de representación”		
Actor		**Sistema**
1.	Accede a la opción “Cargar Archivo de Representación” que pertenece a la barra de herramientas del editor. ***(ver Interfaz 2)***	Muestra una ventana para cargar archivos con un filtro especificado (.drf).
2.	Selecciona el archivo de representación.	El sistema ofrece las opciones: • Abrir • Cancelar

3.	Si el operador selecciona la opción “Abrir”.	El sistema carga el archivo de representación a partir de la dirección especificada visualizando el plano y la información de representación en el editor. Luego de esta acción se cierra la ventana de cargar archivos. ***(ver Interfaz 3)***

Interfaz 2



Interfaz 3



Flujo alterno "Cargar archivo de representación"		
Actor		**Sistema**
3.	Si el operador selecciona la opción "Cancelar"	El sistema cancela el proceso de cargar el archivo de representación y cierra la ventana de cargar archivos.
Relaciones	**CU Incluidos**	No tiene.
	CU Extendidos	No tiene.

CUS "Guardar archivo de representación"

Tabla 3 CUS "Guardar archivo de representación"

Objetivo	Almacenar la información derivada de los procesos de representación realizados sobre el plano en un archivo con extensión (.drf).
Actores	Operador (Inicia)

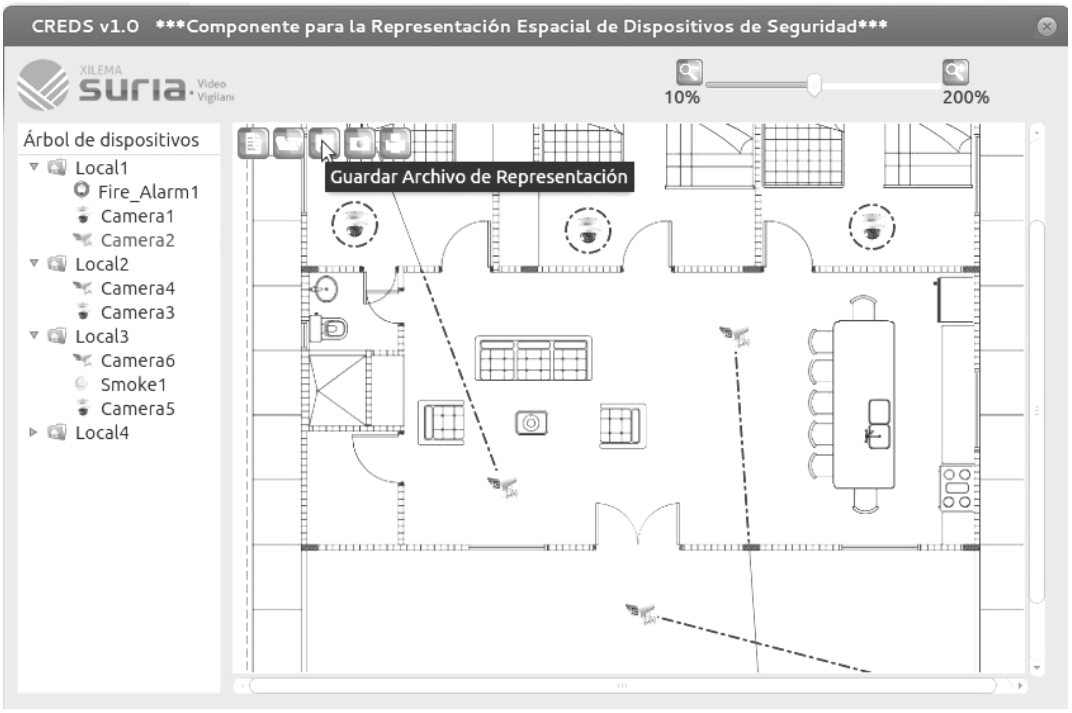
Resumen	El caso de uso inicia cuando el operador del visor accede a la opción “Guardar Archivo de Representación” contenida en la barra de herramientas del editor. El mismo debe especificar el nombre del fichero en la ventana para salvar archivos. El sistema almacena en un fichero con extensión (.drf) la representación de los dispositivos sobre el plano en la dirección especificada finalizando el caso de uso.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	Debe haberse inicializado al menos uno de los casos de uso CU Cargar imagen del plano o CU Cargar archivo de representación.
Postcondiciones	El sistema guarda el archivo de representación con la información almacenada de los procesos de representación con extensión (.drf).

Flujo de eventos

Flujo normal “Guardar archivo de representación”

	Actor	**Sistema**
1.	Accede a la opción “Guardar Archivo de Representación” que pertenece a la barra de herramientas del editor.	Muestra una ventana para salvar archivos con una extensión (.drf).
2.	Asigna un nombre para el archivo de representación.	El sistema ofrece las opciones: • Guardar • Cancelar
3.	Si el operador selecciona la opción “Guardar”	El sistema guarda el archivo de representación con el nombre y en la dirección especificada. Luego de esta acción se cierra la ventana de salvar archivos. ***(ver Interfaz 4)***

Interfaz 4



Flujo alterno "Guardar archivo de representación"

Actor		Sistema
3.	Si el operador selecciona la opción "Cancelar"	El sistema cancela el proceso de guardar el archivo de representación y se cierra la ventana de salvar archivos.
Relaciones	**CU Incluidos**	No tiene.
	CU Extendidos	No tiene.

CUS “Redimensionar plano”

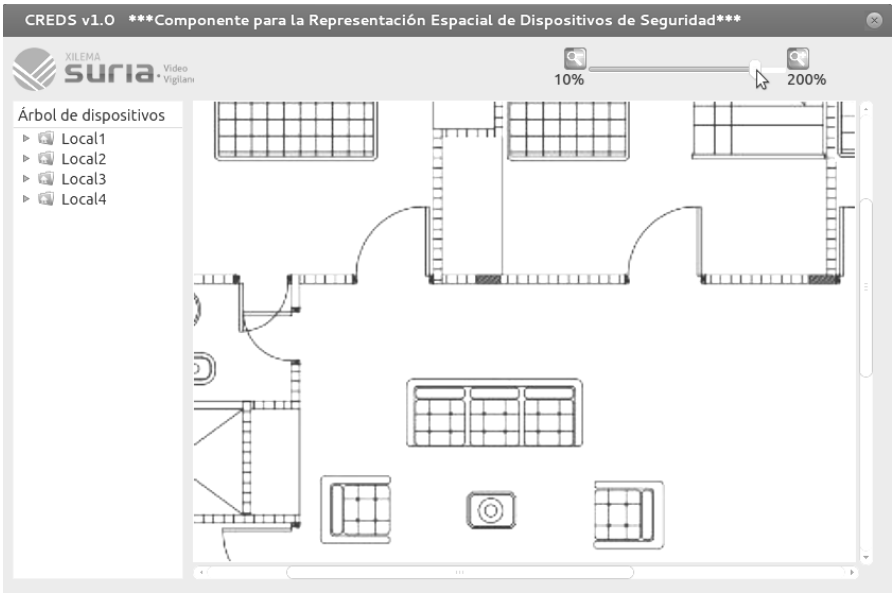
Tabla 4 CUS “Redimensionar plano”

Objetivo	Llevar el plano a diferentes escalas para facilitar los procesos de representación.	
Actores	Operador (Inicia)	
Resumen	El caso de uso inicia cuando el operador del visor regula la escala del plano de acuerdo a su preferencia mediante el deslizador para realizar zoom. La posibilidad de redimensionar el plano ofrece una visión más clara al operador del visor a la hora de realizar los procesos de representación aumentando o disminuyendo las dimensiones del plano.	
Complejidad	Media	
Prioridad	Normal	
Precondiciones	Debe de haberse inicializado al menos uno de los casos de uso CU Cargar imagen del plano o CU Cargar archivo de representación.	
Postcondiciones	El sistema redimensiona el tamaño del plano.	
Flujo de eventos		
Flujo normal “Redimensionar plano”		
Actor		**Sistema**
1.	Utiliza el deslizador de zoom que se encuentra en la parte superior derecha de la interfaz principal.	Muestra por defecto al deslizador en un 100% de escala.

2.	Regula a partir del deslizador un porciento de escala para el plano.	El sistema visualiza en el editor el plano redimensionado teniendo en cuenta el porciento de escala seleccionado por el operador. ***(ver Interfaz 5 e Interfaz 6)***

Interfaz 5

Interfaz 6



Relaciones	**CU Incluidos**	No tiene.
	CU Extendidos	No tiene.

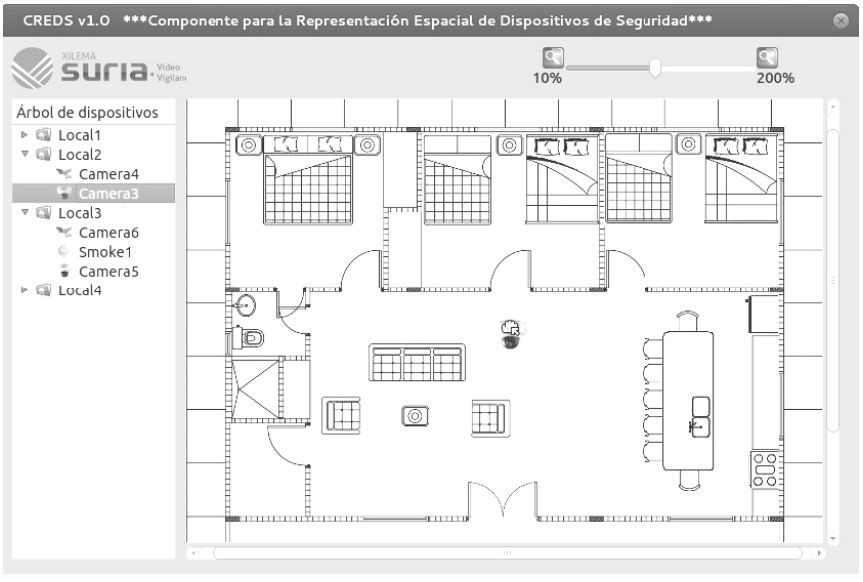
CUS "Ubicar dispositivo"

Tabla 5 CUS "Ubicar dispositivo"

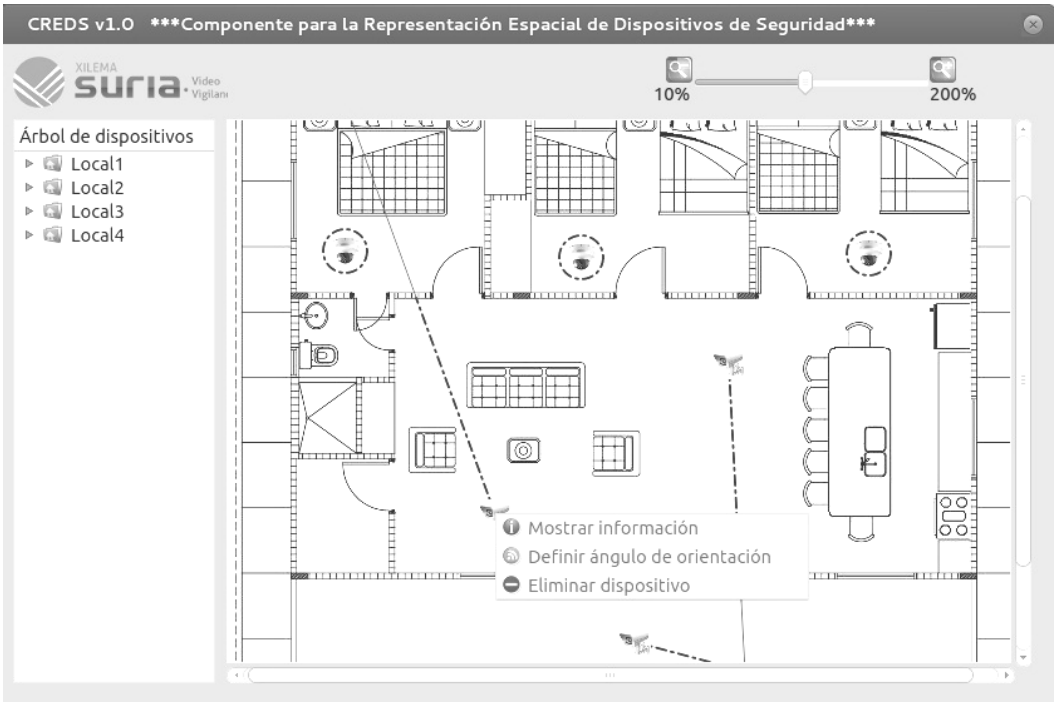
Objetivo	Ubicar sobre el plano los dispositivos de seguridad para realizar los procesos de representación, teniendo en cuenta la información contenida en el mismo.	
Actores	Operador (Inicia)	
Resumen	El caso de uso inicia cuando el operador del visor selecciona del Árbol de dispositivos un dispositivo que es arrastrado con el mouse hacia el editor hasta ubicarlo en una región del plano, para brindar luego las posibilidades de mostrar su información o definir un ángulo de orientación en caso de que sea una cámara fija.	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones	Debe haberse cargado correctamente la imagen del plano o el archivo de representación a partir de los casos de uso CU Cargar imagen del plano o CU Cargar archivo de representación.	
Postcondiciones	El sistema ubica sobre el plano los dispositivos de seguridad que han sido arrastrados hacia el editor en regiones válidas o no válidas según las preferencias del operador.	
Flujo de eventos		
Flujo normal "Ubicar dispositivo"		
Actor		**Sistema**
1.	Arrastra hacia el editor cualquier dispositivo de seguridad que esté registrado en el Árbol de dispositivos.	El sistema verifica si es correcta la ubicación del dispositivo arrastrado. En caso de interceptar algún contenido de la imagen muestra un mensaje de ayuda con el siguiente texto: "El dispositivo ha sido ubicado en un área del plano donde se encuentran posibles paredes u otros objetos"

		¿Está seguro que desea ubicarlo en esta posición? • Si • No
2.	Selecciona la opción “Si”	El sistema ubica el dispositivo en la posición inicial escogida por el operador. ***(ver Interfaz 7)***
3.	Presiona clic derecho sobre el dispositivo de seguridad ubicado en el plano.	El sistema muestra un menú de opciones a realizar en dependencia del tipo de dispositivo ubicado. ***(ver Interfaz 8)*** • Mostrar información. ***(ver Sección 1).*** • Definir ángulo de orientación.***(ver Sección2)*** • Eliminar dispositivo***. (ver CU Eliminar dispositivo)***
Flujo alterno “Ubicar dispositivo”		
Actor		**Sistema**
2.	Selecciona la opción “No”	El sistema cierra la ventana de notificación y cancela el proceso de ubicar el dispositivo.

Interfaz 7



Interfaz 8



Sección 1: "Mostrar información"

Flujo Normal

	Actor	**Sistema**
1.	Selecciona la opción "Mostrar información".	El sistema muestra una tabla con toda la información referente al dispositivo de seguridad ***(ver Interfaz 9)***. La misma contiene los siguientes campos: • Imagen que lo identifica. • Nombre. • Tipo de dispositivo. • Posición en el eje x. • Posición en el eje y. • Ángulo de orientación. • Zona a la que pertenece.

2.	Cierra la ventana.	Se cierra la ventana que contiene la información.

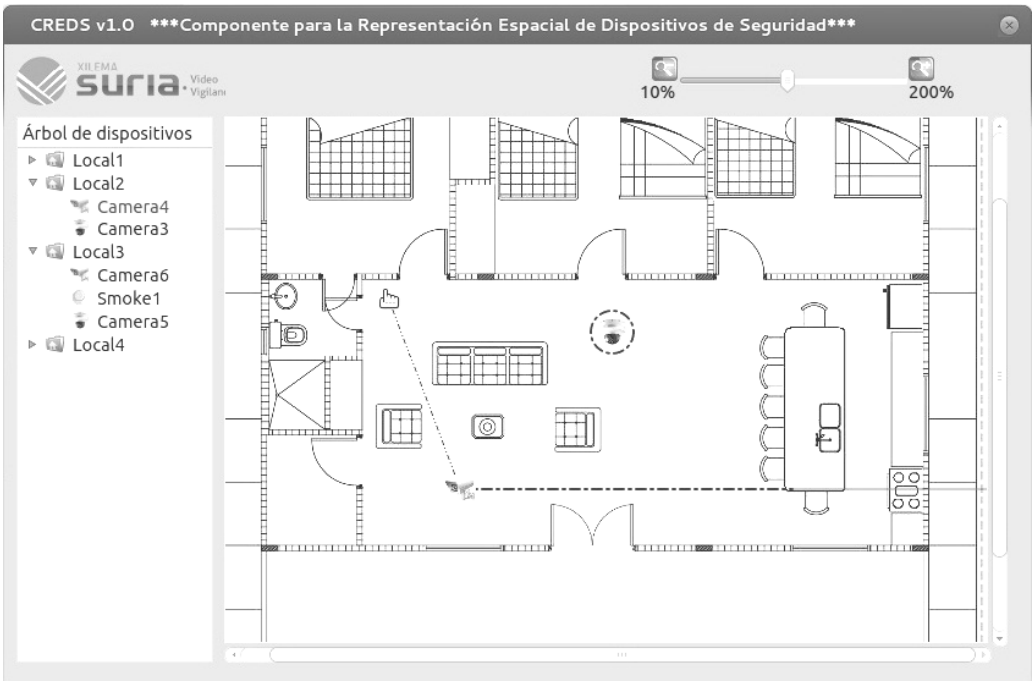
Interfaz 9

Sección 2: "Definir ángulo de orientación"

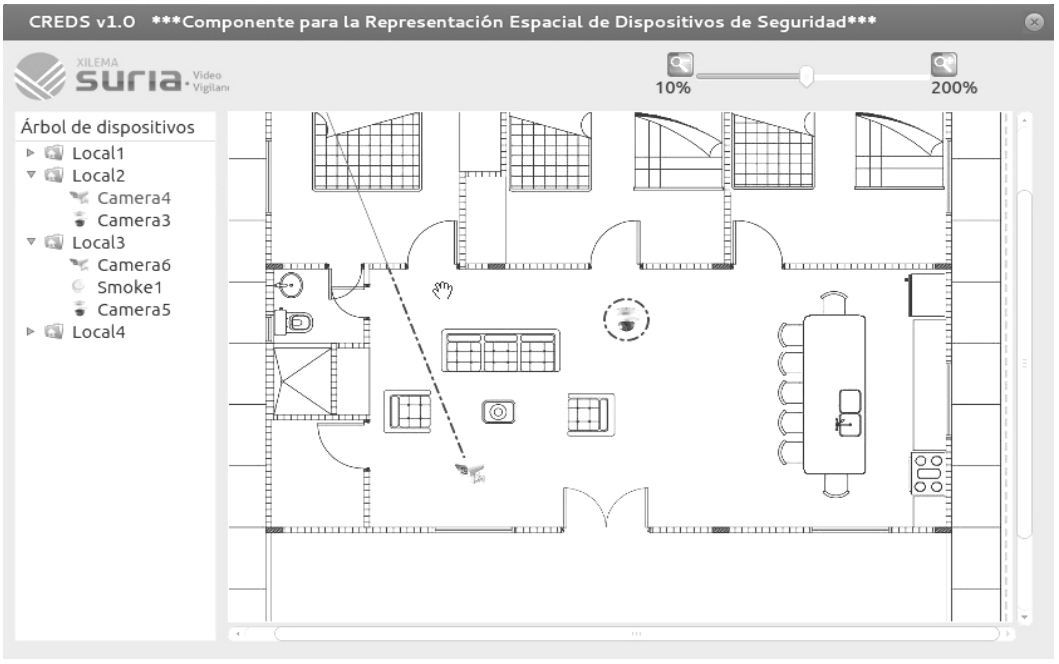
Flujo Normal

	Actor	**Sistema**
1.	Selecciona la opción "Definir ángulo de orientación", si es una cámara IP fija.	El sistema permite redefinirle el ángulo de orientación a la cámara IP fija mediante una línea discontinua que se pinta dinámicamente desde el centro del dispositivo hasta el cursor del mouse.
2.	Define la orientación de la cámara IP fija al accionar el clic izquierdo del mouse. ***(ver Interfaz 10)***	A partir del sentido definido por el operador el sistema redefine el ángulo de orientación de la cámara IP fija. La línea es pintada de color azul discontinua desde el dispositivo hasta el primer punto de intercepción y de color rojo continua desde dicha intercepción hasta el borde del plano. ***(ver Interfaz 11)***

Interfaz 10



Interfaz 11

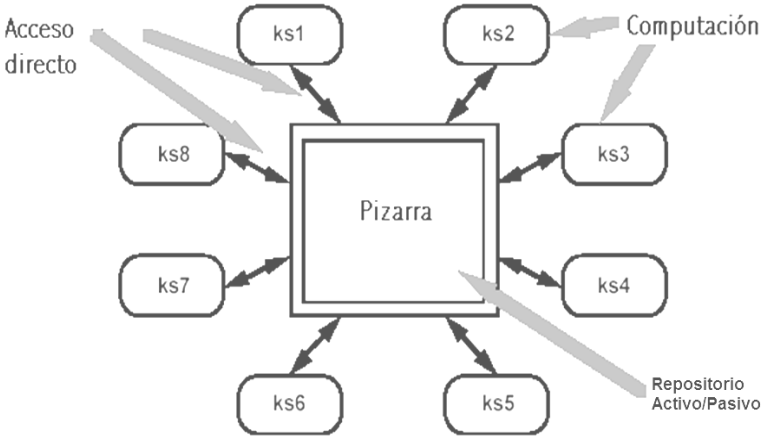


Flujo Alterno "Definir ángulo de orientación"		
2	Acciona sobre el clic derecho del mouse.	El sistema cancela el proceso de redefinir el ángulo de orientación.
Relaciones	**CU Incluidos**	No tiene.
	CU Extendidos	No tiene.

3.5 Descripción de la Arquitectura del Sistema Suria

Es válido aclarar que la solución que se propone será integrada al módulo Visor, el cual forma parte de Suria. Este sistema está diseñado para seguir una arquitectura primaria en forma de pizarra, en su variante de tablero de control. Esta arquitectura permite descomponer el sistema en agentes autónomos que de forma independiente se encargan de realizar atómicamente sus propias funciones, comunicándose entre sí a través de la pizarra que representa el estado actual del proceso. Estos agentes dependen de la entrada de información externa para generar un resultado que puede ser procesado por otro agente y así sucesivamente hasta que de dicha cooperación resulte una solución adecuada (Semanat, 2009).

El módulo Gestor cumple tareas como repositorio activo que entrega y recibe información de los agentes coordinando su funcionamiento, en donde el módulo Visor será un agente autónomo al igual que los otros módulos que forman parte del sistema. Las interacciones entre el repositorio y los demás componentes puede ser variable donde la entrada de los datos es seleccionada por los componentes o el estado actual de los datos del repositorio o sea la pizarra selecciona el proceso a ejecutar.



(Reynoso, 2004)

Figura 7 Arquitectura Suria

Esta arquitectura ofrece beneficios en cuanto a la modularización de Suria mediante la integración de agentes que permiten la resolución de problemas no deterministas. En cuanto a flexibilidad y posibilidades de evolución, debido a que cada agente puede ser flexible a los cambios externos realizados en otros agentes sin afectar su funcionamiento.

3.6 Estilo arquitectónico

Llamada y retorno:

Los estilos de llamada y retorno se enfatizan en la modificabilidad y la escalabilidad. Son los estilos más generalizados en sistemas en gran escala. Miembros de esta familia son las arquitecturas de programa principal y subrutina, los sistemas basados en llamadas a procedimientos remotos, los sistemas orientados a objeto y los sistemas jerárquicos en capas (Reynoso, 2004).

3.7 Patrón arquitectónico

Arquitectura en Capas:

Es una arquitectura organizada jerárquicamente en capas, donde cada capa provee servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inferior. La interacción entre las capas está limitada solo a las que son adyacentes (Reynoso, 2004).

Dada la clara separación entre los diferentes aspectos de desarrollo, la utilización del estilo en capas brinda un conjunto de ventajas, tales como:

- La descomposición del problema en una secuencia de pasos incrementales, basado en niveles de abstracción.
- Soporta la mejora ante posibles optimizaciones y refinamientos, en donde los cambios solo afectarían a las capas vecinas.
- Proporciona amplia reutilización debido a que se pueden cambiar las implementaciones respetando las interfaces con las capas adyacentes.

Arquitectura en 2 Capas:

Para definir una estructura durante el desarrollo del componente para la representación espacial de dispositivos de seguridad se seleccionó dicha arquitectura en su variante de dos capas, las cuales son la de presentación y la de negocio y datos.

Presentación: en esta capa se tratan los aspectos gráficos de la aplicación y es donde se realiza la interacción con el operador. El uso de la capa de presentación permite la captura de datos ingresados por el usuario y visualiza la información solicitada. Además la misma se encarga de mostrar las notificaciones

de alerta ante la ocurrencia de errores, asegurando la validación de los mismos durante la utilización del componente.

Negocio y Datos: en esta capa se establecen todas las reglas que deben cumplirse manteniendo comunicación con la capa de presentación, para recibir las solicitudes y presentar los resultados. Contiene las clases donde se realizan los procesos de transformación y detección de líneas por los cuales atraviesa la imagen del plano. Se encuentra una clase entidad y otra clase para manipular los ficheros generados por el componente.

Figura 8 Patrón Arquitectónico del Componente

3.8 Patrones de diseño:

Los patrones de diseño son soluciones simples con una validez demostrada ante la ocurrencia de problemas comunes de programación orientada a objetos. La aplicación de estos patrones en la programación forma parte de las buenas prácticas que pueden ser empleadas durante el desarrollo de un software (Pavón, 2004). Teniendo en cuenta el patrón arquitectónico seleccionado, los patrones de diseño que se aplican antes de la implementación del componente son los siguientes:

GRASP[19]

- **Alta Cohesión:** la cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Los componentes visuales que forman parte de la interfaz visual del sistema fueron separados por clases (*ToolCollection, Editor, BlueprintPainter, DeviceInfo*), así como las dedicadas al procesamiento de imagen y detección de líneas contenidas en el plano (*ImageProcessor, Detector*) asegurando que las mismas solo cuenten con las responsabilidades

[19] Patrones Generales de Software para Asignación de Responsabilidades (*General Responsibility Asignment Software Patterns*)

que deban asumir. De esta forma se evita un trabajo excesivo en las clases y se logra mantener una alta cohesión.

- **Creador:** guía la asignación de responsabilidades relacionadas con la creación de objetos, encontrando un creador que debemos conectar con el objeto producido en cualquier evento. La clase *MainView* contiene y relaciona objetos que representan otras clases haciendo uso de sus propiedades y funciones, por lo que se evidencia que dicha clase es un "creador".
- **Experto:** se le asigna la responsabilidad al experto en información, o sea a la clase que cuenta con la información necesaria para cumplir la responsabilidad. La clase *BlueprintPainter* contiene la información de los dispositivos que se encuentran ubicados en el plano así como toda la representación asociada a éstos, asumiendo responsabilidad durante las operaciones de representación, por solo citar un ejemplo.
- **Bajo Acoplamiento:** el acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y recurre a ellas. La clase *ImageProcessor* solo depende de la clase *Detector,* y recurre a operaciones de la misma luego de finalizar el procesamiento de la imagen para garantizar la detección de líneas contenidas en el plano.

GoF[20]

- **Instancia única:** permite exactamente una instancia de una clase; puesto que la visibilidad ante las clases tiene un ámbito relativamente global, todo objeto puede llamar directamente al método de la clase. La clase *FileManager* contiene métodos de acceso global para realizar operaciones con los ficheros generados y cargados por el componente luego de haber finalizado los procesos de representación.
- **Observador:** define una dependencia de uno a muchos entre objetos, de tal forma que cuando el objeto cambie de estado, todos sus objetos dependientes sean notificados automáticamente. El sistema de ranuras y señales de Qt constituye una implementación del patrón de diseño Observador.

[20] Pandilla de los cuatro (*Gang of Four*)

3.9 Conclusiones

A partir de la realización y cumplimiento de las actividades que pertenecen al flujo de trabajo Requisitos se hizo posible definir los requisitos funcionales del sistema y las restricciones correspondientes para su desarrollo. Luego de ser precisadas estas características se seleccionó una arquitectura en capas que garantizó definir una estructura lógica para el sistema, asegurando una mejor organización e interrelación armónica para sus componentes. Los patrones de diseño aplicados ayudaron a complementar la arquitectura del sistema evidenciando el uso de soluciones demostradas como parte de las buenas prácticas de la programación.

CAPÍTULO 4: CONSTRUCCIÓN Y VALIDACIÓN DE LA SOLUCIÓN PROPUESTA.

4.1 Introducción

Una vez que son definidas las características y la arquitectura del sistema se comienzan a realizar las tareas constructivas del software como parte del proceso de desarrollo. Durante esta etapa es necesario tener en cuenta elementos importantes como, el refinamiento de los requisitos en caso de ser necesario, la arquitectura definida para el sistema asociada al marco de trabajo utilizado y la forma de almacenar o gestionar la información con la cual se necesita trabajar. Para darle un seguimiento a lo propuesto por RUP, el flujo de trabajo Análisis se encarga de detallar los requisitos cuando éstos resultan complejos, para lograr una mejor comprensión y estructuración de los mismos. Debido a las razones expuestas en este acápite no fue necesario desarrollar este flujo de trabajo, por lo que se prescindió de su realización.

Teniendo en cuenta la metodología seleccionada se prosigue con el Diseño, donde se representan mediante diagramas las relaciones de clases y sus responsabilidades bajo la arquitectura definida. Este flujo asegura un punto de partida para iniciar la implementación, donde se debe obtener como resultado el código fuente de la aplicación y el modelo de implementación. Luego de ser implementado el componente se necesita elevar la calidad del mismo y validar que todas sus funcionalidades son operativas.

Para lograr dicho objetivo es necesario realizarle pruebas haciendo uso de varias técnicas de prueba. La realización de las pruebas permite detectar errores en el componente asegurando que durante su ciclo de vida o al finalizar cada iteración de desarrollo se corrijan los problemas identificados. En el contenido del capítulo se exponen diferentes elementos asociados a la construcción y a la validación del componente que permiten reflejar con claridad las actividades realizadas durante esta etapa.

4.2 Modelo de Diseño

Para lograr la transición al Diseño normalmente se utiliza el flujo de trabajo Análisis debido a que según lo propuesto por RUP este le antecede durante el proceso de desarrollo de software. El flujo de trabajo del Análisis tiene dos objetivos fundamentales: refinar y estructurar los requisitos, partiendo de que los mismos son analizados con mayor profundidad y son estructurados mediante clases de análisis y paquetes.

El modelo de análisis es un artefacto generado en esta etapa que facilita el mantenimiento de los requisitos. Puede ser utilizado por los desarrolladores para saber cómo debe ser diseñado e

implementado el sistema, debido a que este modelo proporciona la estructura interna del mismo. Este modelo ayuda a estructurar la arquitectura del sistema y sirve como primera aproximación al Diseño.

Una variante válida es no utilizar en absoluto el modelo del análisis para describir los resultados del Análisis, producto a que los requisitos se analizan como parte integrada de la captura de requisitos, lo que trae como exigencia durante el proceso un mayor formalismo en la realización del modelo de casos de uso (Jacobson, 2000).

La metodología RUP comprende un proceso configurable, lo que posibilita no hacer uso extensivo de todas las actividades y entregables definidos en la misma. Partiendo de que dicho proceso puede ser adaptable tomando en cuenta solo aquellas partes que se consideren necesarias, por lo que se decidió prescindir de la realización del modelo de análisis en el desarrollo de esta investigación debido a que:

- Es posible obtener un mayor formalismo en el modelo de casos de uso pues el cliente es capaz de comprender los resultados que estos pueden proporcionar.
- Los requisitos son simples, bien conocidos y se cuenta con cierta comprensión de los mismos.
- No se requiere este artefacto en el proyecto VVI.
- Se logra evitar el costo en tiempo y recursos de mantener este flujo.

Teniendo en cuenta lo planteado respecto a la no realización del modelo de análisis, queda justificada la transición directa al Diseño.

Durante el flujo de trabajo del Diseño se modela el sistema teniendo en cuenta la arquitectura definida para el mismo y se genera el artefacto modelo del diseño como punto de partida para comenzar las tareas de implementación.

El modelo del diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además, sirve de abstracción de la implementación del sistema y es, de ese modo, utilizada como una entrada fundamental de las actividades de implementación. En el modelo de diseño, los casos de uso son realizados por las clases de diseño y sus objetos (Jacobson, 2000).

Para realizar el modelo del diseño hay que tener en cuenta el lenguaje de programación que será empleado por lo que a continuación se realiza una descripción general del funcionamiento del marco de trabajo Qt utilizado en la construcción del sistema, para lograr un mejor entendimiento del flujo descrito en los diagramas de diseño.

4.2.1 Descripción de los módulos utilizados de Qt

El marco de trabajo Qt cuenta con una serie de módulos que facilitan el trabajo para realizar operaciones de comunicación con bases de datos como QtSQL, el uso de librearías gráficas de openGL empleando QtOpenGL, un motor web haciendo uso de QtWebKit, la integración con XML mediante QtXML, por solo citar algunos. Durante el desarrollo del componente se utilizan los módulos que se muestran en la Figura 9**:**

Figura 9 Módulos de Qt

- **QtCore:** contiene el núcleo de Qt.
- **QtGui:** colección básica para el uso de componentes gráficos.

4.2.2 Diagrama de clases

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases del diseño y las relaciones entre éstas. En la Figura 10 se muestra el diagrama de clases correspondiente al componente para la representación espacial de dispositivos de seguridad, donde se representan las clases utilizadas para el desarrollo del mismo teniendo en cuenta la estructura del estilo arquitectónico en capas.

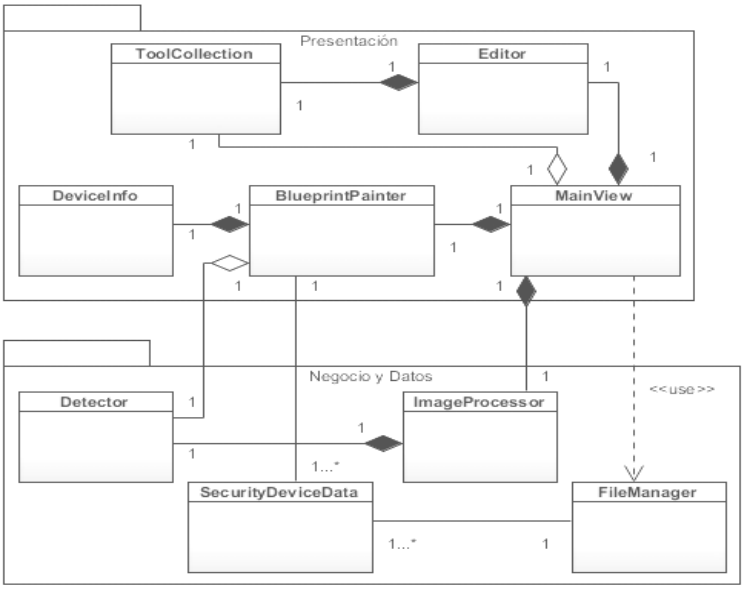


Figura 10 Diagrama de Clases del Diseño

Este diagrama solo contiene las clases y sus relaciones, para observarlo detallado ver **Anexo 1**.

A partir de la Figura 10 se aprecia que las clases son agrupadas por paquetes respetando la arquitectura bajo la que se desarrolla el componente. En la capa de presentación se encuentran clases como MainView, Editor, BlueprinPainter y ToolCollection que en su conjunto conforman la interfaz principal del componente. Esta capa interactúa con el operador durante los procesos de representación y asegura la validez de los mismos. Adyacente a ella se encuentra la capa de Negocio y Datos donde las clases que contiene se encargan de procesar la imagen del plano y facilitar la manipulación con ficheros para almacenar los datos de la representación. La clase ImageProcessor se encarga de realizar las transformaciones a dicha imagen derivando un estado resultante para la misma. Luego este resultado es utilizado por la clase Detector para identificar la mayor cantidad de líneas rectas contenidas en el plano. Asociadas a esta capa se encuentran también la clase entidad SecurityDeviceData y la de FileManager encargada de manipular los archivos de representación generados por el componente.

4.2.3 Diagrama de secuencia

Para describir la interacción entre objetos del diseño es necesario realizar diagramas de secuencia o de colaboración que representen el flujo de eventos para cada caso de uso. Los elementos que se deben encontrar en estos diagramas son las instancias de actores, objetos del diseño y las transmisiones de mensajes. Para representar las interacciones entre los objetos del diseño se realizaron los diagramas de secuencia. A continuación aparecen los diagramas de secuencia que ilustran el flujo de procesos durante la realización de los casos de uso (Jacobson, 2000).

DS CUS Ubicar dispositivo

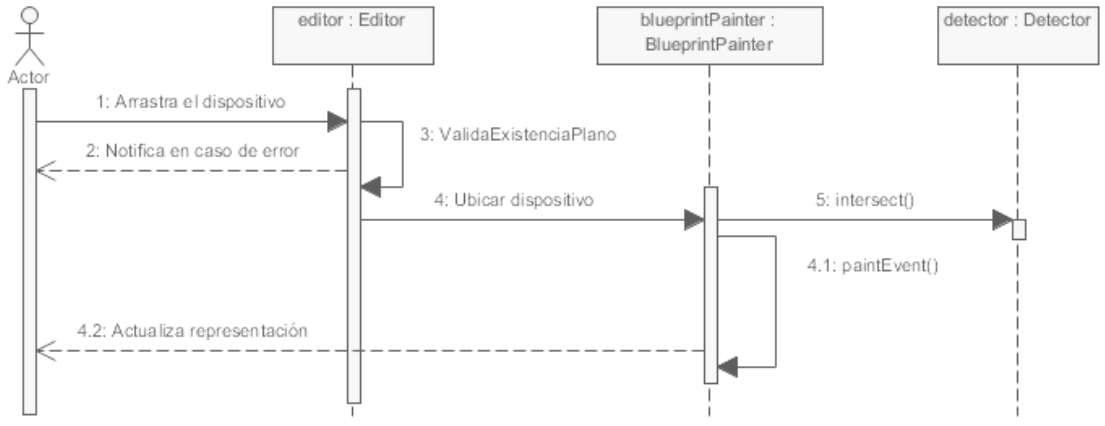


Figura 11 DS CUS Ubicar dispositivo

DS CUS Ubicar dispositivo Sección 1 "Mostrar información"

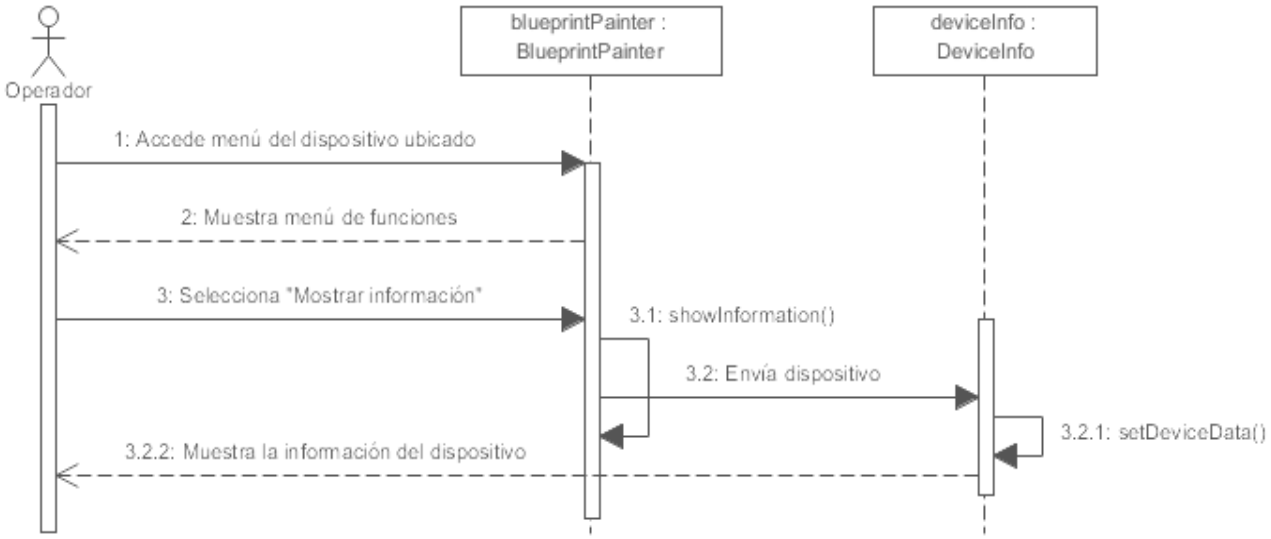


Figura 12 DS CUS Ubicar dispositivo Sección 1 "Mostrar información"

DS CUS Ubicar dispositivo Sección 2 "Definir ángulo de orientación"

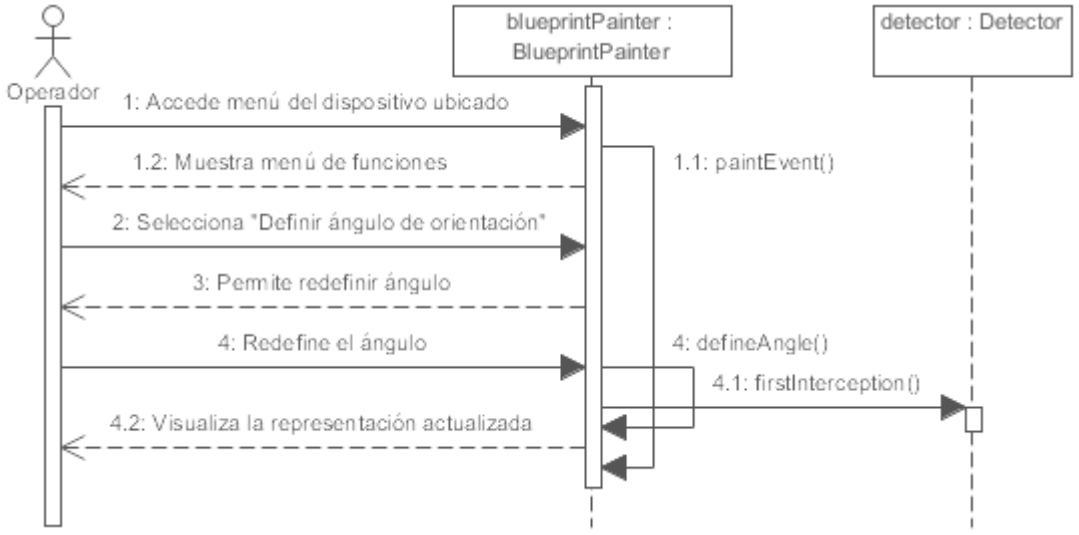


Figura 13 DS CUS Ubicar dispositivo Sección 2 "Definir ángulo de orientación"

DS CUS Guardar archivo de representación

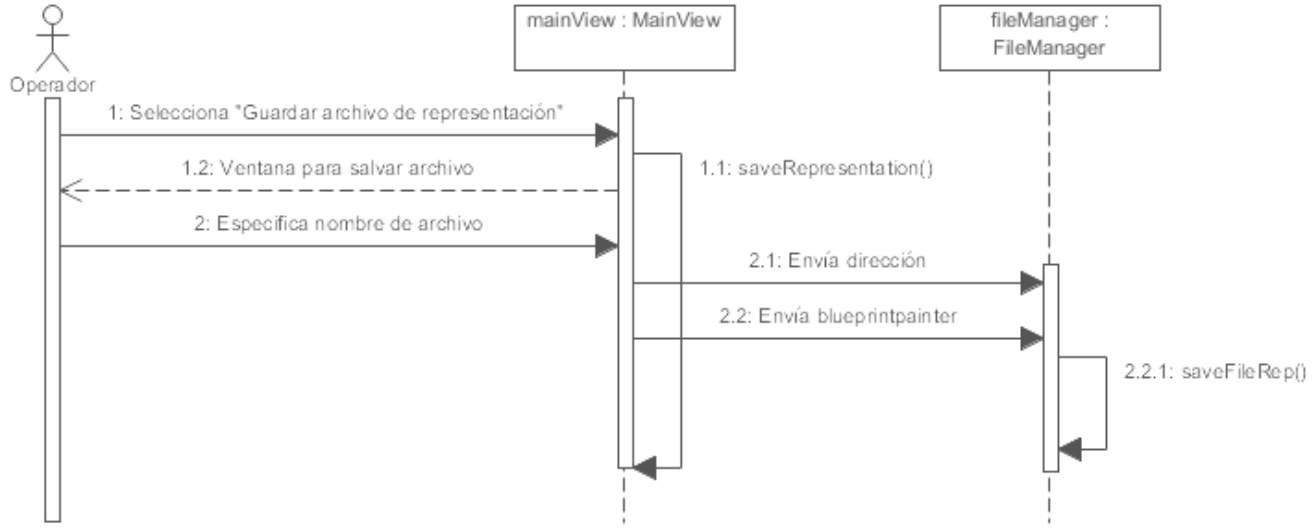


Figura 14 DS CUS Guardar archivo de representación

Para ver los diagramas de secuencia correspondientes a los restantes casos de uso ver **Anexo 2.**

4.3 Modelo de Datos

Un modelo de datos es un sistema formal y abstracto que permite describir los datos de acuerdo con reglas y convenios predefinidos mediante una representación gráfica (Jacobson, 2000). Luego de realizar los procesos de representación sobre la imagen del plano el componente genera un archivo donde se almacena la información de representación serializada. El fichero binario con extensión (.drf) presenta la siguiente estructura que aparece a continuación en la Figura 15**.**

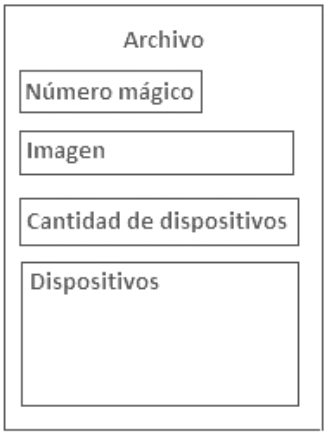


Figura 15 Estructura del Fichero de Representación

Para lograr entender con más claridad la composición del fichero generado por el componente se explican los principales elementos que forman parte de su contenido:

- **Número mágico:** el número mágico es un valor que se utiliza como llave para determinar que el archivo fue generado por el componente demostrando autenticidad por parte del mismo, evitando posibles conflictos a la hora de cargar un archivo de representación.
- **Imagen:** este campo contiene toda la información referente a la imagen del plano arquitectónico sobre la cual se realizó la representación.
- **Cantidad de dispositivos:** permite almacenar la cuantificación de los dispositivos de seguridad que se encuentran representados sobre dicho plano, para luego determinar que dicho indicador se corresponda con la cantidad de dispositivos almacenados en el campo Dispositivos.
- **Dispositivos:** contiene todos los dispositivos de seguridad que fueron representados sobre el plano, así como la información asociada a los mismos.

4.4 Modelo de Despliegue

Es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una influencia principal en su diseño (Jacobson, 2000).

4.4.1 Diagrama de despliegue

En la Figura 16 se representa el diagrama de despliegue para el componente. En la misma se representa un ordenador como estación de visualización donde se encuentra el módulo Visor con el componente incorporado.

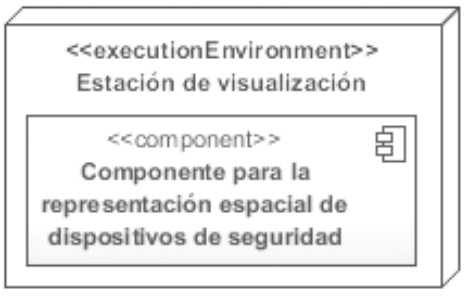


Figura 16 Diagrama de Despliegue

4.5 Modelo de Implementación

Describe cómo los elementos del modelo de diseño se implementan en términos de componentes sin descartar la dependencia que se manifiesta entre éstos, entre los que se pueden encontrar algunos como ficheros de código fuente o ejecutables. Además propone cómo se deben organizar los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación sin dejar de tener en cuenta el lenguaje o lenguajes de programación utilizados (Jacobson, 2000).

4.5.1 Diagrama de componentes

Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en un modelo de diseño. Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes de software, sean estos componentes de código fuente, binarios o ejecutables.

En la Figura 17 se representa el diagrama de componentes de código fuente donde se encuentran los paquetes de componentes y las relaciones entre éstos.

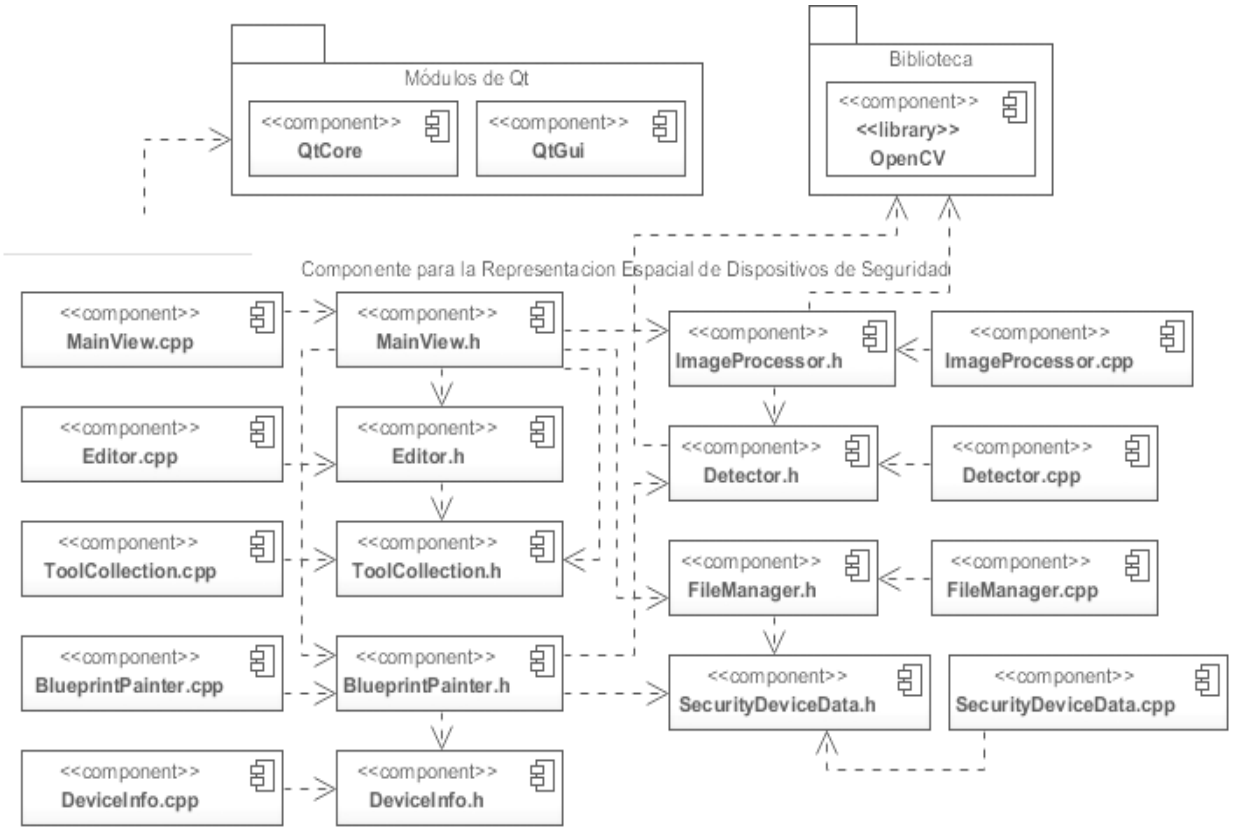


Figura 17 Diagrama de Componentes

4.6 Modelo de pruebas

Describe principalmente cómo se prueban los componentes ejecutables en el modelo de implementación con pruebas de integración y de sistema, comportándose como una colección de casos de prueba, procedimientos de prueba y componentes de prueba (Jacobson, 2000).

4.6.1 Pruebas de la solución

Una vez generado el código fuente es necesario realizar las pruebas del sistema para detectar y corregir la mayor cantidad de errores en busca de elevar la calidad del software. Para lograr dicho objetivo se utilizan técnicas de prueba que permitan diseñar casos de prueba con una alta probabilidad de encontrar errores. Estas técnicas facilitan una guía sistemática para diseñar pruebas que se encargan de comprobar la lógica interna, las interfaces, el comportamiento y el rendimiento del software (Pressman, 2002).

La técnica seleccionada para realizarle las pruebas al componente para la representación espacial de dispositivos de seguridad es la conocida como "caja negra"**.** Las pruebas de caja negra, también denominadas como pruebas de comportamiento, se centran en los requisitos funcionales del software. Son pruebas que se llevan a cabo sobre la interfaz del software, obviando la lógica interna y la estructura del programa. A partir de las pruebas de caja negra se pretenden encontrar los siguientes tipos de errores:

- Funciones incorrectas o ausentes.
- Errores en la interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

4.6.2 Casos de prueba:

Los casos de prueba especifican una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo las cuales debe probarse. Los casos de prueba de caja negra pretenden demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma correcta.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

Para la realización de los casos de prueba se utilizará específicamente el método de prueba partición equivalente. Este método se basa en la evaluación de clases de equivalencia para una condición de entrada, lo que permite representar un conjunto de estados válidos o inválidos. En este caso la condición de entrada es lógica, lo que define una clase de equivalencia válida y una no válida. A continuación se muestran las pruebas realizadas a los CUS (Pressman, 2002).

4.6.3 Diseño casos de prueba al sistema

DCP CUS Cargar imagen del plano

Descripción general:

El caso de prueba inicia cuando se accede a la opción “Cargar imagen del plano” contenida en la barra de herramientas del editor. Se debe seleccionar la imagen del plano a partir de una ventana para cargar archivos. El sistema carga la imagen del plano a partir de la dirección especificada y la visualiza en el editor con su tamaño de origen.

Condiciones de ejecución:

La imagen deberá cumplir con alguna de las siguientes extensiones (.jpg, .png, .jpeg, .bmp) y representar un plano arquitectónico.

DCP CUS Cargar imagen del plano

Tabla 6 DCP CUS Cargar imagen del plano

Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 1.1 Cargar la imagen del plano correctamente.	Al acceder a la opción “Cargar Imagen del Plano” el sistema muestra una ventana para cargar archivos en formatos de imágenes. Luego de seleccionar la imagen del plano y presionar en “Abrir” el sistema cierra la ventana. El sistema verifica que el archivo de la imagen no esté vacío para visualizar la imagen en el editor.	Carga la imagen del plano y la visualiza en el editor.	Paso 1: Clic en la opción “Cargar Imagen del Plano”. Paso 2: Seleccionar la imagen. Paso 3: Clic en “Abrir”.

EC 1.2 Cargar la imagen del plano incorrectamente.	Luego de mostrarse la ventana para cargar archivos se presiona en "Abrir" sin seleccionar la imagen del plano.	No carga la imagen hasta que la misma no sea seleccionada.	Paso 1: Clic en la opción "Cargar Imagen del Plano". Paso 2: Clic en "Abrir".
EC 1.3 Cargar la imagen del plano se cancela.	Luego de mostrarse la ventana para cargar archivos se presiona en "Cancelar".	Cierra la ventana para cargar archivos y cancela el proceso de cargar la imagen del plano.	Paso 1: Clic en la opción "Cargar Imagen del Plano". Paso 2: Clic en "Cancelar".

DCP CUS Cargar archivo de representación

Descripción general:

El caso de prueba inicia cuando se accede a la opción "Cargar Archivo de Representación" perteneciente a la barra de herramientas del editor. Se debe seleccionar dicho archivo a partir de una ventana para cargar archivos. El sistema carga el archivo de representación a partir de la dirección especificada y visualiza en el editor la representación contenida en el mismo.

Condiciones de ejecución:

El archivo de representación deberá cumplir con la extensión (.drf) compatible con el componente.

DCP CUS Cargar archivo de representación

Tabla 7 DCP CUS Cargar archivo de representación

Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 1.1 Cargar archivo de representación correctamente.	Al acceder a la opción "Cargar Archivo de Representación" el sistema muestra una ventana para cargar archivos de representación. Luego de seleccionar el archivo y presionar en "Abrir" el sistema cierra la ventana. Luego verifica que el archivo no sea vacío para visualizar en el editor la representación contenida en el fichero.	Carga el archivo de representación y visualiza en el editor la representación almacenada en el fichero.	Paso 1: Clic en la opción "Cargar Archivo de Representación". Paso 2: Seleccionar el archivo de representación. Paso 3: Clic en "Abrir".

EC 1.2 Cargar el archivo de representación incorrectamente	Luego de mostrarse la ventana de cargar archivos se presiona en "Abrir" sin seleccionar el archivo de representación.	No carga el archivo de representación hasta que el mismo no sea seleccionado.	Paso 1: Clic en la opción "Cargar Archivo de Representación". Paso 2: Clic en "Abrir".
EC 1.3 Cargar el archivo de representación se cancela.	Luego de mostrarse la ventana para cargar archivos se presiona en "Cancelar".	Cierra la ventana para cargar archivos y cancela el proceso de cargar el fichero de representación.	Paso 1: Clic en la opción "Cargar Archivo de Representación". Paso 2: Clic en "Cancelar".

DCP CUS Guardar archivo de representación

Descripción general:

El caso de prueba inicia cuando se accede a la opción "Guardar Archivo de Representación" contenida en la barra de herramientas del editor. Se debe especificar el nombre del fichero en la ventana para salvar archivos. El sistema almacena en un fichero con extensión (.drf) la representación de los dispositivos sobre el plano en la dirección especificada.

Condiciones de ejecución:

Debe haberse cargado correctamente la imagen de un plano o un archivo de representación.

DCP CUS Guardar archivo de representación

Tabla 4 DCP CUS Guardar archivo de representación

Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 1.1 Guardar archivo de representación correctamente.	Al acceder a la opción "Guardar Archivo de Representación" el sistema muestra una ventana para salvar archivos de representación. Luego de especificar un nombre para el mismo se presiona en "Guardar", acto seguido el sistema almacena la información de representación generando un fichero y cierra la ventana.	Genera un fichero con extensión (.drf) que contiene toda la información de representación.	Paso 1: Clic en la opción "Guardar Archivo de Representación". Paso 2: Especificar un nombre para el archivo de representación. Paso 3: Clic en "Guardar".

EC 1.2 Guardar el archivo de representación incorrectamente	Luego de mostrarse la ventana para salvar archivos se presiona en "Guardar" sin especificar un nombre para el archivo de representación.	Cierra la ventana para salvar archivos y muestra un mensaje de advertencia "Escriba un nombre de archivo".	Paso 1: Clic en la opción "Guardar Archivo de Representación". Paso 2: Clic en "Guardar".
EC 1.3 Guardar el archivo de representación se cancela.	Luego de mostrarse la ventana para salvar archivos se presiona en "Cancelar".	Cierra la ventana para salvar archivos y cancela el proceso de guardar el fichero de representación.	Paso 1: Clic en la opción "Guardar Archivo de Representación". Paso 2: Clic en "Cancelar".

DCP CUS Redimensionar plano

Descripción general:

El caso de prueba inicia cuando se regula la escala del plano de acuerdo a su preferencia mediante el deslizador para realizar zoom. La posibilidad de redimensionar el plano ofrece una visión más clara a la hora de realizar los procesos de representación aumentando o disminuyendo las dimensiones del plano.

Condiciones de ejecución:

Debe haberse cargado correctamente la imagen de un plano o un archivo de representación.

DCP CUS Redimensionar plano

Tabla 5 DCP CUS Redimensionar plano

Escenario	Descripción	Respuesta del sistema	Flujo Central
EC1.1 Redimensionar plano correctamente.	Se accede al deslizador de zoom para regular la escala del plano, permite mejorar la visión del mismo para facilitar los procesos de representación.	Visualiza el plano redimensionado en el editor de acuerdo a la escala seleccionada por el operador.	Paso 1: Cursor del mouse sobre el deslizador de zoom.

EC1.2 Redimensionar plano con el deslizador de zoom inhabilitado	El deslizador de zoom queda deshabilitado hasta que no se haya cargado la imagen de un plano o un archivo de representación.	Mantiene deshabilitado el deslizador de zoom.	Paso 1: No puede ser utilizado el deslizador de zoom.

DCP CUS Ubicar dispositivo

Descripción general:

El caso de prueba inicia cuando se selecciona del Árbol de dispositivos un dispositivo que se arrastra con el mouse hacia el editor hasta ubicarlo en una región del plano, para brindar luego las posibilidades de mostrar su información o definir un ángulo de orientación en caso de que sea una cámara fija.

Condiciones de ejecución:

Debe haberse cargado correctamente la imagen de un plano o un archivo de representación.

DCP CUS Ubicar dispositivo

Tabla 6 DCP CUS Ubicar dispositivo

Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 1.1 Ubicar dispositivo correctamente.	Se selecciona del Árbol de dispositivos un dispositivo que es arrastrado hacia el editor hasta ubicarlo en una región del plano. El sistema verifica que el dispositivo no haya sido ubicado con anterioridad y verifica si el dispositivo se intercepta con algún contenido del plano. En dependencia del tipo de dispositivo ubicado se le asocia una información de representación.	Verifica si el dispositivo se intercepta con información del plano. Luego lo ubica sobre el plano con su información de representación.	Paso 1: Selecciona un dispositivo del árbol de dispositivos. Paso 2: Clic izquierdo del mouse presionado. Se arrastra hacia el plano presionando el Clic izquierdo. Paso 3: Se deja de presionar el Clic izquierdo.
EC 1.2 Ubicar dispositivo incorrectamente	Luego de seleccionar y arrastrar el dispositivo, es ubicado dentro del editor pero fuera del área del plano.	Muestra un mensaje de error “El dispositivo ha sido ubicado incorrectamente”.	Paso 1: Clic izquierdo del mouse presionado. Paso 2: Se deja de presionar el Clic izquierdo del mouse dentro del

			editor pero fuera del plano.
EC 1.3 Ubicar dispositivo sobre un área que contiene información del plano.	Luego de seleccionar y arrastrar el dispositivo, es ubicado sobre un área que contiene posibles líneas u objetos del plano. Se muestra una ventana de notificación con una advertencia y se brinda la posibilidad al operador de ubicar o no el dispositivo en la posición inicial. En dependencia de la elección del operador el sistema ubica o no ubica el dispositivo.	Muestra un mensaje de advertencia "El dispositivo ha sido ubicado en un área del plano donde se encuentran posibles paredes u otros objetos" y brinda la posibilidad de ubicarlo o no en dicha posición.	Paso 1: Clic izquierdo del mouse presionado. Paso 2: Se deja de presionar el Clic izquierdo del mouse sobre un área que contiene posibles líneas u objetos del plano.
EC 1.4 Ubicar dispositivo es cancelado.	Luego de seleccionar y arrastrar el dispositivo, el mismo se ubica fuera del editor.	Cancela el proceso de ubicar el dispositivo sobre el plano.	Paso 1: Clic izquierdo del mouse presionado. Paso 2: Se deja de presionar el Clic izquierdo del mouse fuera del área del editor.

SC1: Mostrar información

Tabla 11 SC1: Mostrar información

Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 1.1 Mostrar información correctamente.	Desplegar el menú sobre el dispositivo ubicado y seleccionar la opción "Mostrar información". Muestra la información del dispositivo seleccionado en una ventana agrupando sus atributos y valores en una tabla. Luego el operador cierra la ventana.	Muestra una ventana con la información del dispositivo de acuerdo a los siguientes campos: Imagen, Nombre, Tipo de dispositivo, Posición en el eje x, Posición en el eje y, Ángulo de orientación y Zona a la que pertenece.	Paso 1: Clic derecho sobre el dispositivo. Paso 2: Clic izquierdo en "Mostrar información". Paso 3: Clic izquierdo para cerrar la ventana.

EC 1.2 Mostrar información actualizada.	Luego de seleccionar la opción "Mostrar información" se visualiza en una ventana la información del dispositivo seleccionado. Esta información se actualiza en dependencia del dispositivo que se seleccione o modifique.	Muestra una ventana con la información del dispositivo y la actualiza en dependencia del dispositivo que se seleccione o modifique.	Paso 1: Clic derecho sobre el dispositivo. Paso 2: Clic izquierdo en "Mostrar información". Paso 3: Clic izquierdo sobre otro dispositivo. Paso 4: Modificar el ángulo de orientación de una cámara fija.

SC2: Definir ángulo de orientación

Tabla 7 SC2: Definir ángulo de orientación

Escenario	Descripción	Respuesta del sistema	Flujo Central
EC 1.1 Definir ángulo de orientación correctamente.	Desplegar el menú sobre el dispositivo ubicado y seleccionar la opción "Definir ángulo de orientación". Redefinirle con el cursor del mouse el ángulo de orientación a la cámara fija y luego clic izquierdo dentro del plano.	Pinta una línea discontinua de color negro desde el dispositivo hasta el cursor del mouse. Luego de ser redefinido el ángulo por el operador pinta una línea discontinua de color azul desde el dispositivo hasta el primer punto de intercepción y de rojo del mismo hacia el borde del plano.	Paso 1: Clic derecho sobre el dispositivo. Paso 2: Clic izquierdo en "Definir ángulo de orientación". Paso 3: Redefinir ángulo con el cursor del mouse. Paso 4: Clic izquierdo.
EC 1.2 Definir ángulo de orientación incorrectamente	Luego de seleccionar la opción "Definir ángulo de orientación". Redefinirle con el mouse el ángulo de orientación a la cámara fija y luego clic izquierdo fuera del plano.	No redefine el ángulo de orientación de la cámara fija hasta que el operador no seleccione un ángulo dentro del plano.	Paso 1: Clic derecho sobre el dispositivo. Paso 2: Clic izquierdo en "Definir ángulo de orientación". Paso 3: Redefinir ángulo. Paso 4: Clic izquierdo fuera del plano.
EC 1.3 Definir ángulo de orientación es	Luego de seleccionar la opción "Definir ángulo de orientación" se presiona el	Cancela el proceso de redefinir ángulo de orientación para	Paso 1: Clic derecho sobre el dispositivo. Paso 2: Clic izquierdo

cancelado.	clic derecho del mouse y se cancela el proceso de redefinir el ángulo de orientación.	el dispositivo ubicado.	en “Definir ángulo de orientación”. Paso 3: Clic derecho.

4.6.4 Resumen de las pruebas realizadas

Al finalizar la primera iteración de pruebas al componente se encontraron 7 no conformidades, y se procedió a realizar una segunda iteración donde las mismas fueron corregidas arrojando resultados satisfactorios, debido a esto no se necesitó realizar una tercera iteración. En la Figura 18 se representa un gráfico de barras para establecer una comparativa entre los resultados obtenidos luego de la primera y segunda iteración de pruebas, teniendo en cuenta cada caso de uso.

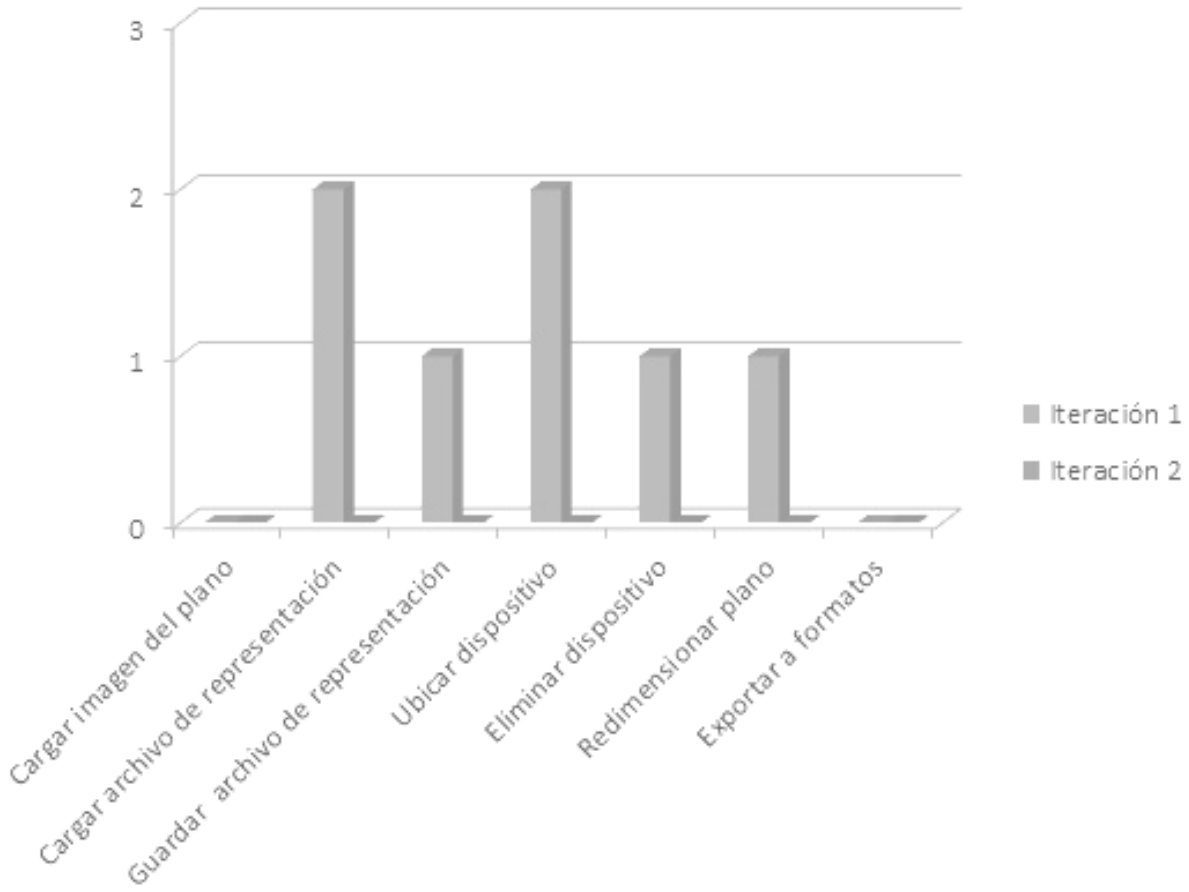


Figura 18 Resultados de las pruebas

4.7 Conclusiones

Al finalizar las actividades correspondientes a este capítulo se apreció que la realización del flujo de Diseño posibilitó modelar la estructura del componente siguiendo el estilo arquitectónico definido, así como, representar el flujo de eventos durante la interacción del operador con el sistema. Siguiendo el proceso de desarrollo donde aparece la implementación, el uso del IDE Qt Creator posibilitó desarrollar una interfaz gráfica sencilla e intuitiva partiendo del uso de widgets como componentes gráficos. A partir de la utilización de la biblioteca OpenCV se garantizaron los procesos de transformación y detección de líneas realizados a la imagen del plano arquitectónico a la hora de representar un dispositivo. Luego de ser generado el código fuente del componente se aplicaron casos de prueba mediante la técnica de caja negra lo que facilitó determinar las no conformidades al culminar la primera iteración de desarrollo. En la segunda iteración de pruebas se alcanzaron resultados satisfactorios debido a que no fue identificada ninguna no conformidad. A partir de la realización de pruebas se validó que los requisitos funcionales definidos para el componente sean totalmente operativos elevando la calidad del software.

CONCLUSIONES

El cumplimiento de las actividades propuestas durante la investigación hizo posible que los objetivos trazados se lograran con éxito. Lo que conllevó arribar a las siguientes conclusiones:

- El análisis de soluciones que utilizan mapas o planos posibilitó extraer características comunes para desarrollar el componente. Además se apreció que estos sistemas se encuentran bajo licencias propietarias y las funcionalidades que presentan no se corresponden con la necesidad de la investigación.
- La valoración de tecnologías, herramientas y metodología permitió realizar una adecuada selección de las mismas para regir y darle soporte al proceso de desarrollo del componente. Logrando que dicha selección se encuentre dentro de las tecnologías propuestas por el departamento de Señales Digitales para el desarrollo del sistema Suria.
- Los artefactos generados durante el proceso de desarrollo del componente servirán para que otros desarrolladores obtengan un mejor entendimiento de su estructura, facilitando la realización de posibles modificaciones o la adición de nuevas funcionalidades al mismo.
- La utilización de herramientas libres para desarrollar el componente contribuyó con la soberanía tecnológica que pretende alcanzar el país.
- El componente desarrollado presenta características multiplataforma lo que permitió ampliar las posibilidades de mercado.
- La incorporación del componente desarrollado al módulo Visor constituye un aumento en las funcionalidades del sistema Suria, debido a que los procesos de representación se lograrán realizar con facilidad. Mediante la representación resultante se le ofrece al operador una amplia visión de cómo pueden estar desplegados los dispositivos, en qué áreas pueden tener mayor utilidad y cómo lograr asegurar la mayor cantidad de áreas según la disponibilidad de estos.

RECOMENDACIONES

Durante el desarrollo del trabajo han surgido ideas que podrían implementarse en versiones posteriores del componente, de forma que se logren agregar o modificar las funcionalidades, para lo cual se recomienda:

- La utilización de un SIG[21] como representación gráfica para realizar los procesos de representación de dispositivos de seguridad.
- La incorporación de nuevos algoritmos de procesamiento de imágenes para elevar la calidad del proceso durante las transformaciones y detecciones realizadas al plano.

[21] Sistema de Información Geográfica

REFERENCIAS

AE. 2005. Asesoria Informática. [En línea] 2005. [Citado el: 15 de Octubre de 2012.] http://www.aseinformatica.com/camarasip.php.

Americati. 2006. [En línea] 2006. [Citado el: 12 de Noviembre de 2012.] http://www.americati.com/doc/ventajas_c.pdf.

Barzanallana, Rafael. 2006. Universidad de Murcia. [En línea] 30 de Diciembre de 2006. [Citado el: 2 de Mayo de 2013.] http://www.um.es/docencia/barzana/IAGP/Iagp2.html.

Bembibre, Cecilia. 2007. Definicion abc. [En línea] 2007. [Citado el: 12 de Octubre de 2012.] http://www.definicionabc.com/general/planos.php.

Booch, Grady. 2000. *El Lenguaje Unificado de Modelado.* Madrid : Series Editors, 2000.

BOSCH. 2012. Innovación para tu vida BOSCH. [En línea] 2012. [Citado el: 17 de Octubre de 2012.] http://www.boschsecurity.es/content/language1/html/6110_ESN_XHTML.asp.

Carrasco, Ignacio. 2005 . BORMART, S.A. [En línea] 2005 . [Citado el: 15 de Octubre de 2012.] http://www.borrmart.es/articulo_seguritecnia.php?id=2275.

CITELUM. 2012. CITELUM otra luz para el planeta. [En línea] 2012. [Citado el: 17 de Octubre de 2012.] http://www.citelum.mx/Tecnolog%C3%ADasdelainformaci%C3%B3n/Videovigilancia.aspx.

CORADIR. 2012. CORADIR S.A. [En línea] 2012. [Citado el: 17 de Octubre de 2012.] http://www.coradir.com.ar/producto/detalle/sistema-de-video-vigilancia-coradir.

DATYS. 2010. [En línea] 2010. [Citado el: 17 de Octubre de 2012.]

Definicion.de. 2008. [En línea] 2008. [Citado el: 15 de Octubre de 2012.] http://definicion.de/mapa/.

Dictionary. 2012. The Free Dictionary. [En línea] 2012. [Citado el: 15 de Octubre de 2012.] http://es.thefreedictionary.com/plano.

Digia. 2012. Qt Digia. [En línea] 2012. [Citado el: 14 de Noviembre de 2012.] http://qt.digia.com/Product/Developer-Tools/.

Jacobson, Ivar. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : Series Editors, 2000.

Mateos, Ginés García. 2010. Universidad de Murcia. [En línea] 2010. [Citado el: 9 de Diciembre de 2012.] http://dis.um.es/~ginesgm/files/doc/pav/guion2.pdf.

Pavón, Juan. 2004. Facultad de Informática.Universidad Complutense de Madrid. [En línea] 2004. [Citado el: 3 de Mayo de 2013.] http://www.fdi.ucm.es/profesor/jpavon/poo/2.14PDOO.pdf.

PCS. 1998. La Página del Conocimiento y del Saber. [En línea] 1998. [Citado el: 13 de Octubre de 2012.] http://www.saber.golwen.com.ar/hmapa.htm.

Pes, Carlos. 2006. carlospes.com. [En línea] 2006. [Citado el: 9 de Diciembre de 2012.] http://www.carlospes.com/minidiccionario/entorno_integrado_de_desarrollo.php.

Pressman, Roger S. 2002. *Ingeniría de software. Un enfoque práctico. Quinta edición.* Madrid : s.n., 2002.

Quesada, Alma Evangelina. 2008. Webs. [En línea] 2008. [Citado el: 7 de Diciembre de 2012.] http://www.freewebs.com/almaquesada/portafolio.htm.

RAE. 2010. Real Academia Española. *Diccionario de la Lengua Española. Vigésima segunda edición.* [En línea] 2010. [Citado el: 15 de Octubre de 2012.] http://www.rae.es/rae.html..

Reynoso, Carlos. 2004. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* Buenos Aires : s.n., 2004.

Semanat, Edmis D. 2009. *Trabajo de Diploma Sistema de Video Vigilancia.* La Habana : s.n., 2009.

Soft. 2010. Soft112. [En línea] 2010. [Citado el: 12 de Diciembre de 2012.] http://visual-paradigm-for-uml-standard.soft112.com/.

Sommerville, Ian. 2005. *Ingeniería del software.Séptima edición.* Madrid : PEARSON ADDISON WESLEY, 2005.

ZonaQt. 2010. ZonaQt. [En línea] 2010. [Citado el: 8 de Diciembre de 2012.] http://www.zonaqt.com/tutoriales/tutorial-b%C3%A1sico-de-qt-4.

BIBLIOGRAFÍA

AE. 2005. Asesoria Informática. [En línea] 2005. [Citado el: 15 de Octubre de 2012.] http://www.aseinformatica.com/camarasip.php.

Americati. 2006. [En línea] 2006. [Citado el: 12 de Noviembre de 2012.] http://www.americati.com/doc/ventajas_c.pdf.

Andrea Catherine Alarcón Aldana, Erika Maria Sandoval Valero. Herramientas CASE para Ingeniería de Requisitos.

Barzanallana, Rafael. 2006. Universidad de Murcia. [En línea] 30 de Diciembre de 2006. [Citado el: 2 de Mayo de 2013.] http://www.um.es/docencia/barzana/IAGP/Iagp2.html.

Bembibre, Cecilia. 2007. Definicion abc. [En línea] 2007. [Citado el: 12 de Octubre de 2012.] http://www.definicionabc.com/general/planos.php.

Booch, Grady. 2000. *El Lenguaje Unificado de Modelado.* Madrid : Series Editors, 2000.

BOSCH. 2012. Innovación para tu vida BOSCH. [En línea] 2012. [Citado el: 17 de Octubre de 2012.] http://www.boschsecurity.es/content/language1/html/6110_ESN_XHTML.asp.

Bradski, Gary y Kaehler, Adrian. 2008. *Learning OpenCV.* Sebastopol : O'Reilly, 2008.

Carrasco, Ignacio. 2005 . BORMART, S.A. [En línea] 2005 . [Citado el: 15 de Octubre de 2012.] http://www.borrmart.es/articulo_seguritecnia.php?id=2275.

CITELUM. 2012. CITELUM otra luz para el planeta. [En línea] 2012. [Citado el: 17 de Octubre de 2012.] http://www.citelum.mx/Tecnolog%C3%ADasdelainformaci%C3%B3n/Videovigilancia.aspx.

CORADIR. 2012. CORADIR S.A. [En línea] 2012. [Citado el: 17 de Octubre de 2012.] http://www.coradir.com.ar/producto/detalle/sistema-de-video-vigilancia-coradir.

DATYS. 2010. [En línea] 2010. [Citado el: 17 de Octubre de 2012.]

Definicion.de. 2008. [En línea] 2008. [Citado el: 15 de Octubre de 2012.] http://definicion.de/mapa/.

Dictionary. 2012. The Free Dictionary. [En línea] 2012. [Citado el: 15 de Octubre de 2012.] http://es.thefreedictionary.com/plano.

Digia. 2012. Qt Digia. [En línea] 2012. [Citado el: 14 de Noviembre de 2012.] http://qt.digia.com/Product/Developer-Tools/.

Jacobson, Ivar. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : Series Editors, 2000.

Laganière, Robert. 2011. *OpenCV 2 Computer Vision Application Programming.* Birmingham : Packt Publishing, 2011.

Martinto, Msc. Pedro Carlos Pérez. Diseño Teórico de la Investigación Científica.

Martinto, Msc. Pedro Carlos Pérez. El Proceso de Investigación Científica. Estructura. Referencias bibliográficas y Normas.

Mateos, Ginés García. 2010. Universidad de Murcia. [En línea] 2010. [Citado el: 9 de Diciembre de 2012.] http://dis.um.es/~ginesgm/files/doc/pav/guion2.pdf.

Pavón, Juan. 2004. Facultad de Informática.Universidad Complutense de Madrid. [En línea] 2004. [Citado el: 3 de Mayo de 2013.] http://www.fdi.ucm.es/profesor/jpavon/poo/2.14PDOO.pdf.

PCS. 1998. La Página del Conocimiento y del Saber. [En línea] 1998. [Citado el: 13 de Octubre de 2012.] http://www.saber.golwen.com.ar/hmapa.htm.

Pes, Carlos. 2006. carlospes.com. [En línea] 2006. [Citado el: 9 de Diciembre de 2012.] http://www.carlospes.com/minidiccionario/entorno_integrado_de_desarrollo.php.

Pressman, Roger S. 2002. *Ingeniría de software. Un enfoque práctico. Quinta edición.* Madrid : s.n., 2002.

Quesada, Alma Evangelina. 2008. Webs. [En línea] 2008. [Citado el: 7 de Diciembre de 2012.] http://www.freewebs.com/almaquesada/portafolio.htm.

RAE. 2010. Real Academia Española. *Diccionario de la Lengua Española. Vigésima segunda edición.* [En línea] 2010. [Citado el: 15 de Octubre de 2012.] http://www.rae.es/rae.html..

Reynoso, Carlos. 2004. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* Buenos Aires : s.n., 2004.

Semanat, Edmis D. 2009. *Trabajo de Diploma Sistema de Video Vigilancia.* La Habana : s.n., 2009.

Soft. 2010. Soft112. [En línea] 2010. [Citado el: 12 de Diciembre de 2012.] http://visual-paradigm-for-uml-standard.soft112.com/.

Sommerville, Ian. 2005. *Ingeniería del software.Séptima edición.* Madrid : PEARSON ADDISON WESLEY, 2005.

UCI, Colectivo de Profesores de ISW. 2010. El Diseño Metodológico de la Investigación Científica.

ZonaQt. 2010. ZonaQt. [En línea] 2010. [Citado el: 8 de Diciembre de 2012.] http://www.zonaqt.com/tutoriales/tutorial-b%C3%A1sico-de-qt-4.

ANEXOS

Anexo 1 Diagrama de Clases del Diseño

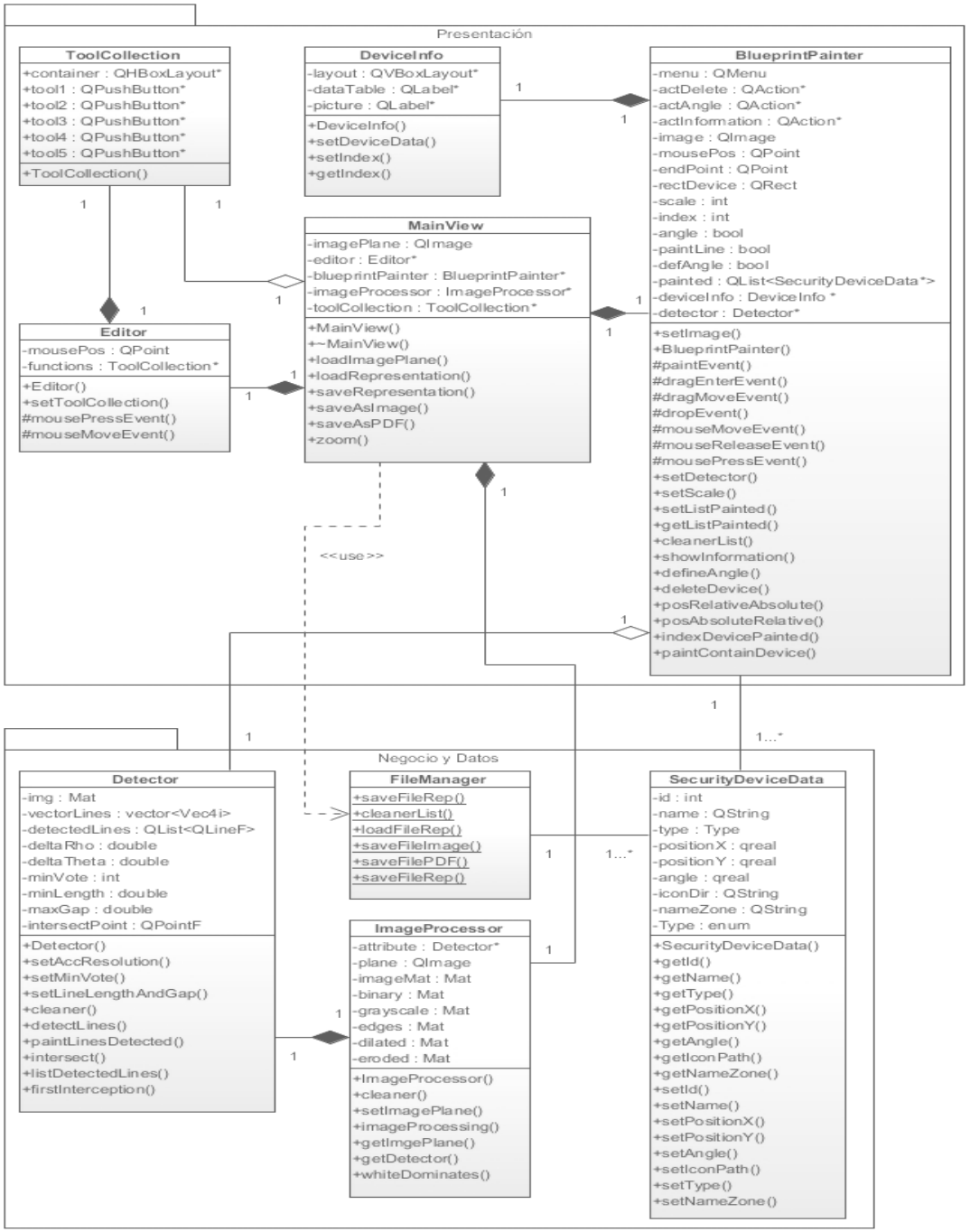


Figura 19 Diagrama de clases del diseño

GLOSARIO

Componente: agrupa elementos o entra en la composición de un todo.

Dispositivo: mecanismo o artificio dispuesto para producir una acción prevista.

Mapa: representación geográfica de una parte de la superficie terrestre.

Plano: representación esquemática, en dos dimensiones y a determinada escala, de un terreno.

Seguridad: cualidad de lo seguro que pueden estar determinadas cosas.

Video: grabación de secuencias de imágenes acompañadas o no de sonidos.

Vigilancia: servicio ordenado y dispuesto para vigilar.