Universidad de las Ciencias Informáticas

Facultad 6

Título: Módulo de autonomía para el control del espacio de almacenamiento en los servidores de medias del sistema Suria.

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas.

Autores: Reina Salmón Cajigal.
Danny Almeida Pérez.

Tutores: Ing. Merlín Milián Díaz.
Ing. Nilo Tomás Díaz Alés.

Co-Tutora: Msc. Yaneisis Pérez Heredia.

Ciudad de La Habana, 2013
"Año 55 de la Revolución".

"*El secreto del éxito es dedicarse a un fin*".

José Martí.

DECLARACION DE AUTORIA

Declaramos que somos los únicos autores del presente trabajo y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmamos la presente a los ___ días del mes de _____ del año 2013.

Autores:

_____ _____

Reina Salmón Cajigal. Danny Almeida Pérez.

Tutores:

_____ _____

Ing. Merlín Milián Díaz. Ing. Nilo Tomás Díaz Alés.

DATOS DE CONTACTO

Tutora: Ing. Merlín Milián Díaz (mmilian@uci.cu).

Graduada de Ingeniero en Ciencias Informáticas en la UCI en el año 2011. Pertenece en la Facultad 6, al departamento Señales Digitales del Centro de Geoinformática y Señales Digitales (GEYSED). Se desempeña como planificadora en el proyecto Video Vigilancia.

Tutor: Ing. Nilo Tomás Díaz Alés (ntdiaz@uci.cu).

Graduado de Ingeniero en Ciencias Informáticas en la UCI en el año 2012. Pertenece en la Facultad 6, al departamento Señales Digitales del Centro de Geoinformática y Señales Digitales (GEYSED). Se desempeña como programador en el proyecto Video Vigilancia.

Co-Tutora: Msc. Yaneisis Pérez Heredia (yheredia@uci.cu).

Graduada de Ingeniero Informático en la Universidad de Holguín en el año 2004. Pertenece al Departamento Docente Metodológico Central de Ingeniería y Gestión de Software. Se desempeña como asesora metodológica principal de la asignatura Gestión de Software. Máster en informática aplicada desde el año 2006.

AGRADECIMIENTOS

De Reina:

A los dos seres más grandes para mí en este mundo, mis padres Reinold y Clara Reina, por haber sido mis guías, consejeros y llevarme siempre por el mejor camino.
A mi abuelo Cajigal que siempre me ha brindado su nobleza y me ha hablado con la verdad.
A mis hermanos carnales por brindarme siempre su cariño.
A toda mi familia, en especial a mis primas Any y Aliuska, a mis tíos Rafaela, Orlando, Evaristico, Yamirlenis, Pepín y Vilmar por apoyar cada una de mis decisiones.
A mí querido tesoro Loismarx, que ha sabido ser novio, hermano, amigo, compañero y consejero en los 20 meses de relación. Gracias a la vida por darme el privilegio de tenerte a mi lado.
A Bárbara, Juan de Dios, Luis Augusto, María Emilia y mis padres postizos Ismary y Pedro por darme tanto aliento.
A mis amigas y hermanas Yanet, Dairilis y Aindamais, por aceptarme como soy.
A mi compañero de tesis Danny, a quien recordaré por toda la vida, gracias por tus consejos.
A mis tutores Merlín y Nilo, por afrontar las mejores ideas para nuestro trabajo de diploma.
A Yaneisis y Alain por su dedicación siempre que los necesité.
A todos en el proyecto, en especial a Olga, por haberme orientado desde tercer año. Gracias por tu apoyo.
A los miembros del tribunal por asumir la tarea de evaluarme y saber transmitirme de manera constructiva todas sus críticas.
A todas las personas que me han ayudado en la vida, especialmente a Joel, Liset, Ramón, Mileidis, Batista, Grisel, Diana y familia, Tania, Vladimir, David y familia, Aymee, Piloto, Sucel, Pedro, Elennis, Lily, Azulito, Baby, Darlon, Radel, Jeandy, Chiqui, Nely, Luis Ángel, Yaima, Nancy, Miguelito, Yenney, Rafael, Lisbet, Mary, Chachi y a mis vecinos, por preocuparse por mí.
A todos mis compañeros de apartamentos, aulas y profesores, por formar parte de mi familia en estos años de estudios.

AGRADECIMIENTOS

De Danny:

Esta es una de las partes más difíciles de la tesis, aquí es donde le agradezco a todas las personas que de una forma u otra que me han ayudado con la realización de este hermoso sueño.

Primero que todo a dos personas que me han permitido hoy ser quien soy, mis padres Marisol y Francisco, que han estado ahí para mí y me han apoyado en todo momento. Gracias por darme ánimo siempre, por aconsejarme, guiarme en la vida, por no permitir que nunca me rindiera, ustedes son la razón por la cual me despierto todos los días buscando la manera de ser una mejor persona para que siempre se sientan orgullosos de mi.

A mi hermano Yosvel, por su gran hermandad eterna.

A mi familia en general por su gran apoyo, muchas gracias por confiar en mí.

Tengo que agradecerle mucho a una persona muy especial en mi vida , a mi cosa Yanet, que fue como una segunda madre para mi, de veras muchas gracias por quererme, apoyarme en todo momento y estar a mi lado cuando más lo necesitaba.

A todos mis amigos que me es imposible mencionarlos a todos, se convirtieron en este tiempo en mi familia, gracias por compartir tanto los momentos buenos como los malos durante estos 5 años y mi gente de la Redbull tengan presente que nunca los olvidaré.

A mi compañera de tesis Reina por tener una paciencia de oro conmigo y saber comprenderme.

A mis tutores Merlín y Nilo que supieron guiarme hacia mi meta, gracias por haberme ayudado y por su grandísimo esfuerzo.

A todos los estudiantes y profesores del proyecto, en especial a Reinier Pupo, gracias por tu ayuda incondicional.

A los miembros del tribunal por su exigencia, dedicación y preocupación durante el periodo de desarrollo del trabajo de diploma.

A la Revolución y a la Universidad por presentarme la oportunidad de formarme como un profesional.

DEDICATORIA

De Reina:

A mis padres por haberme dado tanto amor y apoyo en todos los momentos de mi vida.
A la vida por haberme dado tanta salud y fuerza para lograr esta meta y llegar a ser la persona que soy hoy.

De Danny:

A toda mi familia, en especial a mis padres a quienes les debo todo lo que he logrado en esta vida.
A mi abuelo Pancho que siempre fue un gran ejemplo para mí.

RECONIMIENTO

El presente trabajo se presentó en la 11na Edición de Jornada Científica Estudiantil 2013 de la facultad # 6, obteniendo la máxima calificación de Relevante.

RESUMEN

El proyecto Video Vigilancia del Centro de Geoinformática y Señales Digitales (GEYSED), perteneciente a la Facultad 6 de la Universidad de las Ciencias Informáticas (UCI), actualmente cuenta con el producto Suria. El mismo está compuesto por seis módulos, dentro de ellos está el de autonomía que tiene como objetivo gestionar el espacio de los dispositivos de almacenamiento en los servidores de medias. Las medias son generadas a través de la captura de cámaras IP[1] y guardadas en el servidor destinado, ocupando gran cantidad de espacio debido al tamaño que contienen las imágenes y sonidos que la conforman. Llega el momento que se hace insuficiente la capacidad de las unidades de recepción de las medias, provocando varios inconvenientes como: la saturación del disco duro, el entorpecimiento en la ejecución de las reglas programadas y con ello el colapso del sistema. Este módulo no puede ser integrado al sistema Suria en la nueva versión en Qt, porque solo funciona bajo el sistema operativo *Windows*.

La presente investigación tiene como objetivo desarrollar un Módulo de autonomía para el control del espacio de almacenamiento en los servidores de medias del sistema Suria basado en *plugins*. El mismo funciona en los sistemas operativos Linux y *Windows* permitiendo mejorar el control de los recursos de almacenamiento en dicho sistema. La solución propuesta está acorde con el proyecto de lineamientos de la política económica y social del VI Congreso del PCC.

Palabras clave: autónomo, IP, medias, módulo, *plugins,* Suria.

[1] **IP**: Protocolo de Internet.

ABSTRACT

In Video Surveillance Project of the Geoinformatics and Digital Signal Development Center (GEYSED) belonging to the Faculty 6 at the University of Informatics Sciences (UCI), currently has a product called "Suria", which is integrated of six modules. One of these modules, due to the large volumes of data generated during averages catches through IP cameras, is in charge of the autonomous control of the storage space on media servers avoiding hard disk saturation, the obstruction in the execution of programmed rules and thus, the collapse of the system. This module currently cannot be integrated into Suria system obstructing its scalability.

The results obtained from the research are collected throughout this paper. It describes the current workflow of the process of storage space control on media servers in Suria system. After a detailed study of them, was identified the problematic situation by determining the need to develop a module based on plugins, to ensure control of saturation storage space on media servers of Suria system. The proposed solution is consistent with the draft guidelines of the economic and social policy of the Sixth Congress of the PCC.

Keywords: autonomous, IP, media, module, plugins, Suria.

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

El hombre desde la antigüedad, en su labor científico técnica se ha percatado no solo de la necesidad de tratar y utilizar la información, sino de la manera de ingeniar una alternativa que le propicie almacenar de forma segura todos los datos que ha ido generando a través del tiempo. Según Castañeda: "para un gran número de los países del mundo, es incuestionable la importancia que tiene la conservación de los documentos audiovisuales como películas fílmicas o las grabaciones de audio y vídeo; y no únicamente para almacenarlas debidamente en instalaciones adecuadas que garanticen su supervivencia, sino para disponer de ellas en todo instante" (Castañeda, 1992). Debido a la necesidad mencionada anteriormente, se introduce oportunamente la aparición de las primeras vías de almacenamiento de la información.

Alrededor de los años 60 comenzó el surgimiento de los primeros dispositivos de almacenamiento. Las computadoras a partir de este momento comienzan a tener la facilidad de almacenar información en dispositivos de almacenamiento masivo, entre ellos se encuentran los dispositivos de almacenamiento interno y dispositivo de almacenamiento externo. Las primeras generaciones de los dispositivos de almacenamiento aparecen de la siguiente manera: en la primera generación aparece el tambor magnético interno, en la segunda generación surgen las cintas magnéticas, los discos magnéticos extraíbles y los tambores magnéticos giratorios, en la tercera generación salen a la luz el disco magnético y el *floppy* de 5 ¼, en la cuarta generación emergen los discos duros y el disco *floppy* de 3 ½, y en la quinta generación aparece el CD-ROM[2], los DVD[3], así como las memorias portátiles por puerto USB[4] (Vega, 1986).

En la actualidad, debido a la cantidad de información que es manejada por los usuarios, los dispositivos de almacenamiento se han convertido en elementos tan vitales dentro del ordenador como los propios microprocesadores y las memorias RAM[5]. Un objetivo primordial a lo largo de la historia siempre ha sido buscar el camino para crear sistemas cada vez más pequeños físicamente, con mayor capacidad para almacenar los datos y a su vez tratarlos con gran rapidez. Los servidores ocupan un papel importante e

[2] **CD-ROM:** (*Compact Disc - Read Only Memory*) Disco Compacto de Memoria de Sólo Lectura.

[3] **DVD:** (*Digital Versatile Disc*) Disco Digital Versátil es un dispositivo de almacenamiento óptico.

[4] **USB:** (*Universal Serial Bus*) Bus Universal en Serie es el puerto que permite conectar periféricos a una computadora.

[5] **RAM:** (*Random Access Memory)* Memoria de Acceso Aleatorio es una memoria de semiconductortores en la que se puede escribir o leer información.

imprescindible en el avance de las diferentes tecnologías, por su gran capacidad de almacenamiento de los datos para ser reutilizados en el transcurso de los años. Estos son utilizados en diversas ramas de la sociedad, su utilidad es primordial en los sistemas de video vigilancia.

La video vigilancia, surge cuando el ser humano se vio en la necesidad de crear sistemas que le permitieran mantener el control de todo lo que lo rodea. Además es el resultado de toda aquella actividad que supone la colocación de una cámara de grabación fija o móvil, con la finalidad de garantizar la seguridad de una instalación o de las personas, supervisando el correcto desempeño de las tareas en el entorno laboral. Los servidores resultan de gran utilidad para guardar toda la información captada por cámaras IP de video y una vez almacenada esa información, realizar la supervisión local y remota de imágenes de video. El gran volumen de información que es generada por los sistemas de circuitos cerrados o sistemas de video vigilancia, como comúnmente se les conoce; pudiera ser requerida y/o utilizada para análisis posteriores, por lo que debe evitarse su pérdida parcial o total.

El proyecto Video Vigilancia del Centro de Desarrollo GEYSED, perteneciente a la Facultad 6 de la Universidad de las Ciencias Informáticas, cuenta actualmente con un producto llamado “Suria”, el cual está integrado por seis módulos. Uno de estos módulos, debido a los grandes volúmenes de datos que son generados durante las capturas de las medias a través de cámaras IP, se encarga del control autónomo de espacio de almacenamiento en los servidores de medias evitando la saturación del disco duro, el entorpecimiento en la ejecución de las reglas programadas y con ello el colapso del sistema. Este módulo no puede integrarse al sistema Suria actual, producto a que solo puede ser utilizado sobre el sistema operativo *Windows*. La implementación del módulo, a pesar de ser realizada haciendo uso del *framework* Qt; se utilizó además la API[6] de Windows, impidiendo que pueda migrarse a software libre ocasionando que no se le puedan agregar nuevas funcionalidades que satisfagan las peticiones realizadas por los clientes, obstaculizando la escalabilidad del sistema.

Lo anteriormente explicado permite formular como **problema a resolver**: ¿Cómo garantizar el control de la saturación de los dispositivos de almacenamiento en los servidores de medias del sistema Suria? Para darle solución al problema planteado se define como **objeto de estudio**: Los procesos autónomos para el control del espacio de almacenamiento de los servidores de medias, teniéndose como **campo de acción**:

[6] **API**: *(*Application Programming Interface*)* Interfaz de programación de aplicaciones.

Los procesos autónomos para el control del espacio de almacenamiento de los servidores de medias en el sistema Suria. A partir del problema enunciado anteriormente se puede expresar la siguiente **idea a defender:** Con el desarrollo del módulo de autonomía para el control de espacio de almacenamiento en los servidores de medias del sistema Suria, se logrará un mejor manejo de la saturación del espacio de los dispositivos de almacenamiento de los servidores de medias.

Para dar solución a la situación problemática descrita y al problema planteado, la presente tesis se estructura y desarrolla en función del siguiente **objetivo general: "**Desarrollar un módulo para el control autónomo del espacio de almacenamiento en los servidores de medias del sistema Suria basado en *plugins*".

Para alcanzar el objetivo propuesto, se formulan las siguientes **tareas de investigación**:

1. Establecimiento de los fundamentos teóricos correspondientes al tema de la investigación.
2. Identificación de los requisitos del módulo de autonomía.
3. Realización del diseño del módulo de autonomía.
4. Implementación del diseño.
5. Validación del cumplimiento de los requisitos planteados.

Para el desarrollo de la investigación, se propone el uso de los siguientes **métodos científicos**:

Métodos Teóricos.

Los métodos científicos de investigación servirán de ayuda a la hora de dar cumplimiento al grupo de tareas planteadas, dentro de los métodos teóricos, se emplea el **análisis histórico – lógico** para realizar una amplia investigación sobre los procesos autónomos de gestión para el control del almacenamiento en servidores de media; sus antecedentes, evolución y principales características. También el **analítico - sintético** en el momento de estudiar bibliografías de diferentes autores y sintetizar los elementos más importantes de los procesos autónomos para almacenar en servidores de media y sobre las tecnologías existentes en el mundo. Además, se usó la **modelación** para la comprensión de los objetos y sus relaciones, para determinar la estructura jerárquica y dinámica del módulo.

Métodos Empíricos.

De los métodos empíricos se utilizó la **observación** para realizar valoraciones y obtener información sobre el funcionamiento del sistema en uso actual y sobre los similares, para poder obtener las características, beneficios y dificultades que presentan. Además, se empleó para tener una mejor visión a la hora de implementar el módulo a desarrollar.

El presente documento consta de tres capítulos:

Capítulo 1: En este capítulo se fundamentan los fundamentos teóricos de importancia para la investigación. Se describe la situación problemática actual y se realiza un estudio sobre las soluciones existentes para la administración de recursos de almacenamiento con características similares. Además, se especifica la metodología, herramientas y tecnologías que se utilizarán en la construcción de la idea planteada.

Capítulo 2: En este capítulo se describen las características fundamentales del módulo, mostrando los procesos del negocio a través del modelo de dominio. Se identifican los requisitos funcionales y no funcionales, facilitando la presentación del diagrama de casos de uso del módulo con las especificaciones de los casos de uso correspondientes.

Capítulo 3: En este capítulo se define la estructura de organización del módulo. Se obtienen los diagramas correspondientes al modelo de diseño y el modelo de implementación. También, se describe las pruebas que se le realizaron a la solución implementada.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción.

En este capítulo se hace referencia a un conjunto de conceptos asociados al dominio del problema actual. Se describen las características fundamentales del objeto de estudio y se brinda una panorámica del estado en que se encuentran las aplicaciones para la administración de recursos de almacenamiento. Además, se describe la metodología, herramientas y tecnologías necesarias para la construcción de la solución.

1.1 Conceptos asociados al dominio del problema.

En el presente trabajo el objetivo fundamental es el desarrollo de un **módulo,** que es un software capaz de agrupar a un conjunto de subprogramas y estructuras de datos. Los módulos son unidades que pueden ser compiladas por separado, haciéndolos esta característica reusables y les permite además que múltiples programadores trabajen en diferentes módulos de forma simultánea, produciendo ahorro en los tiempos de desarrollo (ALEGSA, 2009).

Los módulos son creados por varios **procesos** y estos a su vez consisten un conjunto de diferentes fases o etapas sucesivas que tiene una acción o un fenómeno complejo. (ESPAÑOLA, 2011). También se conocen como la acción de avanzar o ir para adelante, al paso del tiempo. La solución obtenida es capaz de decir qué acción realizar ante determinada situación; es decir, que es **autónoma**, entiéndase por autónoma que goza de autonomía, que es independiente, que trabaja sin depender de un empleador (The Free, 2012).

La cantidad de información que es manipulada de forma autónoma por el sistema Suria debe ser gestionada para evitar su pérdida, lo cual no es más que la acción de hacer diligencias que van a conducir al logro de un negocio o de un deseo cualquiera. La **gestión** es el proceso de planear, organizar, dirigir, evaluar y controlar (Restrepo, 2011) el espacio de almacenamiento con el propósito de lograr resultados satisfactorios, rentables y productivos.

En la presente investigación la gestión se le realizan a los espacios de almacenamiento ocupados por las **medias**, estas son cualquier tipo de objeto que usa de forma simultánea varios tipos de contenidos, como son: video, audio, texto, imágenes, animación, entre otros. Aunque el término oficial es multimedia, en numerosos medios de comunicación y en diversas informaciones oficiales se le llama

también: “media” (Baby, 2009). Término que se emplea en diferentes momentos del documento para referirse a las grabaciones que son capturadas por las cámaras IP que contienen secuencias de audio y video.

Como consecuencia del vertiginoso desarrollo de las TIC[7] hoy en día, las medias también han avanzado en términos de calidad, lo que provoca cada día un aumento de su tamaño y como consecuencia se requiere de un mayor espacio para el almacenamiento. Los encargados de administrar este espacio son los **servidores**, se denomina servidor a cualquier recurso de cómputo dedicado a responder a los requisitos de los usuarios. Los servidores se encargan de gestionar el tráfico en la red, para proveer de múltiples servicios a los usuarios tales como: la impresión, el acceso a bases de datos, fax y el procesamiento de imágenes (Zelaya, 2010).

Una vez que se tienen almacenadas una gran cantidad de medias en los servidores, se ocupa en su totalidad el espacio designado y hay que buscar la alternativa de liberar espacio del servidor sin perder la información que ya se encuentra almacenada. Se le denomina **transferencia** al proceso que se centra en pasar o llevar algo desde un lugar a otro (ESPAÑOLA, 2011). Se define transferencia al proceso de trasladar una media desde un dispositivo de almacenamiento origen hasta uno destino.

La transferencia se puede realizar utilizando diferentes variantes, cada una de ellas con sus particularidades especiales que la convierten con un mayor o menor grado de efectividad ante el problema existente. Una de las vías de mayor aceptación es a través de la implementación de ***plugins*****,** por la gran versatilidad que desarrollan los mismos. Un *plugin* es la unidad mínima que inserta una funcionalidad específica en una solución existente (Fontanillo, 2005), es decir, es un complemento que se integra a una aplicación para aportar una función nueva o específica.

1.2 Situación problemática.

En el proyecto Video Vigilancia del Centro de Desarrollo GEYSED, perteneciente a la Facultad 6 de la Universidad de las Ciencias Informáticas, cuenta actualmente con un producto llamado “Suria”, el cual está integrado por seis módulos. El módulo que se encarga del control del espacio de almacenamiento en los servidores de medias, presenta algunas no conformidades en cuanto a su uso. El gran volumen de datos que son generados durante la captura de las medias a través de cámaras IP, se guardan en

[7] **TICs**: Tecnologías de la Información y las Comunicaciones.

los servidores sin conocerse el momento en que se agota el espacio de almacenamiento y desconociendo el tratamiento adecuado que debe dársele a las medias cuando esto sucede. Los usuarios tienen que realizar las acciones de mover, copiar, eliminar y supervisar el espacio del servidor de media en el sistema de forma manual, resultando compleja la presencia de un personal capacitado a tiempo completo para que asegure el correcto funcionamiento de la aplicación informática. En este instante comienza a consumirse innecesariamente tiempo, materiales y recursos humanos. Lo anteriormente descrito trae consigo que se produzca la saturación del disco duro, el colapso del sistema o el entorpecimiento en la ejecución de disímiles tareas y/o reglas por falta de espacio de almacenamiento.

Con el propósito de resolver esta deficiencia se desarrolló el "Sistema para el control autónomo de espacio en disco en servidores de media". El mismo no puede integrarse al sistema Suria en estos momentos, producto a que solo puede ser utilizado sobre el sistema operativo *Windows*. La implementación del módulo, a pesar de ser realizada haciendo uso del *framework* Qt; se utilizó además la API de Windows, impidiendo que pueda migrarse a software libre, además no se le pueden agregar nuevas funcionalidades que satisfagan las peticiones realizadas por los clientes y con ello obstaculizando la escalabilidad del sistema. Por todos los antecedentes expuestos se hace imprescindible idear una nueva solución que responda de forma autónoma a la necesidad de controlar el espacio de almacenamiento en los servidores de medias.

1.3 Procesos autónomos para gestionar el espacio de almacenamiento en los servidores de medias.

Los procesos autónomos son actividades independientes y sucesivas que se programan ante situaciones inadecuadas para tomar decisiones inteligentes sobre políticas futuras. La administración de los recursos de almacenamiento en servidores de medias se realizan a través de las siguientes actividades: aplicar las políticas de administración de espacio, monitorear el entorno de almacenamiento, informar los procesos de alertas, ejecutar las acciones programadas y realizar reporte.

- ✓ **Aplicar las políticas de administración de espacio:** Permite realizar la implementación de diferentes acciones que se ejecutan ante determinadas situaciones. Controlando la cantidad de espacio asignado al servidor para el guardado de las medias, se establecen límites en el

mismo para poder programar en qué momento es que se pueden ejecutar las gestiones de copiar, mover, eliminar o supervisar el porcentaje de las medias y advertir a los usuarios que están llegando a los límites designados (Guerrero, 2003).

- ✓ **Monitorear el entorno de almacenamiento**: Permite monitorear de manera continua el entorno de almacenamiento relacionado con la conectividad y las condiciones del espacio. Informa de cualquier problema relacionado con la disponibilidad o el rendimiento según las políticas predefinidas.

- ✓ **Informar los procesos de alertas:** Permite realizar los avisos de forma sistemática ante cualquier problema relacionado con la disponibilidad o el rendimiento, según las políticas predefinidas con anterioridad (Kuppler, 2005).

- ✓ **Ejecutar las acciones programadas:** Permite tomar decisiones dependiendo de las acciones programadas, luego de haberse activado una alerta producida por cualquier situación inadecuada o la llegada a los límites designados en las políticas. Además, garantiza que se cumplan los niveles de servicio y se evita la pérdida de datos.

- ✓ **Realizar reportes:** Permite conocer la disposición de la máxima capacidad destinada al guardado de las medias, cuánto espacio de almacenamiento se dispone y cuánto se necesita en el futuro. Además permite auditar el funcionamiento del sistema, adquirir una mejor planificación de la capacidad y reducir los gastos (NORTERN, 2009).

1.4 Soluciones existentes

En la actualidad el desarrollo de las tecnologías computacionales, ha desencadenado una fuerte y acelerada competencia en el mercado mundial. Los clientes son más exigentes y existe un mayor número de empresas que se dedican al desarrollo de soluciones informáticas. Las soluciones creadas para el control de espacio de almacenamiento se realizan a través del funcionamiento autónomo, debido al crecimiento exponencial del flujo que se genera a la hora de ser almacenada la información. Existen diversos sistemas desarrollados en Cuba y a nivel mundial que se encargan de gestionar los recursos de almacenamiento. A continuación se describen algunas de estas soluciones:

1.4.1 Soluciones existentes en el mundo.

Solución NORTHERN Storage Suite.

NORTHERN Storage Suite es una solución de software de administración de recursos de almacenamiento. Su objetivo fundamental es controlar el comportamiento de espacio en disco. Contiene una interfaz de usuario amigable, lo que permite que la comunicación con las personas que interactúan con la solución sea exitosa.

Figura 1. Solución *NORTHERN Storage Suite*.

Ventajas de NORTHERN: (NORTERN, 2009)

- ✓ Control de uso en tiempo real.
- ✓ Interacción con los usuarios.
- ✓ Informes exhaustivos sobre el almacenamiento.
- ✓ Automatización y optimización de tareas.
- ✓ Identificar patrones indeseados de uso y tomar acciones inmediatas.

- ✓ Ajustar las políticas de administración de espacio para llegar a los objetivos estratégicos.
- ✓ Realizar reportes programados y reportes de costos del storage por usuarios, grupos y departamentos, entre otros.

Solución Veritas *CommandCentral Storage*.

Veritas *CommandCentral Storage* es la solución de *Symantec* (corporación internacional que produce software, especialmente relacionado a la seguridad). Esta solución ofrece un control altamente centralizado en entornos de almacenamiento heterogéneos, donde aumenta la responsabilidad del departamento Tecnologías de la Información (TI) y su capacidad para satisfacer las necesidades de guardar un conjunto de aplicaciones para los diversos sectores a los que asiste.

Figura 2. Solución Veritas *CommandCentral Storage*.

Ventajas de Veritas *CommandCentral Storage* (*Symantec Corporation*, 1995-2012)**:**

- ✓ Permitir conocer cuánto espacio de almacenamiento se dispone y cuánto se necesita en el futuro, facilita la recuperación del material almacenado perdido o mal utilizado e implementa prácticas de compra según los datos históricos y actuales.

- ✓ Visualizar la ruta de datos completa desde la aplicación hasta cada unidad de disco individual en entornos de almacenamiento y servidores físicos y virtuales, lo que ayuda a garantizar el rendimiento óptimo y la disponibilidad de las aplicaciones comerciales críticas.

- ✓ Monitorear de manera continua el entorno de almacenamiento relacionado con el rendimiento, la conectividad, las estadísticas del entorno y las condiciones del espacio. Informa de cualquier problema relacionado con la disponibilidad o el rendimiento, según las políticas predefinidas.

- ✓ Optimizar el proceso de aprovisionamiento con una sola herramienta de administración en una infraestructura de almacenamiento heterogéneo.

- ✓ La aplicación de políticas de servicios de calidad a toda la ruta de datos ofrece a los administradores una visibilidad completa de la relación entre las aplicaciones y los recursos.

EMC VisualSRM.

EMC VisualSRM es un software de clase central de datos, creado específicamente para brindar una administración de recursos de almacenamiento centralizado para entornos de acaparamiento de nivel medio. Ofrece funcionalidades avanzadas para infraestructuras de varios proveedores y plataformas.

Ventajas de EMC VisualSRM:

- ✓ Administrar de manera eficaz la cantidad y el tamaño en aumento de archivos mientras se eliminan los gastos recurrentes.
- ✓ Seguir el consumo de almacenamiento individual y del sistema para garantizar que la capacidad esté disponible cuando y donde se la necesite.
- ✓ Reducir o eliminar interrupciones del servidor por consumo descontrolado del espacio del disco.

- ✓ Implementar la automatización de la administración de almacenamiento basada en políticas para garantizar que se cumplen con los niveles de servicio y que no hay riesgo de pérdida de datos.
- ✓ Aumentar la utilización del almacenamiento mientras se ayuda a reducir el costo total de la propiedad de la solución de administración de almacenamiento (Dell, 1999).

1.4.2 Soluciones existentes en Cuba.

Sistema para el control autónomo de espacio en disco en servidores de media.

Es una aplicación autónoma para el manejo de espacio de memoria en los servidores en el sistema Suria. Su principal objetivo es controlar la capacidad en disco de un servidor de medias y tomar decisiones dependiendo de reglas que se hayan definido, logrando un control estricto de las medias.

Ventajas del Sistema para el control autónomo de espacio en disco en servidores de media:

- ✓ Inspeccionan de manera automática el monitoreo de la capacidad del servidor y brinda un informe del estado de este.
- ✓ Permite tomar decisiones dependiendo de las acciones programadas a atender.
- ✓ Disminuye el costo correspondiente al personal utilizado en la administración de estos sistemas.
- ✓ Muestra el espacio libre en un dispositivo de almacenamiento, lo que permite reducir los gastos y obtener una mejor planificación de la capacidad disponible (Álvarez, 2011).

XYMA SAFE VISION.

Es un producto desarrollado por DATYS empresa perteneciente al MININT desde el año 2005. Este producto es un software de video vigilancia profesional basado en tecnología IP, con un alto grado de modularidad, que permite realizar de manera segura la monitorización y control en tiempo real y de forma histórica de cada uno de los movimientos que ocurren en las áreas que se identifiquen (DATYS, 2010).

Ventajas de XYMA SAFE VISION:

- ✓ Tiene independencia del hardware.

- ✓ Admite grabar y ver varias cámaras continuamente.
- ✓ Realiza una traza de las incidencias del sistema y su uso.
- ✓ El módulo de Alertas actúa como un vigilante, chequeando el funcionamiento de los módulos del sistema, y generando avisos a los usuarios y administradores de los eventos detectados.
- ✓ Permitir expandir el sistema a las necesidades del cliente.
- ✓ Reportes sobre las operaciones y sucesos.
- ✓ Adaptable a una gran cantidad de entornos, flexible, escalable y con el uso de tecnologías analógicas (DATYS, 2010).

1.4.3 Conclusiones de las aplicaciones existentes.

Una vez estudiadas las soluciones destinadas a la administración de almacenamiento en Cuba y en el mundo, se pudo observar que presentan características en común. Dentro de ellas se pueden mencionar:

- ✓ Monitorean de manera continua el entorno de almacenamiento.
- ✓ Implementan la automatización de la administración de almacenamiento basada en políticas.
- ✓ Realizan las acciones de copiar, mover, eliminar las medias.
- ✓ Permiten conocer cuánto espacio de almacenamiento está disponible en un dispositivo seleccionado.
- ✓ Realizan reportes.

A pesar de estos beneficios, existen inconformidades con las soluciones existentes en el mundo a la hora de ser adquiridas, pues la mayoría de ellas son privativas, altamente costosas y han sido desarrolladas para satisfacer las necesidades de un usuario específico y no de manera general. Algunas son imposibles de migrar hacia plataformas libres, dado que su código no permite adicionarle nuevas funcionalidades como NORTHERN, por lo cual no es recomendable utilizarlo. También hay soluciones como Veritas *CommandCentral Storage* y EMC VisualRSM, que presentan una compleja implementación, convirtiéndolo en un software poco flexible.

En el ámbito nacional, aunque el tema de video vigilancia ha avanzado en cuanto a equipamiento y software, aún se hallan descontentos con las soluciones desarrolladas. El país está inmerso en un difícil proceso de migración hacia el software libre que posibilite una soberanía tecnológica. El “Sistema para el control autónomo de espacio en disco en servidores de media” del sistema Suria, no

permite ser utilizado por otro sistema operativo que no sea Windows, porque aunque su implementación se realizó con el *framework* Qt, la biblioteca utilizada es una API de Windows, además Xyma SafeVision solo es recomendable instalar en el Sistema Operativo Windows XP SP2 y Windows 2003 o sistemas operativos superiores para ambos casos, por lo que no son soluciones convenientes como complemento de una aplicación que se pudiera utilizar en el sistema Suria en estos momentos.

Por todo lo anteriormente planteado, se concluye que las soluciones existentes no satisfacen las necesidades de los clientes, por lo que se hace necesario seguir trabajando en la búsqueda o creación de un software que permita la administración de los recursos de almacenamiento en el proyecto Video Vigilancia.

1.5 Metodología, herramientas y tecnologías de desarrollo.

La metodología, herramientas y tecnologías definidas para el desarrollo del módulo son el resultado de un estudio realizado por el arquitecto del proyecto y fueron aprobadas bajo el consenso de todo el equipo de desarrollo del mismo. El presente trabajo se rige por las selecciones realizadas con anterioridad.

1.5.1 Metodología de desarrollo del software.

Una metodología de desarrollo de software es un conjunto de pasos y procedimientos que deben seguirse para desarrollar software. La metodología a utilizar es el Proceso Unificado de Desarrollo de Software (RUP), por ser una metodología robusta, que facilita la organización de toda la documentación de los artefactos que se generan. Además soporta el Lenguaje Unificado de Modelado (UML) para el modelado y la herramienta Case[8] Visual Paradigm para la realización de los diagramas del módulo.

Proceso Unificado de Desarrollo de Software (RUP).

RUP es un proceso de desarrollo de software y junto con el UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. No

[8] **CASE**: (Computer Aided Software Engineering) Ingeniería de Software Asistida por Computación.

es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización. Incluye artefactos[9] y roles[10] (Phylum, 2012).

RUP se caracteriza por:

- ✓ Proceso Dirigido por los Casos de Uso: Se refiere a la utilización de los Casos de Uso para el desenvolvimiento y desarrollo de las disciplinas con los artefactos, roles y actividades necesarias. Los Casos de Uso son la base para la implementación de las fases y disciplinas RUP. Un Caso de Uso es una secuencia de pasos a seguir para la realización de un fin o propósito, y se relaciona directamente con los requisitos, ya que un caso de uso es la secuencia de pasos que conlleva la realización e implementación de un requisito planteado por el cliente.

- ✓ Proceso Iterativo e Incremental: Es el modelo utilizado por RUP para el desarrollo de un proyecto de software. Este modelo plantea la implementación del proyecto a realizar en Iteraciones, con lo cual se pueden definir objetivos por cumplir en cada iteración y así poder ir completando todo el proyecto iteración por iteración, con lo cual se tienen varias ventajas, entre ellas se pueden mencionar la de tener pequeños avances del proyecto que son entregables al cliente el cual puede probar mientras se está desarrollando otra iteración del proyecto, con lo cual el proyecto va creciendo hasta completarlo en su totalidad.

- ✓ Proceso Centrado en la Arquitectura: Define la arquitectura de un sistema, y una arquitectura ejecutable construida como un prototipo evolutivo. La arquitectura de un sistema es la organización o estructura de sus partes más relevantes. Una arquitectura ejecutable es una implementación parcial del sistema, construida para demostrar algunas funciones y propiedades. RUP establece refinamientos sucesivos de una arquitectura ejecutable, construida como un prototipo evolutivo.

Fases de RUP:

[9] **artefacto**: son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.

[10] **roles**: papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso.

- ✓ Conceptualización (Concepción o Inicio): Se describe el negocio y se identifican los casos de uso del sistema.
- ✓ Elaboración: se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.
- ✓ Construcción: El producto se desarrolla a través de iteraciones donde cada iteración involucra tareas de análisis, diseño e implementación. Se documenta tanto el sistema construido como el manejo del mismo. Se obtiene un producto listo para su utilización que está documentado y tiene un manual de usuario.
- ✓ Transición: Se libera el producto y se entrega al usuario para un uso real. Se incluyen tareas de marketing, empaquetado atractivo, instalación, configuración, entrenamiento, soporte, mantenimiento, etc. Se realizan pruebas y reparación de errores.
- ✓ Al finalizar cada fase se le presentan al cliente los artefactos definidos, es decir, el avance del proyecto para una evaluación de la calidad del mismo desde el punto de vista del cumplimiento o no con las necesidades planteadas por él.

1.5.2 Lenguaje de Modelado Unificado (UML 2.0).

UML es el Lenguaje Unificado de Modelado, está consolidado como el lenguaje estándar en el análisis y diseño de sistemas de cómputo. Mediante UML es posible establecer la serie de requisitos y estructuras necesarias para plasmar un sistema de software previo al proceso intensivo de escribir código (Osmosislatina, 2007). Posee ventajas que le propicia a los desarrolladores proveer un vocabulario y reglas que permiten una comunicación estándar, para visualizar, especificar, construir y documentar todo lo referente al proceso de desarrollo, añadiendo a esta última robustez, aumentando las posibilidades de obtener un producto con éxito y calidad.

Ventajas de UML:

- ✓ Visualizar: UML permite expresar de una forma gráfica un sistema de forma que otros desarrolladores lo puedan entender.
- ✓ Especificar: UML permite especificar cuáles son las características de un sistema antes de su construcción.
- ✓ Construir: A partir de los modelos especificados se pueden construir los sistemas diseñados y a la inversa (a partir del código fuente generar los modelos).

- ✓ Documentar: Los propios elementos gráficos sirven como documentación del sistema desarrollado para su futura revisión.

1.5.3 Herramientas CASE.

Las herramientas CASE son un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software. Como es sabido, los estados en el ciclo de vida de desarrollo de un software son: investigación preliminar, análisis, diseño, implementación e instalación (Reyna, 2010).

Visual Paradigm 8.0.

Como herramienta CASE se utilizó Visual Paradigm, herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Soporta los principales estándares de la industria tales como Lenguaje de Modelado Unificado (UML) lo cual ayuda a una rápida construcción de aplicaciones de calidad. Además permite dibujar todos los diagramas de clases, código inverso, generar código desde diagramas y generar documentación (Visual Paradigm, 2011).

Visual Paradigm ofrece:

- ✓ Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ✓ Capacidades de ingeniería directa (versión profesional) e inversa.
- ✓ Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- ✓ Disponibilidad en múltiples plataformas.

1.5.4 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 pueda comunicarse con los dispositivos hardware y software existentes (Soto, 2010).

C++.

El lenguaje de programación utilizado fue C++, diseñado a mediados de los años 1980 por Bjarne Stroustrup. La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido. Posteriormente se añadieron facilidades de programación genérica, que se sumó a los otros dos paradigmas que ya estaban admitidos (programación estructurada y la programación orientada a objetos). Por esto se suele decir que el C++ es un lenguaje de programación multi paradigma (Brokken, 2012).

1.5.5 Entorno de Desarrollo Integrado (IDE).

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación. Ofrece un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, entre otras (EcuRed, 2012).

Características de un IDE:

- ✓ Multiplataforma.
- ✓ Soporte para diversos lenguajes de programación.
- ✓ Reconocimiento de Sintaxis
- ✓ Extensiones y Componentes para el IDE
- ✓ Integración con *Framework*.
- ✓ Depurador.
- ✓ Importar y Exportar proyectos.
- ✓ Múltiples idiomas.
- ✓ Manual de Usuarios y Ayuda.

Qt Creator 2.5.0.

Qt Creator es un IDE multi-plataforma, se ejecuta en Windows, Linux y Mac OS, sistemas operativos de escritorio, y permite a los desarrolladores crear aplicaciones tanto para escritorio, como para plataformas de dispositivos móviles. Dentro de las características fundamentales de Qt Creator se encuentan: (Nokia Corporation, 2008)

- ✓ Editor de código sofisticado: Avanzado editor de código de Qt Creator, ofrece soporte para la edición de C + + y QML (Java Script), ayuda sensible al contexto, completado de código y navegación.
- ✓ Construcción y gestión proyecto: Si se importa un proyecto existente o crear uno desde cero, Qt Creator genera todos los archivos necesarios. Apoyo a CMake y compilación cruzada, con qmake está incluido.
- ✓ Aplicaciones de escritorio y móvil: Qt Creator ofrece soporte para crear y ejecutar aplicaciones Qt para equipos de escritorio y dispositivos móviles. Las opciones de construcción permiten cambiar rápidamente los destinos de generación.

1.5.6 XML-RPC como tecnología de comunicación.

XML–RPC es un protocolo de llamada remota a procedimientos que funcionan sobre Internet, mediante el intercambio de mensajes entre cliente y servidor, utilizando el protocolo HTTP para el transporte de los mensajes, además constituye un lenguaje de mensajería basada en XML (XMLRPC, 1998).

Ventajas de XML-RPC:

- ✓ Herramienta factible que garantiza el desarrollo eficiente del sistema,
- ✓ Garantiza compatibilidad y alta capacidad de integración.
- ✓ En cuestiones de licencia no existen limitantes, pues esta tecnología es totalmente libre,
- ✓ Utiliza una biblioteca llamada *libqxmlrpc* para que aplicaciones desarrolladas en el *framework* Qt puedan utilizar este protocolo estandarizado desde hace décadas, integrado al IDE seleccionado para el desarrollo del módulo.

1.6 Conclusiones parciales.

- ✓ Se establecieron los principales conceptos del dominio del problema actual, permitiendo un mejor entendimiento de los procesos autónomos de gestión para el control de almacenamiento en servidores de media.

- ✓ Se realizó un análisis de las soluciones existentes en el mundo relacionadas con la gestión de almacenamiento en servidores de media, lo que permitió determinar las características

principales que persiguen las diversas aplicaciones destinadas a la administración de recursos de almacenamiento, analizándose detalladamente sus ventajas y desventajas.

- ✓ Se caracterizaron la metodología, herramientas y tecnologías a utilizar en el desarrollo del módulo, analizándose las principales características que ofrecen cada una de ellas, permitiendo determinar que el módulo a desarrollar sea escalable y multiplataforma.

CAPÍTULO II: CARACTERÍSTICAS DEL MÓDULO.

Introducción.

En este capítulo, se ofrecen las características fundamentales del módulo. Se estudia el entorno donde se desarrollan los procesos del negocio, proporcionando los conceptos relacionados con la situación problemática planteada, los cuales son representados en el modelo de dominio. Se identifican los requisitos funcionales y no funcionales facilitando la presentación del diagrama de casos de uso y sus especificaciones correspondientes.

2.1 Modelo de dominio.

Una vez estudiado el entorno donde se encuentra el módulo a desarrollar, al no tenerse bien definidos los procesos de negocio, ni los responsables de estos procesos, por la existencia de solapamiento de responsabilidades y la dificultad en el establecimiento de las reglas del funcionamiento del producto a implementar, se determina realizar un modelo de dominio.

2.2.1 Descripción del diagrama del modelo de domino.

El sistema Suria brinda una plataforma para la vigilancia de cualquier entorno que se desee proteger, además garantiza el monitoreo utilizando cámaras IP de distintos fabricantes. Las cámaras capturan flujos de videos en tiempo real que son almacenados en un espacio de almacenamiento destinado al guardado de los mismos. Una vez ubicadas allí una gran cantidad de flujos de videos, ocasionan la saturación del servidor, por lo que se hace necesario realizarle actividades programadas tales como: mover, copiar o eliminar. Los usuarios pueden gestionar los espacios de almacenamiento de los diferentes dispositivos y las políticas de ejecución, a través de las cuales se controlan los espacios de almacenamiento, garantizando la disponibilidad de los recursos de almacenamiento y evitando la pérdida de información.

2.1.2 Diagrama de modelo de dominio.

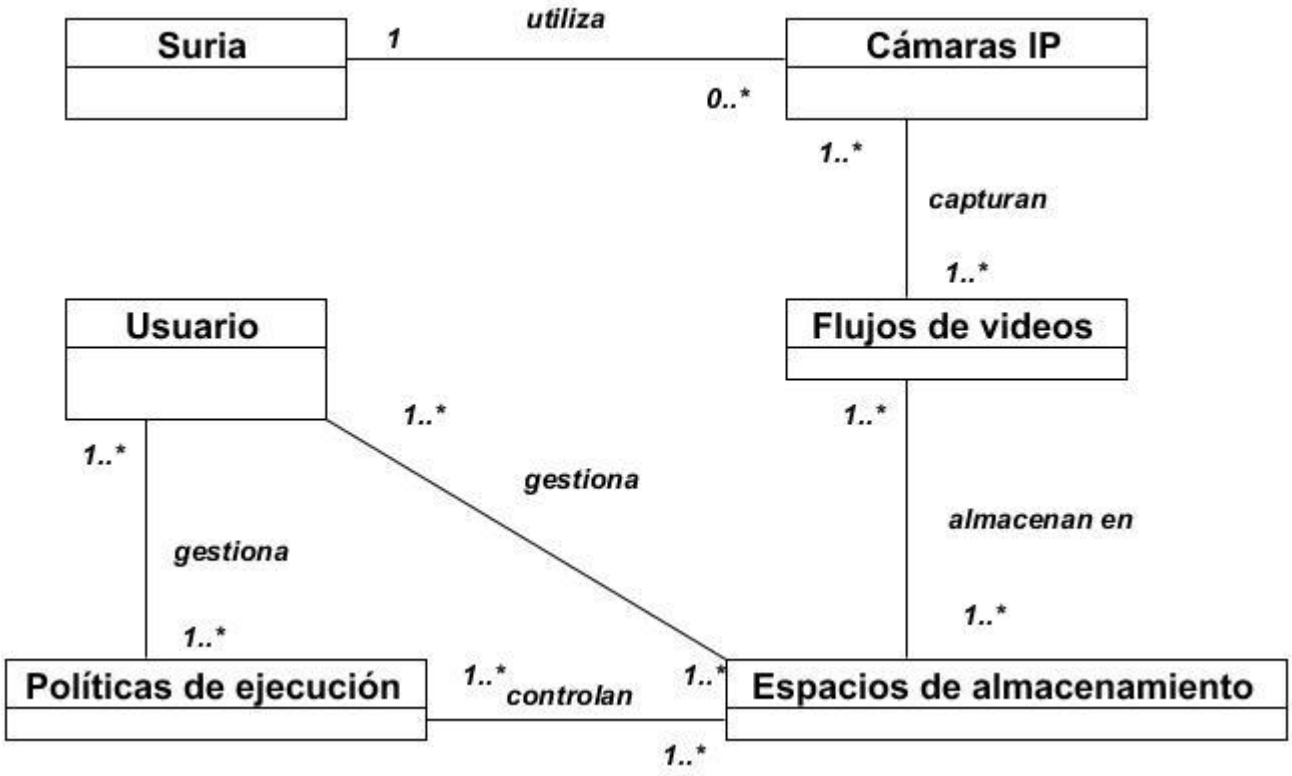


Figura 3. Diagrama del modelo de dominio.

2.1.3 Descripción de las clases del modelo de dominio.

- **Suria**: sistema para la video vigilancia que brinda una plataforma para la vigilancia de cualquier entorno que se desee proteger, mediante el monitoreo de cámaras IP de distintos fabricantes. El mismo está compuesto por seis módulos (análisis, recuperador, gestor, autonomía, visor y grabador).

- **Cámaras IP**: hardwares que brindan la posibilidad de obtener los flujos de videos de un área determinada mediante la red.

- **Flujos de videos**: representan las secuencias de las imágenes capturadas por cámaras IP.

- **Usuario**: personas con permisos que interactúan con las funcionalidades que brinda el módulo con el objetivo de obtener un buen funcionamiento del mismo.

- **Políticas de ejecución**: son los criterios que se tienen en cuenta para ejecutar diferentes acciones ante determinadas situaciones, logrando la autonomía del módulo.

- **Espacio de almacenamiento**: es toda área de almacenamiento donde se puede leer o escribir datos de forma permanente o temporal.

2.3 Requisitos del módulo.

Una etapa inicial y muy importante dentro del proceso de la Ingeniería de Software, es la Ingeniería de Requisitos, donde se lleva a cabo el proceso de descubrir, analizar, describir y verificar los servicios y restricciones, del módulo de software que se desea producir. Este proceso se realiza mediante la obtención, el análisis, la especificación, la validación y la administración de los requisitos del software. Los requisitos de software son las necesidades de los clientes y las restricciones en las que debe operar la solución (Medina, 2004).

2.3.1 Requisitos funcionales del módulo.

Los requisitos funcionales representan las funcionalidades que el módulo debe ser capaz de realizar, incluyendo las acciones que pueden ser ejecutadas por el usuario y las condiciones extremas a determinar por el mismo para dar respuesta a todas las peticiones realizadas. Se utiliza el prefijo RF en su nomenclatura.

Las funcionalidades identificadas son:

RF1 Insertar dispositivos de almacenamiento físico.
Descripción: El módulo debe permitir a los usuarios insertar un dispositivo de almacenamiento físico y además especificar el directorio de grabación.

RF2 Modificar el directorio de grabaciones de un dispositivo de almacenamiento físico.
Descripción: El módulo debe permitir a los usuarios modificar el directorio de grabaciones de un dispositivo de almacenamiento físico insertado con anterioridad.

RF3 Eliminar un dispositivo de almacenamiento físico.
Descripción: El módulo debe permitir a los usuarios eliminar un dispositivo de almacenamiento físico insertado con anterioridad, eliminándose todas las reglas asociadas a dicho dispositivo.

RF4 Insertar un dispositivos de almacenamiento virtual[11].

Descripción: El módulo debe permitir a los usuarios insertar un dispositivo de almacenamiento virtual y además definir un tamaño que no sea mayor a la capacidad del dispositivo de almacenamiento físico donde se cree.

RF5 Modificar el tamaño de un dispositivo de almacenamiento virtual.

Descripción: El módulo debe permitir a los usuarios cambiar el tamaño de un dispositivo de almacenamiento virtual insertado con anterioridad, el cual no puede ser mayor a la capacidad del dispositivo de almacenamiento físico donde fue creado.

RF6 Eliminar un dispositivo de almacenamiento virtual.

Descripción: El módulo debe permitir a los usuarios eliminar un dispositivo de almacenamiento virtual insertado con anterioridad, eliminándose todas las reglas asociadas a dicho dispositivo.

RF7 Insertar reglas de administración de almacenamiento.

Descripción: El módulo debe permitir a los usuarios insertar reglas de administración de almacenamiento, especificando el desencadenador que realizará la acción de mover, copiar o eliminar de la unidad o a las unidades de almacenamiento seleccionadas para gestionar los espacios donde se almacenan las medias.

RF8 Eliminar reglas de administración de almacenamiento.

Descripción: El módulo debe permitir a los usuarios eliminar una regla de administración de almacenamiento, insertado con anterioridad.

RF9 Mover medias.

Descripción: El módulo verifica que la acción a ejecutar es mover y mueve las medias que cumplen las condiciones definidas previamente en las reglas asociadas.

RF10 Eliminar medias.

Descripción: El módulo verifica que la acción a ejecutar es eliminar y elimina las medias que cumplan con las condiciones definidas previamente en las reglas asociadas.

[11] **almacenamiento virtual**: Crear una carpeta en una unidad de almacenamiento física.

RF11 Copiar medias.

Descripción: El módulo verifica que la acción a ejecutar es copiar y copia las medias que cumplen las condiciones definidas previamente en las reglas asociadas.

RF12 Generar reporte Reglas que más se ejecutan en un período.

Descripción: El módulo debe permitir generar un reporte para conocer las siete reglas que más han sido ejecutadas en un período determinado.

RF13 Generar reporte Unidad más vulnerable al llenado de almacenamiento.

Descripción: El módulo debe permitir generar un reporte que muestre las unidades que más han ejecutado el desencadenador de Disco lleno.

RF14 Generar *logs* de dispositivos almacenamiento físico.

Descripción: El módulo debe permitir generar y archivar en el fichero de *logs* las operaciones de los dispositivos de almacenamiento físico a medida que se ejecuten.

RF15 Generar *logs* de dispositivos almacenamiento virtual.

Descripción: El módulo debe permitir generar y archivar en el fichero de *logs* las operaciones de los dispositivos de almacenamiento virtual a medida que se ejecuten.

RF16 Generar *logs* de reglas de administración de almacenamiento.

Descripción: El módulo debe permitir generar y archivar en el fichero de *logs* las operaciones de las reglas de administración de almacenamiento a medida que se ejecuten.

RF17 Generar *logs* de reportes de autonomía.

Descripción: El módulo debe permitir generar y archivar en el fichero de *logs* las operaciones de los reportes de autonomía a medida que se ejecuten.

RF18 Generar *logs* de administrar medias.

Descripción: El módulo debe permitir generar y archivar en el fichero de *logs* las operaciones de copiar, mover y eliminar medias a medida que se ejecuten.

RF19 Generar *logs* de configuración de la conexión con el sistema Suria.

Descripción: El módulo debe permitir generar y archivar en el fichero de *logs* las operaciones de configuración de la conexión con el sistema Suria a medida que se ejecuten.

RF20 Filtrar por Todas las Unidades.

Descripción: El módulo debe permitir a los usuarios filtrar por todas las unidades obteniéndose como resultado todas las unidades físicas y virtuales que han sido registradas previamente.

RF21 Filtrar por Unidades Virtuales.

Descripción: El módulo debe permitir a los usuarios filtrar por las unidades virtuales obteniéndose como resultado todas las unidades virtuales que han sido registradas previamente.

RF22 Filtrar por Unidades Físicas.

Descripción: El módulo debe permitir a los usuarios filtrar por las unidades físicas obteniéndose como resultado todas las unidades físicas que han sido registradas previamente.

RF23 Filtrar por Todas las Reglas.

Descripción: El módulo debe permitir a los usuarios filtrar por todas las reglas obteniéndose como resultado todas las reglas de administración de almacenamiento.

RF24 Filtrar por Unidad.

Descripción: El módulo debe permitir a los usuarios mostrar como resultad todas las reglas de administración de almacenamiento correspondiente a la unidad seleccionada en el filtro.

RF25 Monitorizar el estado de los dispositivos de almacenamiento.

Descripción: El módulo debe monitorizar el estado de los dispositivos de almacenamiento verificando constantemente si se desencadena alguna regla.

RF26 Visualizar el estado de los dispositivos de almacenamiento.

Descripción: El módulo debe permitir a los usuarios visualizar el estado de los dispositivos de almacenamiento, es decir, mostrar todos los datos relacionados con el dispositivo seleccionado.

RF27 Visualizar la ejecución del estado de los proceso mover, copiar y eliminar medias.

Descripción: El módulo debe permitir visualizar el estado de ejecución de los procesos de mover, copiar y eliminar medias, mostrando el progreso de la acción que se encuentra en ejecución.

RF28 Insertar configuración de la conexión con el sistema Suria.

Descripción: El módulo debe permitir a los usuarios insertar la configuración de la conexión con el sistema Suria, guardándolo en un fichero de configuración.

RF29 Editar configuración de la conexión con el sistema Suria.

Descripción: El módulo debe permitir a los usuarios modificar la configuración de la conexión con el sistema Suria, insertada con anterioridad, actualizando el fichero de configuración.

RF30 Notificar sucesos.

Descripción: El módulo debe permitir notificar al sistema Suria los sucesos que ocurran cuando no se ha cumplido con las reglas de validación de campos requeridos en los formularios y/o cuando ocurre algún suceso relevante como pérdidas de conexión.

2.3.2 Requisitos no funcionales del módulo.

Los requisitos no funcionales son propiedades o cualidades que el módulo debe poseer. Son características que hacen al producto atractivo, usable, rápido o confiable. Se utiliza el prefijo RNF en su nomenclatura. Estos requisitos se agrupan en varias categorías:

Usabilidad.

RNF1 El módulo debe permitir ser utilizado por el usuario del proyecto Video Vigilancia.

RNF2 Se debe hacer uso de botones con imágenes que indiquen de modo intuitivo la función que realizan.

Fiabilidad.

RNF3 El módulo debe mostrar mensajes a los usuarios cuando no se cumpla con las reglas de validación de campos requeridos en los formularios y/o cuando ocurre algún suceso relevante como pérdidas de conexión o error.

RNF4 El módulo debe estar disponible de forma permanente, aunque requiere de la intervención de una persona con permisos para inicializar de forma manual el funcionamiento del módulo cuando ocurre algún fallo del fluido eléctrico o pérdida de la conexión.

Diseño e Implementación.

RNF5 El módulo debe ser implementado en C++, ajustándose a las funcionalidades del *framework* de desarrollo Qt.

Soporte.

RNF6 El módulo debe permitir ser integrado al sistema Suria y su arquitectura garantizar la incorporación de nuevas funcionalidades, asegurando su escalabilidad y logrando mejores prestaciones de servicios a los usuarios.

Interfaz de usuario.

RNF7 Se requiere que el módulo tenga interfaces gráficas que permita la interacción del usuario de forma amena e intuitiva.

Funcionamiento.

RNF8 El módulo debe incluir las bibliotecas de Qt versión 4.8.0 o superior al ser liberado para garantizar su correcto funcionamiento.

RNF9 El módulo debe ser multiplataforma.

2.4 Definición de los casos de uso.

Los casos de uso (CU) son la agrupación de uno o más requisitos funcionales, lo cuales son representados en el diagrama de casos de uso. Estos son inicializados por los actores del módulo.

2.4.1 Definición de los actores.

Un actor del módulo no es más que un conjunto de roles que los usuarios desempeñan cuando interaccionan con los casos de usos. Definiéndose como actores del módulo el sistema y el usuario.

Actor	Descripción
Sistema	Representa los procesos autónomos que se ejecutan en el módulo.
Usuario	Representa a las personas con los permisos necesarios para acceder a todas las funcionalidades del módulo.

Tabla 1. Actores del módulo.

2.4.2 Listado de los Casos de Uso.

CU Gestionar configuración de la conexión con el sistema Suria.

RF28 Insertar configuración de la conexión con el sistema Suria.

RF29 Editar configuración de la conexión con el sistemas Suria.

RF19 Generar *logs* de configuración de la conexión con el sistema Suria.

CU Administrar medias.

RF9 Mover medias.

RF10 Eliminar medias.

RF11 Copiar medias.

RF18 Generar *logs* de administrar medias.

CU Gestionar dispositivos de almacenamiento físico.

RF1 Insertar dispositivos de almacenamiento físico.

RF2 Modificar el directorio de grabaciones de un dispositivo de almacenamiento físico.

RF3 Eliminar dispositivos de almacenamiento físico.

RF14 Generar *logs* de dispositivos almacenamiento físico.

CU Gestionar dispositivo de almacenamiento virtual.

RF4 Insertar un dispositivos de almacenamiento virtual.

RF5 Modificar el tamaño de un dispositivo de almacenamiento virtual.

RF6 Eliminar dispositivos de almacenamiento virtual.

RF15 Generar *logs* de dispositivos almacenamiento virtual.

CU Gestionar reglas de administración de almacenamiento.

RF7 Insertar reglas de administración de almacenamiento.

RF8 Eliminar reglas de administración de almacenamiento.

RF16 Generar *logs* de reglas de administración de almacenamiento.

CU Monitorizar el estado de los dispositivos de almacenamiento.

RF25 Monitorizar el estado de los dispositivos de almacenamiento.

CU Generar reportes de autonomía.

RF12 Generar reporte Reglas que más se ejecutan en un período.

RF13 Generar reporte Unidad más vulnerable al llenado de almacenamiento.

RF17 Generar *logs* de reportes de autonomía.

CU Filtrar las unidades de almacenamiento.

RF20 Filtrar por Todas las Unidades.

RF21 Filtrar por Unidades Virtuales.

RF22 Filtrar por Unidades Físicas.

CU Filtrar reglas de administración de almacenamiento.

RF23 Filtrar por Todas las Reglas.

RF24 Filtrar por Unidad.

CU Visualizar el estado de los dispositivos de almacenamiento.

RF26 Visualizar el estado de los dispositivos de almacenamiento.

CU Visualizar la ejecución del estado de los proceso.

RF27 Visualizar la ejecución del estado de los proceso.

CU Notificar sucesos.

RF30 Notificar sucesos.

2.4.3 Diagrama de casos de usos.

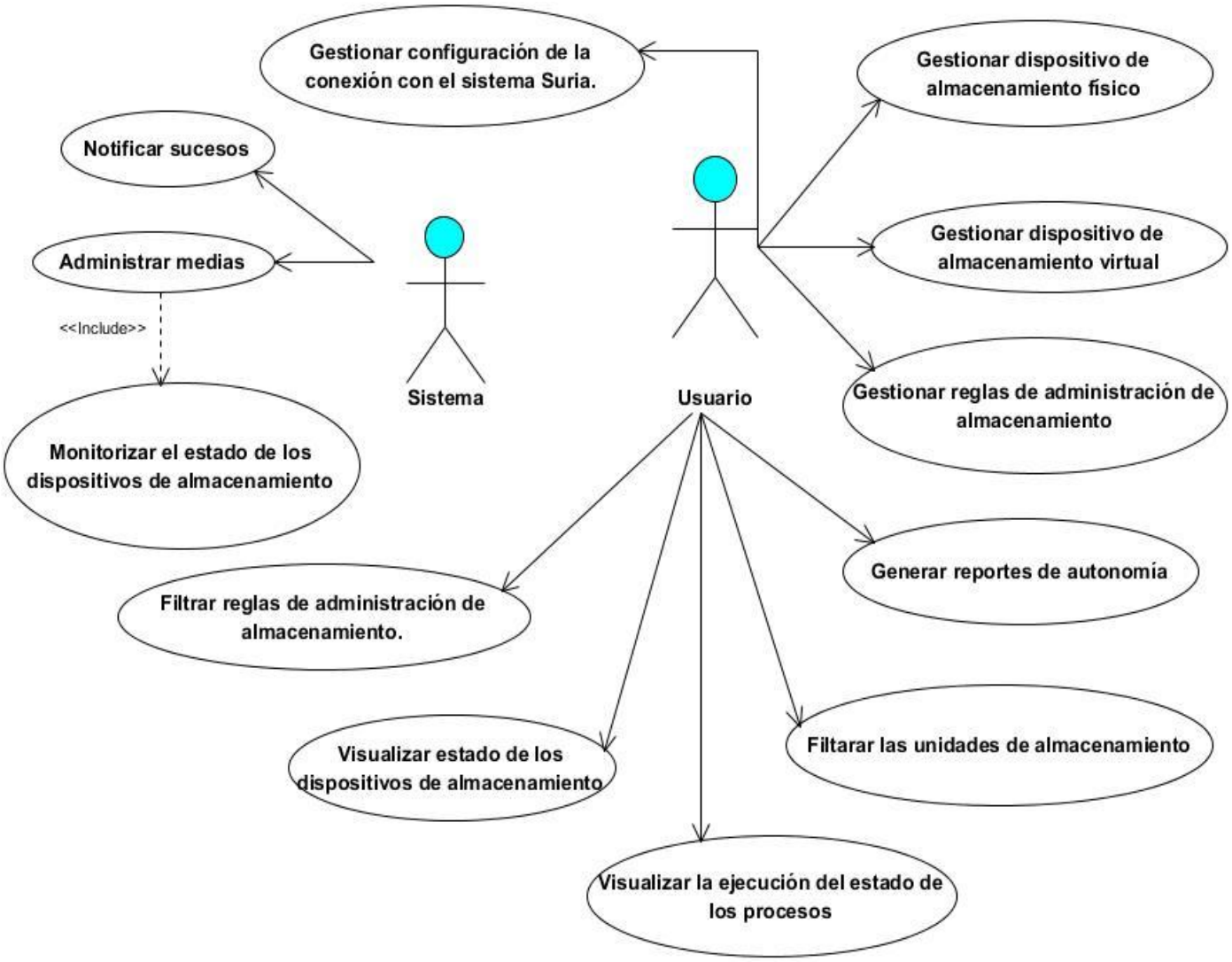


Figura 4. Diagrama de Casos de Usos.

2.5 Casos de Usos Expandidos.

En el presente epígrafe se especifican los casos de uso Gestionar reglas de administración de almacenamiento y Administrar medias. Los casos de usos expandidos restantes se pueden encontrar en el Anexo I del presente trabajo.

Caso de Uso:	Gestionar reglas de administración de almacenamiento.
Actores:	Usuario.
Resumen:	El caso de uso se inicia cuando el usuario selecciona en el módulo de autonomía la

	opción de insertar o eliminar una regla de administración de almacenamiento.
Precondiciones:	Exista conexión entre el módulo y el sistema Suria. Exista conexión con la base de datos.
Referencias	RF7, RF8, RF16.
Prioridad	Alta

Flujo Normal de Eventos "Gestionar reglas de administración de almacenamiento".	
Sección "Insertar reglas de administración de almacenamiento".	
Acción del Actor	**Respuesta del módulo**
1.- El usuario selecciona una de las siguientes opciones: - Crear regla de administración de almacenamiento. - Eliminar regla de administración de almacenamiento. (Ver Sección "Eliminar regla de administración de almacenamiento"). 2.- El usuario selecciona la opción de Crear regla de administración de almacenamiento. (Interfaz 1) (A).	3.- El módulo le muestra al usuario una ventana para seleccionar los datos: (Interfaz 2) B: Desencadenador. C: Siguiente. D: Cancelar.
4.- El usuario selecciona el desencadenador[12]. 5.- El usuario selecciona una de las siguientes opciones: (Interfaz 3) E: Agregar (Ver sección "Agregar"). F: Siguiente. G: Cancelar (Ver sección Cancelar). 6.- Si el usuario selecciona la opción Siguiente.	7.- El módulo le muestra al usuario la ventana para que seleccione las "Acciones a ejecutar.
8.- El usuario selecciona una de las siguientes acciones : (Interfaz 4) H: Borrar videos (Ver sección "Borrar videos"). I: Mover videos (Ver sección "Mover videos"). J: Copiar videos (Ver sección "Copiar videos"). 9.- El usuario selecciona: (Interfaz 4)	12.- El módulo muestra un resumen con los datos seleccionados de la regla. (Interfaz 5)

[12] Desencadenador: define el evento del módulo que inicia la ejecución de la regla.

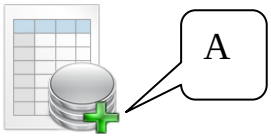
K: Tipo de video L: Porciento alcanzado para ejecutar la regla. 10.- El usuario selecciona una de las siguientes opciones: (Interfaz 4) M: Siguiente. N: Atrás. (Ver sección Atrás). Ñ: Cancelar. (Ver sección "Cancelar"). 11.- Si el usuario selecciona la opción "Siguiente".	
13.- El usuario selecciona una de las siguientes opciones: (Interfaz 5) O: Finalizar. P: Atrás. (Ver sección Atrás). Q: Cancelar. (Ver sección "Cancelar"). 14.- Si el usuario selecciona la opción "Finalizar".	15.- El módulo se conecta a Suria, crea una regla de administración de almacenamiento, actualiza el listado de las reglas, mostrando la regla creada (Interfaz 6) (R), deja en espera las acciones a ejecutar definidas y registra en el fichero de *log* esta acción. Termina así el caso de uso.
Flujos alternos Sección "Insertar reglas de administración de almacenamiento".	
4a. No selecciona el desencadenador.	
Acción del Actor	**Respuesta del módulo**
	4a.- El módulo muestra un mensaje al usuario informando que debe seleccionar un desencadenador. Regresa al paso 4 del flujo básico.
8a. No selecciona la acción a ejecutar.	
Acción del Actor	**Respuesta del módulo**
	8a.- El módulo muestra un mensaje al usuario informando que tienes que seleccionar una acción a ejecutar. Regresa al paso 8 del flujo básico.
9a. No selecciona el porciento alcanzado para ejecutar la regla.	
Acción del Actor	**Respuesta del módulo**
	9a.- El módulo muestra un mensaje al usuario informando que tiene que seleccionar el porciento alcanzado para ejecutar la regla. Regresa al paso 10

	del flujo básico.
14a. Regla solapada.	
Acción del Actor	**Respuesta del módulo**
	14a.- El módulo muestra un mensaje al usuario informando que la regla no puede ser insertada porque se solapa con una regla ya creada. Termina así el caso de uso.
Sección "Agregar".	
Acción del Actor	**Respuesta del módulo**
	2.- El módulo muestra al usuario una ventana para agregar una o varias unidades que desencadenarán la ejecución de la regla.
3.- El usuario selecciona una la unidad que desea agregar (Interfaz 7) (S) 4.- El usuario selecciona una de las siguientes opciones: (Interfaz 7) T: Aceptar. U: Cancelar (Ver Sección "Cancelar"). 5.- Si el usuario selecciona la opción Aceptar.	6.- El módulo agrega y muestra la unidad que desencadenarán la ejecución de la regla. (Interfaz 7)(V)
7.- El usuario selecciona una de las siguientes opciones: (Interfaz 2) W: Agregar (Regresar al paso 3 del flujo básico). X: Eliminar (Ver sección "Eliminar"). Y Limpiar (Ver sección "Limpiar"). Z: Siguiente. 1: Cancelar (Ver Sección "Cancelar"). 8.- Si el usuario selecciona la opción Siguiente.	9.- El módulo le muestra al usuario la ventana siguiente para que seleccione las Acciones a ejecutar. Regresa el paso 9 del flujo básico.
Flujo alterno Sección "Agregar".	
3a. No selecciona la unidad.	
Acción del Actor	**Respuesta del módulo**

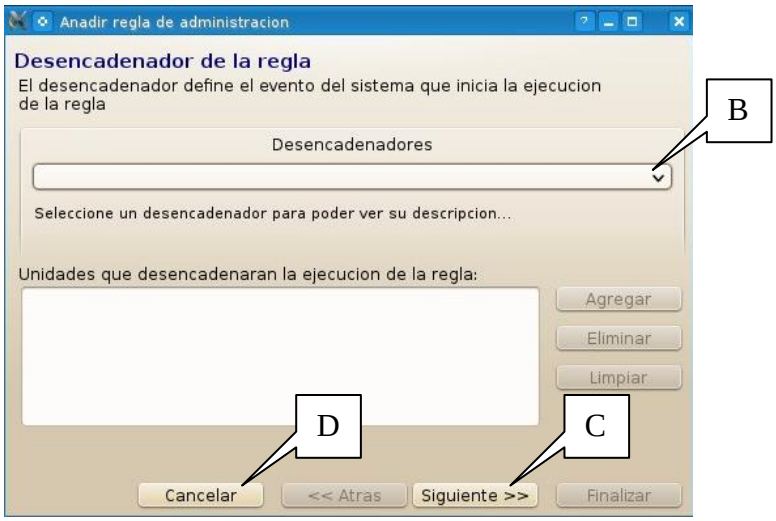
	3a.- El módulo muestra un mensaje al usuario informando que deben seleccionar al menos una unidad. Regresa al paso 3 del flujo básico.
Sección "Eliminar".	
Acción del Actor	**Respuesta del módulo.**
	2.- El módulo muestra al usuario un mensaje de confirmación "Desea eliminar la unidad seleccionada".
3.- El usuario selecciona una de las siguientes opciones: - Aceptar. - Cancelar (Ver Sección "Cancelar"). 4.- Si el usuario selecciona la opción Aceptar.	5.- El módulo elimina la unidad seleccionada. Regresa al paso 4 del flujo básico.
Sección "Limpiar".	
Acción del Actor	**Respuesta del módulo.**
	2.- El módulo elimina todas las unidades que se encuentran en el área Unidades que desencadenarán la ejecución de la regla. Regresa al paso 4 del flujo básico.
Sección "Borrar videos".	
Acción del Actor	**Respuesta del módulo.**
	2.- Regresa al paso 10 del flujo básico.
Sección "Mover videos".	
Acción del Actor	**Respuesta del módulo**
	2.- El módulo activa el campo de las especificaciones.
3.- El usuario selecciona: 2: Unidad destino. (Interfaz 9).	4.- Regresa al paso 10 del flujo básico.
Sección "Copiar videos".	
Acción del Actor	**Respuesta del módulo**

	2.- El módulo activa el campo de las especificaciones.
3.- El usuario selecciona: 2: Unidad destino. (Interfaz 9).	4.- Regresa al paso 10 del flujo básico.
Poscondiciones:	El módulo inserta una regla de administración de almacenamiento, se visualiza la lista de reglas actualizada y se registra en el fichero de log esta acción.
Prototipo interfaz "Sección Insertar reglas de administración de almacenamiento".	

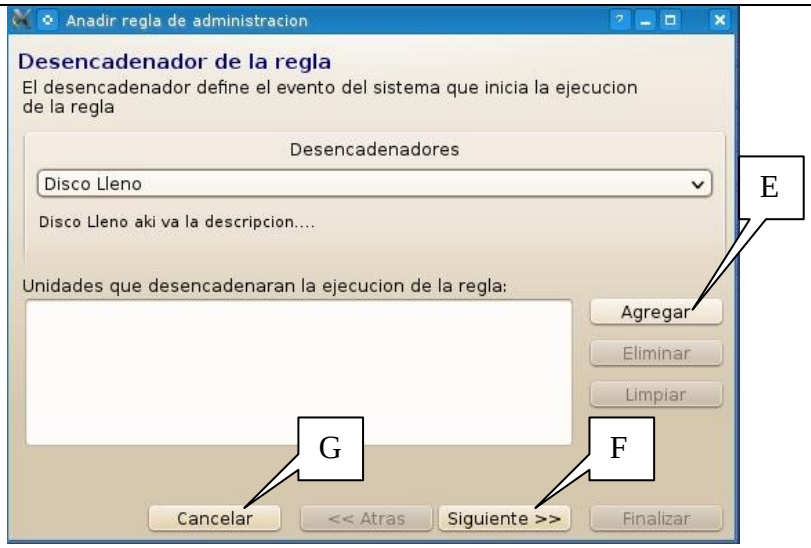
Interfaz 1



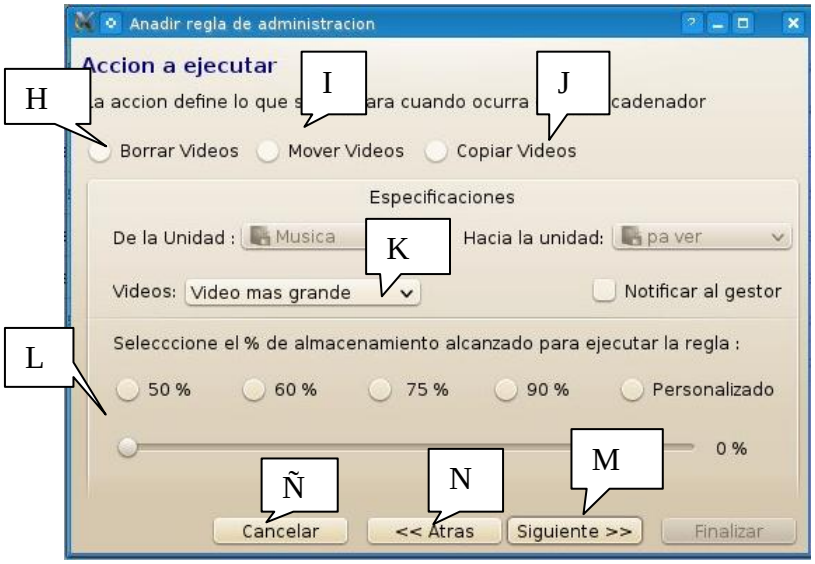
Interfaz 2



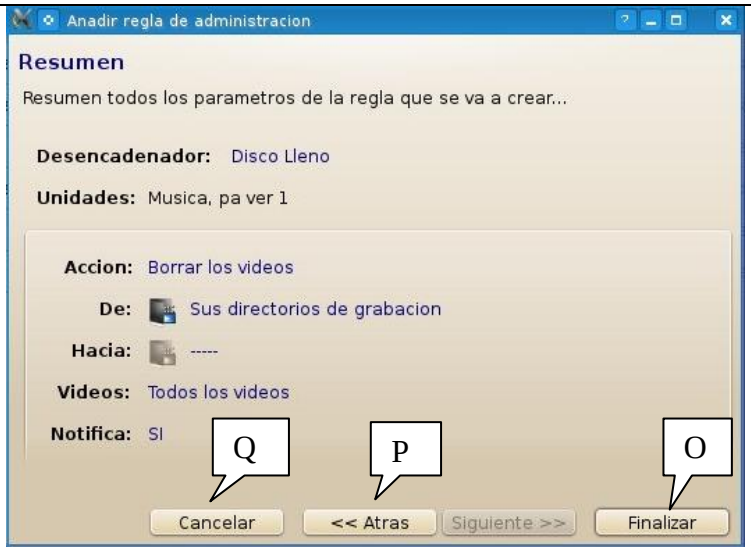
Interfaz 3



Interfaz 4



Interfaz 5

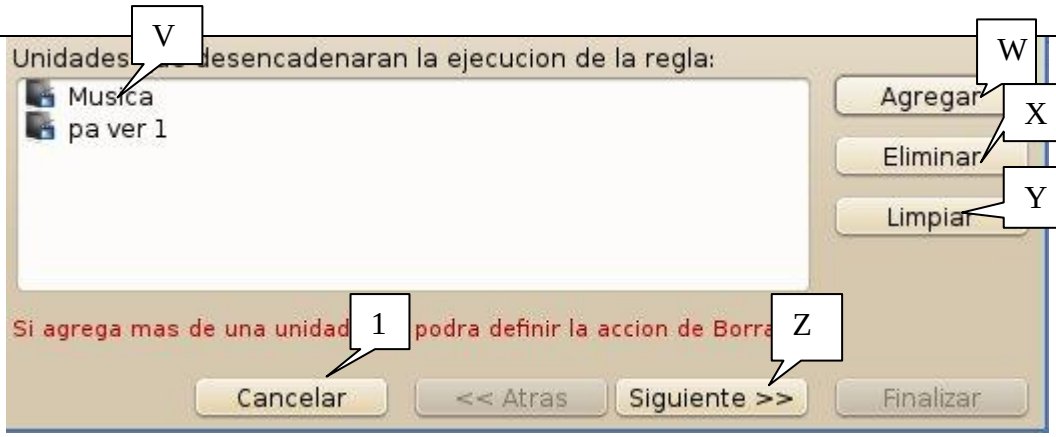


Interfaz 6

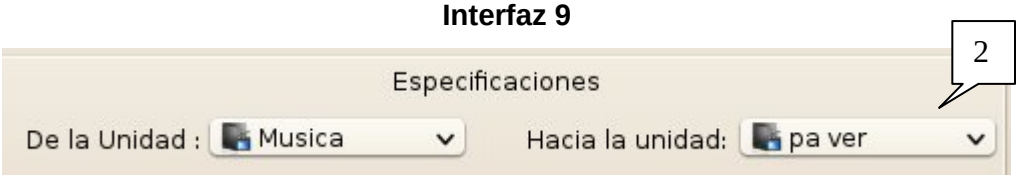
	Unidad	%	Desencadenador	Accion	Unidad DEstino	Videos	Notificacion
1	pa ver	95	Disco Lleno	Borrar	-----	Mas antiguo	SI
2	Videos	95	Disco Lleno	Borrar	-----	Mas antiguo	SI
3	pa ver	60	Espacio de almacenamiento Alcanzado	Borrar	-----	Video mas grande	SI
4	Datos	95	Disco Lleno	Borrar	-----	Todos los videos	SI
5	pa ver	50	Espacio de almacenamiento Alcanzado	Copiar	Datos...	Video mas grande	SI
6	pa ver 1	60	Espacio de almacenamiento Alcanzado	Borrar	-----	Video mas grande	SI

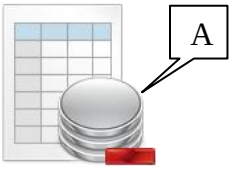
Interfaz 7

Interfaz 8



Interfaz 9



Sección "Eliminar reglas de administración de almacenamiento".	
Acción del Actor	**Respuesta del módulo**
2.- El usuario selecciona de la lista de reglas una regla y luego selecciona la opción de Eliminar una regla de administración de almacenamiento. (Interfaz 1) (A)	3.- El módulo muestra al usuario un mensaje de confirmación.
4.- El usuario selecciona una de las siguientes opciones: - Si. - No. (Ver sección "No"). 5.- Si selecciona Si.	6.- El módulo se conecta a Suria, elimina la regla deseada y con ella todos los procesos que se encuentren definido en la misma. Además visualiza el listado de reglas actualizado y registra en el fichero de *log* esta acción. Termina así el caso de uso.
Prototipo interfaz "Sección Insertar reglas de administración de almacenamiento".	
Interfaz 1 	
Sección "Cancelar".	
Acción del Actor	**Respuesta del módulo.**

1.- El usuario selecciona la opción “Cancelar”.	2.- El módulo cancela la operación, cierra la ventana y termina el caso de uso.
Sección “No”.	
Acción del Actor	**Respuesta del módulo.**
1.- El usuario selecciona la opción “No”.	2.- El módulo cancela la operación, cierra la ventana y termina el caso de uso.
Sección “Atrás”.	
Acción del Actor	**Respuesta del módulo.**
1.- El usuario selecciona la opción “Atrás”.	2. El módulo regresa a la ventana anterior.
Sección “Siguiente”.	
Acción del Actor	**Respuesta del módulo.**
1.- El usuario selecciona la opción “Siguiente”.	2. El módulo muestra la siguiente ventana.
Poscondiciones:	El módulo elimina una regla de administración de almacenamiento, se visualiza la lista de reglas actualizada y se registra en el fichero de *log* esta acción.

Tabla 2. Descripción del CU Gestionar reglas de administración de almacenamiento.

Caso de Uso:	Administrar medias.
Actores:	Sistema
Resumen:	El caso de uso se inicia cuando el módulo de autonomía se conecta con Suria y detecta un desencadenador activado. Luego verifica si la acción definida es copiar, mover o eliminar medias y ejecuta dicha acción.
Precondiciones:	Exista conexión entre el módulo y el sistema Suria. Exista conexión con la base de datos. El sistema debe haber detectado la activación de un desencadenador. Verifica la acción que tiene definida y ejecuta dicha acción sobre las medias.
Referencias	RF9, RF10, RF11, RF18
Prioridad	Alta.
Flujo Normal de Eventos “Administrar medias”.	
Sección “Mover medias”.	
Acción del Actor	**Respuesta del módulo.**

1.- Se activa una regla y se realiza la acción definida en ella: - Mover medias. - Eliminar medias. (Ver sección “Eliminar medias”). - Copiar medias. (Ver sección “Copiar medias”). 2.- Si la acción de la regla es mover medias.	3.- El módulo manda a mover las medias a la unidad seleccionada en las reglas de administración de almacenamiento, actualiza el estado de almacenamiento en la unidad origen y en la unidad destino y actualiza el estado de almacenamiento en la unidad origen y en la unidad destino. Termina el caso de uso.
Poscondiciones:	Se mueven las medias, se actualiza el estado de almacenamiento de las unidades origen y destino y se registra en el fichero de log esta acción.
Sección “Copiar medias”.	
Acción del Actor	**Respuesta del módulo**
	2.- El módulo manda a copiar las medias a la unidad seleccionada en las reglas de administración de almacenamiento, actualiza el estado de almacenamiento en la unidad destino y actualiza el estado de almacenamiento en la unidad destino. Termina el caso de uso.
Poscondiciones:	Se copian las medias, se actualiza el estado de almacenamiento de la unidad destino y se registra en el fichero de log esta acción.
Sección “Eliminar medias”.	
Acción del Actor	**Respuesta del módulo**
	2.- El módulo manda a eliminar las medias de la unidad seleccionada en las reglas de administración de almacenamiento, actualiza el estado de almacenamiento de la unidad y registra en el fichero de log esta acción. Termina el caso de uso.
Poscondiciones:	Se eliminan las medias, se actualiza el estado de almacenamiento de la unidad origen y se registra en el fichero de log esta acción.

Tabla 3. Descripción del CU Administrar medias.

2.6 Conclusiones parciales.

- ✓ Se obtuvo una representación visual del dominio del problema actual, identificándose los conceptos más importantes de la situación problemática, lo que permitió comprender el contexto en el cual se encuentra el módulo a implementar y su funcionamiento.

- ✓ Se identificaron los requisitos funcionales y no funcionales lo que generó un total de 13 casos de uso, describiendo las respuestas del módulo ante las acciones de los actores, facilitando a los desarrolladores una mejor comprensión a la hora de implementar las funcionalidades requeridas.

Capítulo III: Diseño del Módulo.

Introducción.

En este capítulo se define el diseño del módulo. Se especifica la estructura de organización de la solución a construir. Se explican los patrones de diseños empleados. Se realizan los diagramas de despliegue y de componentes. Se describen las pruebas de caja negra que se realizan con el propósito de evaluar la calidad del módulo.

3.1 Patrones.

Según G. Booch: "Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles." Los patrones están basados en la experiencia acumulada por los desarrolladores, con el objetivo de describir la solución más factible a problemas comunes, ayudando a no cometer errores consumados con anterioridad.

Se pueden agrupar en diferentes categorías según el nivel de abstracción (Booch, 2011):

- ✓ Patrones arquitectónicos: Aquellos que expresan un esquema organizativo estructural fundamental para sistemas software.
- ✓ Patrones de diseño: Aquellos que expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas software.
- ✓ Idiomas: Patrones de bajo nivel específicos para un lenguaje de programación o entorno concreto.

3.1.1 Arquitectura.

Arquitectura en capas.

El patrón arquitectónico en capas define cómo organizar el modelo de diseño en capas, que pueden estar físicamente distribuidas, lo cual quiere decir que los componentes de una capa sólo pueden hacer referencia a componentes en capas inmediatamente inferiores. Este patrón es importante porque simplifica la comprensión y la organización del desarrollo de sistemas complejos, reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores.

Arquitectura en dos capas

El sistema Suria cuenta entre sus módulos fundamentales con el de autonomía, que utiliza la base de datos del sistema para la persistencia y el acceso a los datos necesarios para su funcionamiento. Por tal motivo se utiliza la arquitectura en dos capas. La primera capa se denomina Capa de presentación y la segunda consiste en la Capa de lógica de negocio.

- ✓ Capa de presentación: También se le denomina "capa de usuario", es la que permite la comunicación entre el módulo y los usuarios, además, se encarga de obtener las peticiones de los usuarios, comunicárselas a la Capa de lógica de negocio y presentar los datos al usuario. Para realizar estas tareas se apoya en una serie de componentes visuales como son: formularios, botones, listas desplegables, entre otros. Para esto la Capa de presentación genera las vistas para que el usuario actúe sobre ellas y recoge los datos introducidos por el mismo y se lo comunica a la Capa de lógica de negocio mediante el uso de las funciones.

- ✓ Capa de lógica de negocio: Se denomina capa de lógica de negocio porque es aquí donde se establecen todas las reglas que deben cumplirse, es decir, es donde se gestiona y se realizan las operaciones solicitadas por el usuario, además de realizar todas las operaciones consecuentes. Esta capa maneja los datos, comprueba su autenticidad y los modifica de ser necesario, pero no se preocupa de su presentación, ni de su almacenaje (Reynoso, 2004).

Arquitectura basada en plugins.

Un plugin es la unidad mínima de funcionalidad que puede ser distribuida de manera separada. Herramientas pequeñas, se escriben como un único plugin, mientras que en las complejas la funcionalidad está en varios plugins. La arquitectura basada en plugins viene marcada por los puntos de extensión y las aportaciones de los plugins a la plataforma así como entre ellos (Fontanillo, 2005).

En el desarrollo del módulo las funcionalidades mover, copiar, eliminar y supervisar la capacidad de almacenamiento a los dispositivos disponibles se implementan mediante plugins, lo que brinda ventajas como:

- ✓ Extensibilidad del módulo: Se pueden agregar plugins al módulo para monitorizar y controlar los nuevos dispositivos de almacenamiento sin tener que recompilar todo el módulo.

- ✓ Ambiente controlado: Cada plugin implementa un ambiente controlado de visualización. En caso de que ocurra un error, puede ser descargado sin afectar al resto de la aplicación (Fontanillo, 2005).

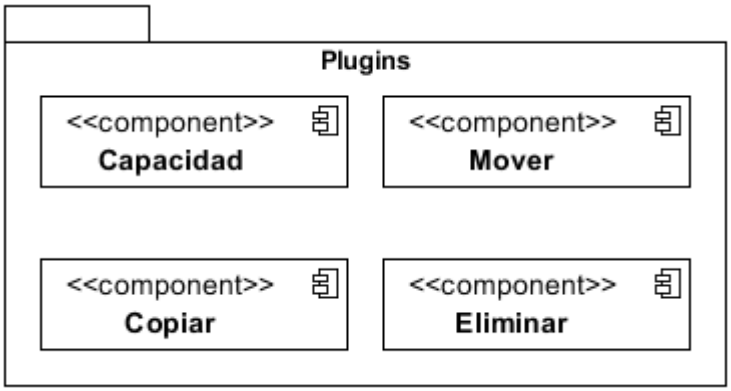


Figura 5. Plugins de capacidad, eliminar, copiar y mover.

3.1.2 Patrones Generales de Software para Asignar Responsabilidades (GRASP).

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a clases, expresados en forma de patrones. En el desarrollo del módulo se utiliza el patrón Experto en las clases DialogoDireccion, DialogoModificarTamano, DialogoSeleccionUnidades, Dispositivo, Regla y Procesos como el principio de que una clase debe poseer la información necesaria para cumplir con sus responsabilidades. Al utilizar el patrón Creador en las clases Pantalla, DialogoDireccion y DialogoAnadirUnidadFisica, donde Pantalla contiene un objeto de DialogoAnadirUnidadFisica y DialogoDireccion y a su vez DialogoAnadirUnidadFisica con tiene un objeto de DialogoDireccion; creando de esta manera instancias con las clases que colaboran para cumplir su responsabilidad, tratando de lograr siempre un Bajo acoplamiento y una Alta cohesión. Con el cumplimiento de estos dos últimos patrones se logra que cada clase recurra solamente a las clases que son imprescindibles para su trabajo y tenga asociada las responsabilidades que le corresponden de acuerdo con su comportamiento y el patrón Controlador, se estructura en el módulo con la clase Sistema que se encarga de los eventos y funcionalidades que representan los caso de usos.

3.1.3 Patrones Banda de Cuatros (GoF).

En el desarrollo del módulo de autonomía, además de la utilización de los patrones GRASP, también se emplearon los patrones GoF. Dentro de los patrones GoF se utiliza el patrón fachada a través de la clase Pantalla, teniendo una interfaz unificada que haga de intermediaria entre un usuario y el resto de las interfaces. Además se usó el patrón observador mediante la clase Plataforma, este patrón proporciona al módulo que cuando un objeto cambia de estado, se actualicen automáticamente todos los objetos relacionados con él.

3.2 Modelo de diseño.

El Modelo de diseño describe la realización física de los casos de uso centrándose en el cumplimiento de los requisitos funcionales y no funcionales. Además, permite tener una abstracción de la implementación del módulo, lo cual facilita el trabajo a los desarrolladores.

3.2.1 Diagrama de clases del diseño.

A continuación, se muestra el Diagrama de clases del diseño del módulo de autonomía agrupado en dos paquetes, con sus clases y relaciones correspondientes. En el Anexo II se encuentran el Diagrama de clase del diseño con sus métodos y atributos.

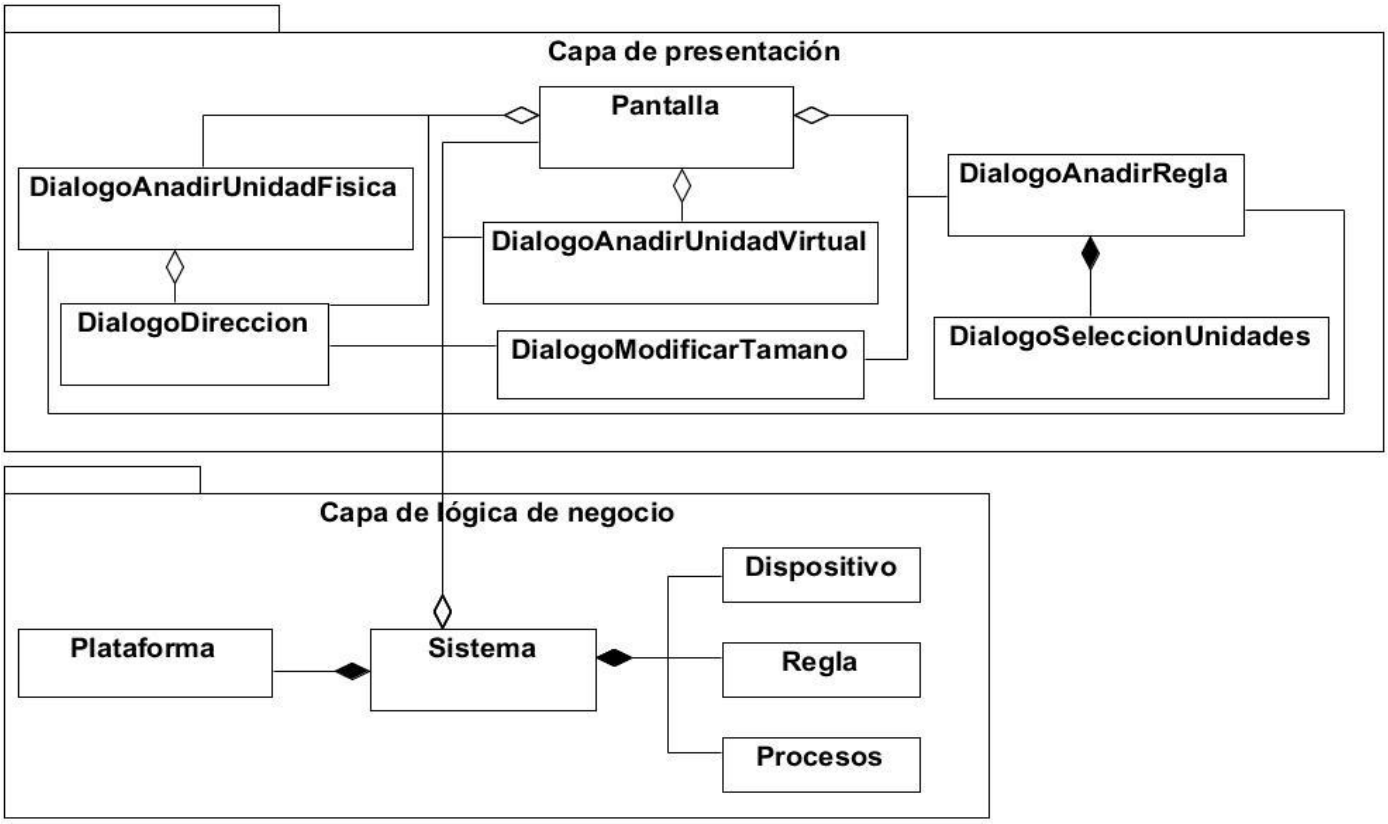


Figura 6. Diagrama de Clases del Diseño del Módulo de Autonomía.

El diagrama de clases del diseño del módulo autonomía se agrupó en dos paquetes, el paquete Capa de presentación y el de Capa de lógica de negicio. El paquete Capa de presentación contiene la clase Pantalla que permite al usuario acceder a las funcionalidades que brinda el módulo. La clase DialogoAnadirUnidadFisica permite al usuario insertar una unidad física, con la clase DialogoAnadirUnidadVirtual se puede insertar una unidad virtual, a través de la clase DialigoAnadirRegla puede insertar una regla, con la clase DialogoDireccion se selecciona y modifica el directorio de grabaciones de una unidad física, la clase DialogoSeleccionUnidades permite seleccionar la unidad que se le realizará el desencadenador seleccionado por el usuario cuando define una regla.

El paquete de Capa de lógica de negocio contiene la clase controladora Sistema, donde se encuentran implementadas todas las funcionalidades que debe cumplir el módulo. Esta clase contiene un objeto de plataforma, regla, dispositivo y procesos. La clase Plataforma se encarga de enviar las peticiones de los *plugins* a la plataforma y recibir los datos enviados por los *plugins* ejecutados. Las clases

Dispositivo, Regla y Procesos contienen los atributos y métodos necesarios para poder crear un objeto de su tipo, además la clase Dispositivo se encarga de calcular el porciento usado y el porciento disponible del dispositivo.

3.2.2 Diagrama de secuencia del diseño.

Los Diagramas de secuencia muestran en detalles como interactúan las clases y guían al programador durante la fase de implementación. Los Diagramas de secuencia se encuentran en el Anexo III.

3.3 Modelo de implementación.

El Modelo de implementación está compuesto por un conjunto de subsistemas y componentes que establecen la composición física de la implementación del módulo. Utiliza los Diagramas de despliegue y Diagrama de componentes para comprender cómo se organizan y dependen unos de otros.

3.3.1 Diagrama de despliegue.

El Diagrama de despliegue es utilizado para mostrar las relaciones físicas de los distintos nodos que componen el módulo. Describen la topología del módulo, la estructura de los elementos de *hardware* y el *software* que se ejecuta. A continuación se muestra en la siguiente figura el Diagrama de despliegue del módulo.

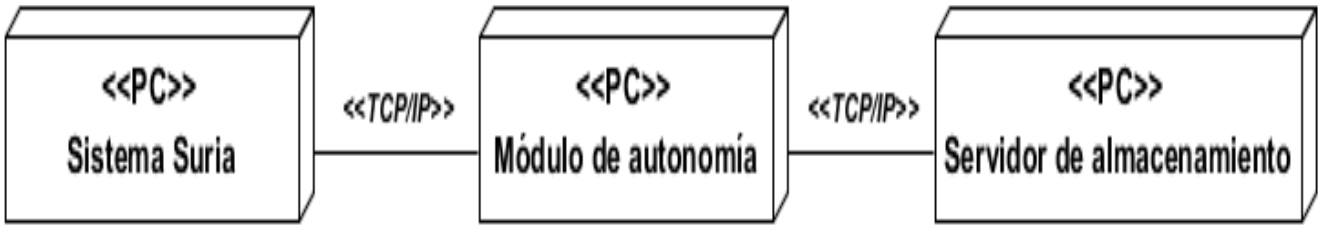


Figura 7. Diagrama de despliegue.

3.3.2 Diagrama de componentes.

El Diagrama de componentes describe los elementos físicos del módulo y sus relaciones. Muestra las opciones de realización incluyendo código fuente, binario y ejecutable. Los componentes representan todos los tipos de elementos de *software* que entran en la fabricación de aplicaciones informáticas.

Pueden ser simples archivos, paquetes y bibliotecas cargadas dinámicamente. A continuación, se muestra en la siguiente figura el Diagrama de componentes de la aplicación.

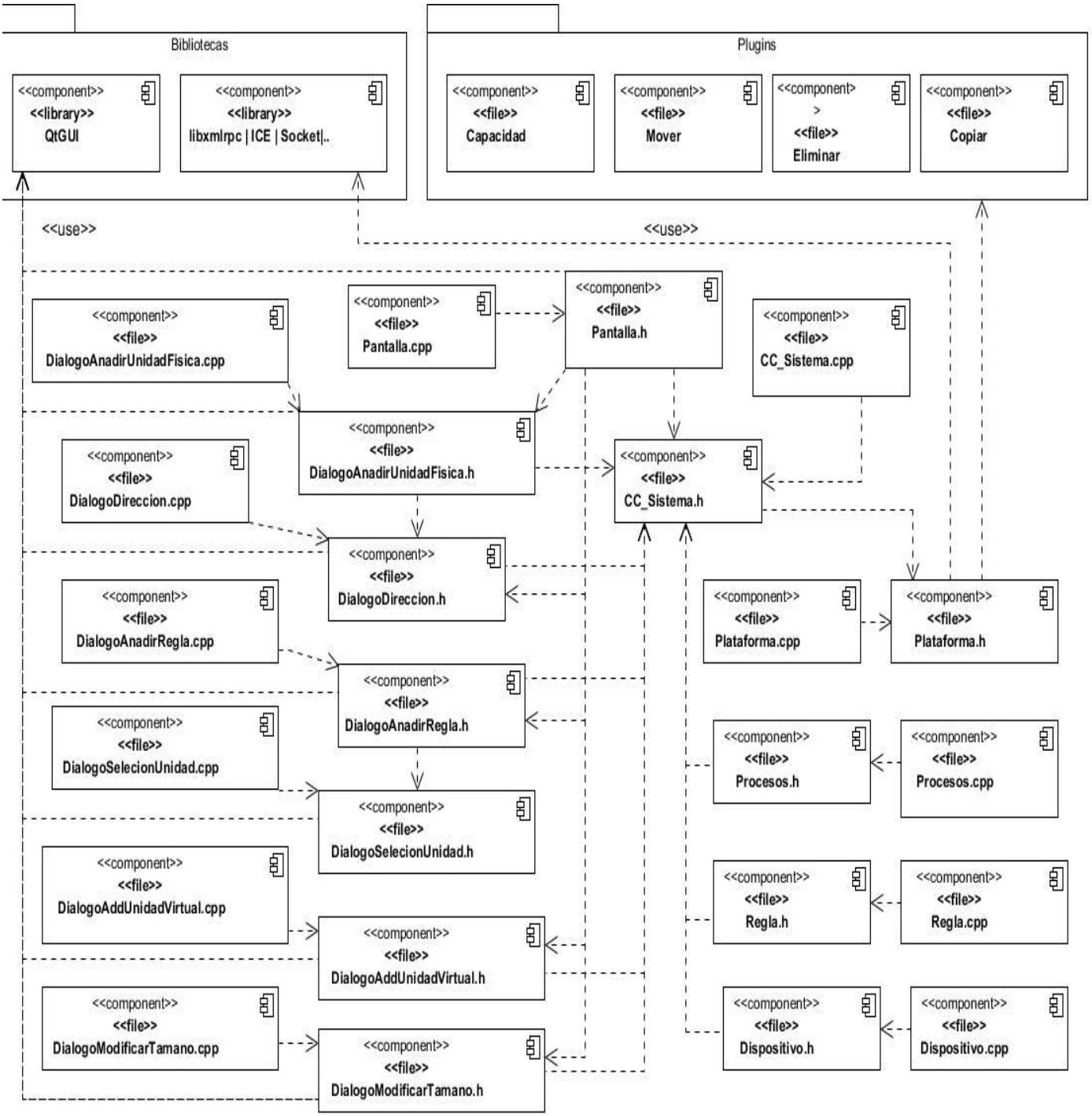


Figura 8. Diagrama de componentes.

3.4 Prueba del módulo.

Según Roger Pressman, existen dos formas de probar cualquier producto construido. Las pruebas de caja blanca se realizan si se tiene conocimiento del funcionamiento interno del producto y mediante las pruebas de caja negra o de comportamiento se centran en los requisitos funcionales del software.

Para realizar las pruebas al módulo se seleccionó la prueba de caja negra. El sistema Suria actualmente se encuentra en proceso de implementación, pues la versión anterior estaba desarrollada en un lenguaje de programación que no respondía a las necesidades del cliente en estos momentos. Por lo que fue indispensable la utilización de una base de datos y una aplicación (Plataforma) auxiliar que facilitaran la ejecución de dichas pruebas.

Dentro de este tipo de prueba se escogió la técnica de partición equivalente, la cual consiste en realizar casos de pruebas (CP) para examinar los valores válidos e inválidos de las entradas existentes en el módulo. Mediante el uso de esta técnica se descubren clases de errores, reduciendo así el número de clases de prueba que hay que desarrollar (Pressman, 2005). Con este propósito fueron diseñados los siguientes casos de pruebas para identificar las deficiencias que surjan y mediante iteraciones ir corrigiendo paulatinamente cada una de las no conformidades que aparezcan.

3.4.1 Listado de los casos de prueba del módulo.

En el presente epígrafe se especifican el CP # 1 Gestionar reglas de administración de almacenamiento y el CP # 2 Administrar medias. Los casos de prueba restantes se pueden encontrar en el Anexo IV del presente trabajo.

Caso de Prueba # 1 CU Gestionar reglas de administración de almacenamiento.

Sección 1 " Insertar regla de administración de almacenamiento**".**

Escenario	Descripción	Respuesta del módulo	Flujo central
EC 1.1 Insertar regla de administración de almacenamiento, correctamente.	El usuario selecciona la opción Crear regla de administración de almacenamiento, el módulo le muestra	El módulo adiciona la regla, deja acciones pendientes a ejecutar, actualiza el listado de las reglas y registra	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/ botón

	una ventana con los campos a seleccionar, el usuario selecciona todos los datos.	esta acción en el fichero de log.	Siguiente/ botón Siguiente/ botón Finalizar
EC 1.2 Insertar regla de administración de almacenamiento Siguiente.	El usuario selecciona la opción Siguiente.	El módulo muestra la siguiente ventana.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/ botón Siguiente
EC 1.3 Insertar regla de administración de almacenamiento Atrás.	El usuario selecciona la opción Atrás.	El módulo muestra la ventana anterior.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/botón Siguiente/ botón Atrás
EC 1.4 Insertar regla de administración de almacenamiento, Cancelar.	El usuario selecciona la opción Cancelar.	El módulo no adiciona la regla, cierra la ventaja y vuelve a la ventana anterior.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/botón Cancelar. Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/botón Siguiente/ botón Cancelar.

EC 1.5 Insertar regla de administración de almacenamiento, Finalizar.	El usuario selecciona la opción Finalizar.	El módulo insertar la regla si todos los datos seleccionados son correctos, deja pendiente las acciones a ejecutar y registra en el fichero de log la acción de insertar regla. Si hay algún campo vació o se solapa una regla, no adiciona la regla y muestra un mensaje avisando el error ocurrido.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/botón Siguiente/ botón Siguiente/ botón Finalizar.
EC 1.6 Insertar regla de administración de almacenamiento, campos sin seleccionar.	El usuario selecciona la opción Crear reglas de administración de almacenamiento, dejando algún campo sin seleccionar.	El módulo muestra mensajes informando a los usuarios el campo o los campos que no han sido seleccionados.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/botón Siguiente. Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/botón Siguiente/ botón Siguiente.
EC 1.7 Insertar regla de administración de	El usuario selecciona la opción Agregar, el	El módulo agrega la unidad seleccionada.	Módulo de autonomía/ pestaña Reglas/ clic en el

almacenamiento, Agregar.	módulo muestra una ventana para que el usuario seleccione la unidad a agregar.		botón Crear Regla de Administración de Almacenamiento/Agregar/ Aceptar.
EC 1.8 Insertar regla de administración de almacenamiento, Limpiar.	El usuario selecciona la opción Limpiar.	El módulo quita todas las unidades insertadas por los usuarios.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/Limpiar.
EC 1.9 Insertar regla de administración de almacenamiento, Eliminar.	El usuario selecciona la una unidad y luego el botón Eliminar.	El módulo elimina la unidad seleccionada.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/Eliminar.
EC 1.10 Insertar regla de administración de almacenamiento, Mover videos.	El usuario selecciona la opción Mover videos y además las unidades origen y destino.	El módulo muestra un formulario con los datos a seleccionar para definir la acción de mover videos.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/botón Siguiente/Mover videos.
EC 1.11 Insertar regla de administración de almacenamiento, Copiar videos.	El usuario selecciona la opción Copiar videos y además las unidades origen y destino.	El módulo muestra un formulario con los datos a seleccionar para definir la acción de copiar videos.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/botón Siguiente/Copiar videos.

EC 1.2 Insertar regla de administración de almacenamiento, Borrar videos.	El usuario selecciona la opción Mover videos y además la unidad destino.	El módulo muestra un formulario con los datos a seleccionar para definir la acción de borrar videos.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/botón Siguiente/Borrar videos.

Tabla 4. Diseño de caso de prueba Sección "Insertar regla de administración de almacenamiento".

Sección 2 "Eliminar regla de administración de almacenamiento**".**

Escenario	Descripción	Respuesta del módulo	Flujo central
EC 1.1 Eliminar regla de administración de almacenamiento correctamente.	El usuario selecciona la opción Eliminar regla de administración de almacenamiento, el módulo le muestra un mensaje de confirmación y el usuario acepta la confirmación.	El módulo muestra el mensaje “Desea realmente eliminar la regla de administración seleccionada”, el usuario selecciona el botón Yes, el módulo elimina la regla, actualiza el listado de las reglas y registra esta acción en el fichero de log.	Módulo de autonomía/ pestaña Reglas/ clic en la regla a eliminar/clic en el botón Eliminar Regla de Administración de Almacenamiento/Yes. Módulo de autonomía/ pestaña Reglas/ clic derecho en la regla a eliminar/seleccionar Eliminar Regla de Administración de Almacenamiento/Yes.
EC 1.2 Eliminar regla de administración de almacenamiento No.	El usuario selecciona la opción Eliminar regla de administración de almacenamiento, el módulo le muestra un mensaje de	El módulo muestra el mensaje “Desea realmente eliminar la regla de administración seleccionada”, el usuario selecciona el botón No, el módulo no	Módulo de autonomía/ pestaña Reglas/ clic en la regla a eliminar/clic en el botón Eliminar Regla de Administración de Almacenamiento/No. Módulo de autonomía/

	confirmación y el usuario no acepta la confirmación.	elimina la regla y cierra la ventana del mensaje.	pestaña Reglas/ clic derecho en la regla a eliminar/seleccionar Eliminar Regla de Administración de Almacenamiento/No.

Tabla 5. Diseño de caso de prueba Sección "Eliminar regla de administración de almacenamiento".

Caso de Prueba # 2 CU Administrar medias.

Sección 1 "Copiar medias".

Escenario	Descripción	Respuesta del módulo	Flujo central
EC 1.1 Copiar medias, correctamente.	El módulo manda a copiar las medias a la unidad seleccionada en las reglas programadas.	El módulo copiar las medias, actualiza el estado de almacenamiento de la unidad destino y registra en el fichero de log esta acción.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/ botón Siguiente/ botón Siguiente/ botón Finalizar/ acción Copiar medias

Tabla 6. Diseño de caso de prueba Sección “Copiar medias”.

Sección 2 "Mover medias".

Escenario	Descripción	Respuesta del módulo	Flujo central
EC 1.1 Mover medias, correctamente.	El módulo manda a mover las medias a la unidad seleccionada en las reglas	El módulo mueve las medias, actualiza el estado de almacenamiento de la	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de

	programadas.	unidad origen y destino y registra en el fichero de log esta acción.	Almacenamiento/ botón Siguiente/ botón Siguiente/ botón Finalizar/ acción Mover medias.

Tabla 7. Diseño de caso de prueba Sección “Mover medias”.

Sección 3 "Eliminar medias".

Escenario	Descripción	Respuesta del módulo	Flujo central
EC 1.1 Eliminar medias, correctamente.	El módulo manda a eliminar las medias de la unidad seleccionada en las reglas programadas.	El módulo elimina las medias, actualiza el estado de almacenamiento de la unidad y registra en el fichero de log esta acción.	Módulo de autonomía/ pestaña Reglas/ clic en el botón Crear Regla de Administración de Almacenamiento/ botón Siguiente/ botón Siguiente/ botón Finalizar/ acción Eliminar medias.

Tabla 8. Diseño de caso de prueba Sección “Eliminar medias”.

3.4.2 Resultados de las pruebas.

Una vez diseñados los casos de pruebas se realizó una primera iteración, clasificando las no conformidades de alta, media y baja. Obteniéndose 7 no conformidades, 2 de criticidad alta, 1 de criticidad media y 4 de baja. En la siguiente tabla se encuentran reflejadas las no conformidades identificadas:

No	No conformidad	Clasificación
1	Se insertó una regla correctamente y se solapaba con una ya insertada.	Alta.
2	Se insertó una unidad virtual que ya existía.	Alta.
3	Al eliminar una regla no se muestra el mensaje de confirmación.	Media.

4	En el mensaje de “Fechas incorrectas”, el texto identificativo comienza con minúscula.	Baja.
5	En el mensaje de “Eliminar una unidad física” aparecen espacios innecesarios entre las palabras.	Baja.
6	En los mensajes de confirmación de las preguntas solo aparece el signo de interrogación al final.	Baja.
7	Cuando se quiere insertar una unidad física y en el módulo no hay ninguna, aparece un mensaje que contiene más de un punto al final.	Baja.

Tabla 9. Resultado de la 1ra iteración.

Luego de terminada la primera iteración, se realizó una segunda iteración, identificando una no conformidad de criticidad baja relacionada con la internacionalización de los textos. Esta no conformidad se muestra en la siguiente tabla:

No	No conformidad	Clasificación
1	Los mensajes de confirmación aparecen en español y el texto identificativo del botón de aceptación aparece en inglés.	Baja.

Tabla 10. Resultado de la 2ra iteración.

Se realizó una tercera iteración no encontrándose ninguna no conformidad, demostrando que la solución desarrollada cumple con las necesidades solicitadas por el proyecto Video Vigilancia, garantizándole un módulo autónomo que garantiza el control de la saturación de los dispositivos de almacenamiento en el servidor de media del sistema. La siguiente figura muestra una gráfica con una representación del comportamiento de las iteraciones desarrolladas.

Figura 9. Comportamiento de las no conformidades.

3.5 Conclusiones parciales.

- ✓ Se describieron los elementos arquitectónicos que fueron definidos a nivel de proyecto y que eran necesarios para lograr el desarrollo de la solución, permitiendo una mejor organización en la estructura del módulo desarrollado.
- ✓ Se confeccionaron los diagramas de clase, de despliegue y de componente, plasmando los nodos y relaciones necesarias para una posterior integración con el sistema Suria.
- ✓ Se realizaron varias iteraciones de prueba de caja negra, específicamente la técnica de partición equivalente, permitiendo corregir las no conformidades identificadas, quedando el módulo con un mayor grado de perfección para satisfacer las peticiones de los clientes.

Conclusiones generales.

Con la realización del presente trabajo, se cumplieron los objetivos propuestos al comienzo de la investigación, pudiendo arribar a las siguientes conclusiones:

- ✓ Se demostró que con el desarrollo del módulo de autonomía para el control del espacio de almacenamiento en los servidores de medias del sistema Suria, se garantiza la gestión de la saturación del espacio en los dispositivos de almacenamiento en el servidor de media.

- ✓ Se realizó un estudio del arte de las soluciones similares existentes en el mundo, demostrando la necesidad de implementar un módulo autónomo para el control de la saturación del espacio de almacenamiento en el sistema Suria.

- ✓ Se obtuvo durante la implementación del módulo las funcionalidades de copiar, mover, eliminar y capacidad, que posibilitarán su utilización desde otras soluciones, pudiendo tener vida independiente y contribuyendo a la reutilización de código entre los diferentes subsistemas, módulos y/o componentes que han sido y serán implementados para el sistema Suria.

- ✓ Se comprobó el cumplimiento de los atributos de calidad definidos, así como los requisitos funcionales a partir de la realización de varias iteraciones de pruebas que permitieron corregir las no conformidades identificadas, quedando el módulo con un mayor grado de perfección para satisfacer las necesidades del sistema Suria.

Recomendaciones.

Los objetivos de la presente investigación han sido desarrollados satisfactoriamente, dándole cumplimiento a todos los requisitos trazados. Aunque se recomienda para investigaciones venideras que estén relacionadas, enriquecer el sistema en cuanto a:

- ✓ Integrar el módulo obtenido en la implementación de la presente investigación con el sistema Suria, con el fin de mejorar la administración de los recursos de almacenamiento y evitar la pérdida de las medias.
- ✓ Implementar una nueva funcionalidad que permita el almacenamiento distribuido entre varios servidores, logrando que el módulo sea más eficiente a la hora de manejar las medias generadas por el sistema Suria.
- ✓ Incorporar la funcionalidad que permita cancelar una transferencia de medias en proceso.

Referencia Bibliográfica.

ALEGSA. 2009. *Diccionario Informático.* [En línea] 2009. [Citado el: 5 de marzo de 2013.] http://www.alegsa.com.ar/Dic/modulo.php.

Álvarez, Alejandro Figueras. 2011. *Tesis Sistema para el control autónomo de espacio en disco en servidores de media.* La Habana, 2011.

Baby, Ronald y Almeida, Enrique. 2009. *Tesis Sistema de Catalogación de Medias.* La Habana, 2009.

Booch, Grady. 2011. El Clud del Programador. *Introducción a los Patrones de Diseño.* [En línea] 29 de octubre de 2011. [Citado el: 5 de marzo de 2013.] http://www.elclubdelprogramador.com/.../patrones-de-diseno-introduccion.

Brokken, Frank B. 2012. *C++ Annotations Version 9.7.2.* ***Center of Information Technology,*** University of Groningen, 2012. . [Citado el: 5 de marzo de 2013.] http://www.icce.rug.nl/documents/cplusplus/

Castañeda, Gerardo Ojeda. 1992. LOS ARCHIVOS AUDIOVISUALES EN LAS REDES DIGITALES. *INFORME DE INVESTIGACIÓN Y DOCUMENTACIÓN ANALÍTICA.* [En línea] 1992. [Citado el: 5 de noviembre de 2012.] http://unesdoc.unesco.org/images/0012/001256/125637s.pdf. 84-369-4135-7.

Dell. 1999. Dell Storage Management. *Administración de recursos de almacenamiento.* [En línea] 1999. [Citado el: 25 de noviembre de 2012.] http://www1.la.dell.com/content/topics/global.aspx/sitelets/solutions/management/storage_resource_mgmt?c=bo&l=es&cs=bobiz1.

DATYS. 2010. *XYMA SAVE VISION.* SISTEMA DE VIDEO VIGILANCIA IP. [En línea] 2010. [Citado el: 7 de diciembre de 2012.] http://www.datys.cu/descargas/XYMA_Safe_Vision_V2.7_WP_12.04.10_E01__Es_D.pdf

EcuRed. 2012. IDE de Programación. [En línea] 24 de septiembre de 2012. [Citado el: 18 de febrero de 2013.] http://www.ecured.cu/index.php/IDE_de_Programaci%C3%B3n

ESPAÑOLA, REA ACADEMIA. 2011. REA ACADEMIA ESPAÑOLA. *DICCIONARIO DE LA LENGUA ESPAÑOLA - Vigésima segunda edición.* [En línea] 2011. [Citado el: 5 de noviembre de 2012.] http://buscon.rae.es/draeI/SrvltConsulta?LEMA=transferir.

ESPAÑOLA, REAL ACADEMIA. 2011. ESPAÑOLA, REAL ACADEMIA. *DICCIONARIO DE LA LENGUA ESPAÑOLA - Vigésima segunda edición.* [En línea] 2011. [Citado el: 29 de octubre de 2012.] http://buscon.rae.es/draeI/SrvltConsulta?LEMA=proceso.

Fontanillo, Laura. 2005. PROYECTO ECLIPSE. *Programación Orientada a Objetos*. [En línea] Universidad de Salamanca, mayo de 2005. [Citado el: 5 de marzo de 2013.] http://zarza.usal.es/~fgarcia/docencia/poo/04-05/Trabajos/Eclipse.pdf

Guerrero, Inés Friss de Kereki. 2003.*Tesis Doctoral Modelo para la Creación de Entornos de Aprendizaje basado en técnicas de Gestión del Conocimiento.* Madrid : Facultad de Informática de la Universidad Politécnica de Madrid, 2003.

Kuppler, Gerhard. 2005. Oracle para SAP es segura, confiable y escalable. *Oracle para SAP es segura, confiable y escalable.* [En línea] abril de 2005. [Citado el: 2 de diciembre de 2012.] http://www.oracle.com/us/solutions/volume14-sp-183633.pdf. IBM.

Medina, Juan Carlos. 2004. *Análisis Comparativo de Técnicas, Metodologías y Herramientas de Ingeniería de Requerimientos.* D.F. México. En línea] 2004. [Citado el: 2 de marzo de 2013.] http://www.cs.cinvestav.mx/tesisgraduados/2004/resumenJuanCarlosM.html

Nokia Corporation. 2008. QT. [En línea] 2008. [Citado el: 06 de 02 de 2012.] http://qt.nokia.com/products/developer-tools/.

NORTHERN. 2009. NORTHERN. *NORTHERN_una_solucion_de_administracion_de_recursos_de_almacenamiento.* [En línea] 2009. [Citado el: 2 de diciembre de 2012.] http://www.powershow.com/view/2822e8-MjI5M/NORTHERN_una_solucion_de_administracion_de_recursos_de_almacenamiento

Osmosislatina. 2007. Osmosislatina.com. *Guía de UML: Importancia de UML.* [En línea] 31 de 12 de 2007. [Citado el: 28 de Enero de 2013.] http://www.osmosislatina.com/lenguajes/uml/basico.htm.

Pressman, Roger S.. 2005. *Ingeniería de Software. Un enfoque práctico.* 2005. 5ta Edición.

Phylum. 2012. Phylum. *Mejores Prácticas - RUP.* [En línea] 2012. http://phylum.com.mx/es/soluciones/systems-integration-a-technology/67-metodologias.html?start=5.

Reyna, Rafaela. 2010. Ingeniería de Software I. *Herramientas Case.* [En línea] 2010. [Citado el: 25 de enero de 2013.] http://www.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE.

Reynoso, Carlos. 2004. Estilos y Patrones en la Estrategia de Arquitectura de Microsoft. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* [En línea] UNIVERSIDAD DE BUENOS AIRES, Marzo de 2004. [Citado el: 5 de marzo de 2013.] http://pis.unicauca.edu.co/moodle/file.php/312/.../Estilos_y_patrones.doc

Restrepo, Guillermo González. 2011. Facultad de Ingeniería - Universidad de Antioquia. *El Concepto y Alcance de la Gestión Tecnológica.* [En línea] 2011. [Citado el: 12 de noviembre de 2012.] http://jaibana.udea.edu.co/producciones/guillermo_r/concepto.html

Soto, Lauro. 2010. Lenguaje de programacion. *Tecnológico.* [En línea] 2010. [Citado el: 11 de enero de 2013.] http://www.mitecnologico.com/Main/DefinicionDeLenguajeDeProgramacion

Symantec Corporation. 1995-2012. Software de almacenamiento. Administración de recursos de almacenamiento | Symantec CommandCentral Storage. *Veritas CommandCentral StoragE.* [En línea] 1995-2012. [Citado el: 7 de NOVIEMBRE de 2012.] http://www.symantec.com/es/mx/commandcentral-storage

The Free, Dictionary. 2012. The Free Dictionary. *autonomo - significado de autonomo diccionario.* [En línea] 2012. [Citado el: 5 de noviembre de 2012.] http://es.thefreedictionary.com/aut%C3%B3nomo

Visual Paradigm. 2011. UML tool for software application development. *UML tool for software application development.* [En línea] 2011. [Citado el: 20 de enero de 2013.] http://www.visual-paradigm.com/product/vpuml/

XMLRPC. 1998. *XMLRPC.* [En línea] 14 de 6 de 1998. [Citado el: 9 de enero de 2013.] http://xmlrpc.scripting.com/

Zelaya, Hazel Edith Ramírez. 2010. Ingeniería de Software 1. *Arquitectura Cliente/Servidor .* [En línea] Universidad de Politécnica de Nicaragua, 16 de noviembre de 2010. [Citado el: 5 de diciembre de 2012.] www.slideshare.net/.../arquitectura-cliente-servidor. ISBN.

Bibliografía.

Chacón, Julio César Rueda. 2006. *Aplicación de la metodología RUP para el desarrollo rápido de aplicaciones basado en el estándar J2EE.* Guatemala: Universidad de San Carlos de Guatemala, 2006. [En línea] 2006. [Citado el: 20 de febrero de 2013.] http://biblioteca.usac.edu.gt/tesis/08/08_0308_CS.pdf

Cubaminrex. 2004. Cuba en la Cumbre Mundial sobre la Sociedad de la Informatización. *La informatización en Cuba.* [En línea] Ministerio de Relaciones Exteriores de la República de Cuba, 2004. [Citado el: 8 de diciembre de 2012.]

Definiciones. 2007. Definiciones. *Definiciones.com.* [En línea] 2007. [Citado el: 28 de octubre de 2012.] http://www.definiciones.com.mx/definicion/C/codificar/

EcuRed. 2010. Historia de la Informática en Cuba . *Historia de la Informática en Cuba .* [En línea] 14 de diciembre de 2010. [Citado el: 8 de dicienbre de 2012.] http://www.ecured.cu/index.php/Historia_de_la_Inform%C3%A1tica_en_Cuba.

García Hernández, Alién y Concepción Hidalgo, Dolennis. *Concurso sobre Historia de la Informática.*

Graells, Dr. Pere Marquès. 1995. INTRODUCCIÓN A LA INFORMÁTICA. *INTRODUCCIÓN A LA INFORMÁTICA.* [En línea] 1995. [Citado el: 30 de octubre de 2012.] http://www.peremarques.net/INFMULTI.htm. HTM

Harmonic Inc. 2011. Harmonic ProMediaCarbon. [En línea] 2011. [Citado el: 24 de octubre de 2012.] http://www.carboncoder.com/promedia carbon.html

Hernández, Enrique Orallo. 2012. *El Lenguage Unificado de Modelado (UML).* [Citado el: 20 de febrero de 2013.] http://www.disca.upv.es/enheror/pdf/ActaUML.PDF

Hernando, Roberto. 2001.*Metodologías de desarrollo de software***.** [En línea] 2001. [Citado el: 28 de Enero de 2012.] http://www.rhernando.net/modules/tutorials/doc/ing/met_soft.html.

Larman, Craig. 1999. UML y Patrones*. Introducción al análisis y diseño orientado a objetos.* Naucalpan de Juárez, Estado de México: Dawn Speth White, 1999. 970-17-0261-1.

Martinez, Indira Bravo. 2010. Informática. *¿Qué es un sistema de gestión de base de datos (SGBD)?* [En línea] 2010. [Citado el: 9 de enero de 2013.] http://indira-informatica.blogspot.com/.

Mejía, Jesús Hernando Maya. 2009. Informática. Qué es. Sistemas de información. *Capítulo Informática. Qué es. Sistemas de información del curso Comunicación e informática. Historia y computación.* [En línea] 8 de septiembre de 2009. [Citado el: 15 de noviembre de 2012.] http://www.emagister.com/curso-comunicacion-informatica-historia-computacion/informatica-que-es-sistemas-informacion

Muro, Pedro. 2010. ARP CALIDAD. *Definición de proceso.* [En línea] 2010 de mayo de 2010. [Citado el: 10 de noviembre de 2012.] http://arpcalidad.com/definicin-de-proceso/

PostgreSQL Global Development Group.2012. PostgreSQL. *PostgreSQL: Documentation: Manuals: PostgreSQL 8.3: What is PostgreSQL?* [En línea] [Citado el: 22 de enero de 2013.] http://www.postgresql.org/docs/8.3/interactive/preface.html

Rasabal, Heriberto. 2011. Cuba en Noticias. *Presentan Industria Cubana del Software.* [En línea] 22 de diciembre de 2011. [Citado el: 8 de diciembre de 2012.] Presentan Industria Cubana del Software .

Sánchez, Bernardo. 2011. Servidor. *Servidor.* [En línea] 2011. [Citado el: 4 de diciembre de 2012.] http://besanchez.files.wordpress.com/2011/02/3-servidores1. ISSN.

Semanat, Edmis Deivis Aldana. 2009. Sistema de Video Vigilancia. *Sistema de Video Vigilancia.* La Habana 2009.

TEDIAL, Tecnologías Digitales Audiovisuales. 2009. TEDIAL Tecnologías Digitales Audiovisuales, S.L. *TEDIAL Tecnologías Digitales Audiovisuales, S.L.* [En línea] BPM, now a reality. Málaga, España, 2009. [Citado el: 2 de noviembre de 2012.] http://www.tvbeurope.com/c/document_library/get_file?p_l_id=32408&folderId=153861&name=DLFE-1703

Tesiorowska, Aleksandra. 2009. Linux +. *Boots.* [En línea] 2009. [Citado el: 10 de enero de 2013.] http://www.fedora-es.com/manuales/Linux+/Linux_02_2009_ES.pdf. 1732-7121.

Troya, José M. 2000. *Desarrollo de Software basado un componente.* Ciudad Madrid, España : Ediciones de la Universidad de Castilla-La Mancha, 2000. 84-8427-077-7.

Vallespir, Diego. 2011. Facultad de Ingeniería. Universidad de la República - Uruguay. [En línea] 2011. http://www.fing.edu.uy/inco/cursos/ingsoft/pis/proceso/MUM/dat/intro/intro.htm

Vega, Juan Carlos. 1986. Introducción a la Informática. *Introducción a la Informática (Primera Edición).* [En línea] 1986. [Citado el: 1 de noviembre de 2012.] http://www.alibri.es/tecnica/informatica/introduccion-a-la-informatica

WEBNOVA. 2011. WEBNOVA. *Web Services - XML-RPC, SOAP, sobre PHP, Perl, y otros conceptos.* [En línea] 2011. [Citado el: 2 de noviembre de 2012.] http://webnova.com.ar/articulo.php?recurso=426

Glosario de términos.

API: (*Application Programming Interface*) Interfaz de Programación de Aplicaciones.

Autónomo: Que goza de autonomía o independencia, que toma solo las decisiones de qué acción debe realizarse ante determinada situación.

CASE: (*Computer Aided Software Engineering*) Ingeniería de Software Asistida por Computación.

CD-ROM: (*Compact Disc - Read Only Memory*) Disco Compacto de Memoria de Sólo Lectura.

Circuito cerrado: es aquel que permite la circulación de la corriente (interruptor cerrado) y de esa forma hacer funcionar la carga.

Desencadenador: es la acción que hace que se comience a ejecutar una regla.

DVD: (*Digital Versatile Disc*) Disco Digital Versátil es un dispositivo de almacenamiento óptico.

IDE: (*Integrated Develop Environment*) Entorno Integrado de Desarrollo.

IP: (*Internet Protocol*) Protocolo de Internet.

Medias: cualquier tipo de objeto que usa de forma simultánea varios tipos de contenidos, como son: video, audio, texto, imágenes, animación, entre otros.

Módulo: es una porción de un programa de computadora.

Plugin: es la unidad mínima de funcionalidad que puede ser distribuida de manera separada.

RAM: (*Random Access* Memory*)* Memoria de Acceso Aleatorio es una memoria de semiconductores en la que se puede escribir o leer información.

RUP: (*Rational Unified Process*) Proceso Unificado Racional.

TICs: Tecnologías de la Información y las Comunicaciones.

Qt: es una biblioteca multiplataforma ampliamente usada para desarrollar aplicaciones con interfaz gráfica de usuario.

XML: (*Extensible Markup Language*) Lenguaje de Marcas Extensibles.

XML-RPC: (*XML Remote Producer Call*) Protocolo de Llamada Remota.

UML: (*Unified Modeling Language*) Lenguaje Unificado de Modelado.

USB: (*Universal Serial Bus*) Bus Universal en Serie es el puerto que permite conectar periféricos a una computadora.