UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 6

Título: Desarrollo del Módulo de Gestión de Errores para el subsistema de transmisión de PRIMICIA.

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

Autor: Alejandro Blanco Peña

Tutor: Ing**.** Rayner Pupo Gómez

La Habana, Junio de 2012.

"Año 54 de la Revolución"

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

Albert Einstein

DECLARACIÓN DE AUDITORÍA

Declaro que soy el único autor de la presente tesis y autorizo al departamento de Señales Digitales del centro de Geoinformática y Señales Digitales (GEySED) de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2012.

Alejandro Blanco Peña	Ing. Rayner Pupo Gómez
_____	_____
Firma del Autor	Firma del Tutor

DATOS DE CONTACTO

Tutor: Ing. Rayner Pupo Gómez

Categoría docente: Adiestrado.

Centro de trabajo: Universidad de las Ciencias Informáticas.

Título de la especialidad de graduado: Ingeniero en Ciencias Informáticas.

Año de graduación: 2011

Institución en la que se graduó: Universidad de las Ciencias Informáticas.

Correo electrónico: rpgomez@uci.cu

AGRADECIMIENTOS

A mi mamá y padrastro por apoyarme incondicionalmente en cada paso de mi vida y por hacerme la persona que hoy soy.

A mi hermano Leo por quererme y por estar orgulloso de mí.

A mi novia Yamila le estoy eternamente agradecido por ayudarme en cada momento que lo necesité, por amarme, apoyarme y darme fuerzas para seguir adelante. Te amo mucho.

A la familia de mi novia por aconsejarme y por preocuparse por mis resultados en el transcurso del desarrollo del presente trabajo de diploma.

A mi tutor por ayudarme y atenderme cada vez que surgía una duda.

A mi oponente y tribunal de tesis, por cada sugerencia para ser un mejor ingeniero, a ustedes también muchas gracias.

A los profesores que han estado en el proyecto y a los que continúan en él, especialmente a Enrique, Lisandra y Geovannys.

A mi tribunal de tesis en Venezuela Rosaida, Jorge y Dysan y al tribunal del curso pasado.

A mis amistades Carlos Simón, Luiso, Hendrik, Alexi, Fili, Oscar, Dysan, Ana Lizandra y Lourdes por estar pendientes de mis resultados y también por ayudarme y apoyarme en etapas de mi vida.

A Fidel y a la Revolución por brindarme la oportunidad de poder formarme como un profesional.

Y en general a toda aquella persona que aportó un poquito para que fuera ingeniero, a todos ustedes muchas gracias.

DEDICATORIA

Dedico este trabajo de diploma a:

A mi madre Florinda por ser, sin duda alguna, la mejor del mundo.

A mi padrastro Pedro por ser como un padre para mí.

A mi hermano Leo.

A mi amor Yamila Tamayo.

A mi tío Jorge por ayudarme cuando lo necesité.

A mis suegros Mongui y Cleiris.

A mi cuñada Yaimé y su esposo Alexander.

A todos mis amigos por compartir lo mejor de sí mismos y por aceptarme en sus vidas.

RESUMEN

El departamento de Señales Digitales perteneciente al centro de desarrollo GEySED de la facultad 6 dedica sus esfuerzos al desarrollo de varias aplicaciones informáticas que tienen como objetivo ampliar la utilización de los medios de comunicación en Cuba y el mundo. Durante varios años se ha trabajado en el perfeccionamiento de la Plataforma de Televisión Informativa PRIMICIA para que esta sea totalmente configurable y pueda adaptarse a las necesidades de los usuarios. Actualmente en el departamento de Señales Digitales, específicamente en el proyecto productivo PRIMICIA no existe una vía de identificar posibles errores que pudieran suceder en la transmisión del canal informativo. La presente investigación tiene como objetivo fundamental la implementación de un módulo que permita detectar errores que ocurren durante la utilización de la plataforma. Mediante el desarrollo de este sistema se podrán detectar los fallos ocurridos y se conocerá cómo darle solución y seguimiento a los mismos. Para el desarrollo de la aplicación se utilizará el lenguaje de programación C++, apoyándose en el framework QT y como metodología de desarrollo RUP. Esto aporta al subsistema de transmisión un complemento que facilitaría la detección de errores a menor y gran escala.

Palabras claves
Detectar, errores, módulo, transmisión.

ABSTRACT

The department of Digital Signals that belongs to the GEySED development center of school 6 dedicates its efforts to the development of various software applications that aim to expand the use of communications media on Cuba and the world. For several years the department worked on perfecting the Informative TV Platform PRIMICIA, for it to be totally configurable and can adapt to the needs of the users. Nowadays in the department of Digital Signals, specifically the productive project PRIMICIA there is no way to identify possible errors that may occur in the transmission of the informative channel. This research has as main objective the implementation of a module that allows to detect errors that occur while using the platform. Through the development of this system it will be possible to detect faults and know how to follow up and solve them. For the development of the application we will use C++ programming language, QT framework and RUP as development methodology. This brings to the transmission subsystem a complement to facilitate error detection in small and large scales.

Keywords

Detect, bugs, module, transmission.

Índice de contenido

Índice de tablas

Índice de figuras

INTRODUCCIÓN

El ser humano ha buscado durante siglos una manera de satisfacer su necesidad de comunicación. Hoy día las Tecnologías de la Información y las Comunicaciones (TIC), se han expandido aceleradamente llegando a todos los aspectos de la vida del hombre. Los medios de comunicación constituyen una herramienta eficaz para mantener a las personas informadas sobre la situación social, política y económica que vive el mundo. La televisión ha sido uno de los factores con mayor influencia en el desarrollo de la cultura actual y visión del mundo. Se ha convertido en un medio de comunicación que influye en la vida de las personas, y tener el control de esta es una aspiración para muchos.

Con la necesidad de expandir las comunicaciones en Cuba se han desarrollado plataformas de televisión informativa y productos aislados en la Universidad de las Ciencias Informáticas (UCI), creada en el año 2002. La facultad 9 fue la primera en incursionar en este campo con la creación del canal informativo Señal 3 en el año 2005, con el objetivo de llevar a la inmensa comunidad universitaria un canal de noticias que los mantuviera informados de forma dirigida y concreta. Luego en el 2006 surgió el canal informativo ACN de la Agencia Nacional de Noticias, una solución similar a la anterior, implementada para hacer llegar noticias y otras informaciones de Cuba y el mundo reflejadas en la prensa plana a los colaboradores internacionalistas que prestan sus servicios en una gran cantidad de países y a los habitantes de lugares intrincados del país, llamados zonas de silencio. Más tarde, con la creación del Polo Video y Sonido Digital en esta facultad surgió la personalización TVEnergía, a solicitud del Ministerio de Energía y Petróleo de Venezuela, a la que hubo que adaptarle nuevas funcionalidades acordes a las necesidades del cliente pues querían que estuviera desarrollada con herramientas libres. Finalmente se crea la plataforma PRIMICIA, un producto en construcción adaptable a las necesidades de cualquier cliente y que hereda las mejores características de las soluciones desarrolladas anteriormente. Actualmente el proyecto PRIMICIA pertenece al Centro de desarrollo GEySED de la facultad 6.

La Plataforma de Televisión Informativa PRIMICIA provee un canal de teletexto a una red de televisión. PRIMICIA tiene como objetivo la transmisión de noticias realizando una labor informativa y formativa. Está pensada para realizar la transmisión de informaciones de manera fácil y rápida, se plantea que puede ser utilizado en aeropuertos para brindar información sobre los vuelos; en terminales de trenes o de ómnibus para mostrar la información relacionada con los viajes; en hoteles para visualizar noticias de relevancia nacional, internacional o publicidad turística; en empresas con la necesidad de brindar informaciones de último momento a sus empleados y clientes, en fin, en las instituciones que necesiten brindar cualquiera de estos servicios.

El producto PRIMICIA desde su origen estuvo compuesto por dos subsistemas: el de administración donde se configura el canal y se gestionan las noticias y recursos multimedia y el de transmisión que se encarga de visualizar las noticias y materiales publicados.

Los productos que se realizaron antes de la Plataforma de Televisión Informativa PRIMICIA poseían errores en la transmisión no siempre debido a fallas internas, sino también externas, como la caída de la base de datos o del servidor de medias ocasionando el mal funcionamiento del canal o su interrupción. Las primeras versiones del subsistema de transmisión estaban desarrolladas para la web y tenían un mecanismo de detección de errores que pasó a ser obsoleto cuando se decidió pasar la aplicación a su segunda versión, usando para transmitir una aplicación de escritorio. PRIMICIA es una versión mejorada de estas versiones aisladas, que también trae consigo errores similares en la transmisión del canal.

Son varios los problemas asociados a la interrupción de la transmisión del canal informativo, puede ser por fallos en los servidores, o la no disponibilidad de los orígenes de datos a los cuales necesita acceder la aplicación, dando como resultado, que por ejemplo el fondo musical no se reproduzca o que algún video no se visualice. Por estos motivos los clientes pueden mostrar cierto desinterés hacia el producto, pues no cuentan con una forma de controlar los fallos que puedan ocurrir durante su utilización. Hay errores que son visibles al observar el canal; por ejemplo cuando un video o el fondo musical no se reproducen o no se ven las imágenes. Hoy día los clientes dependen de los desarrolladores del producto para solucionar estos errores puesto que son los únicos que conocen cómo darle seguimiento y solución. Dichos acontecimientos resultan incómodos para ambas partes e impide que más clientes se interesen en adquirir este producto.

Partiendo de la situación anterior se define como **problema de la investigación**: ¿Cómo garantizar la detección y registro de errores en el subsistema de transmisión de la Plataforma de Televisión Informativa PRIMICIA?

El **objeto de estudio** de esta investigación son los procedimientos para la detección y registro de errores y el **campo de acción** es la automatización de los procesos de detección y registro de los errores.

El **objetivo general** es desarrollar un módulo para el subsistema de transmisión de la Plataforma de Televisión Informativa PRIMICIA que garantice la detección y registro de errores.

A partir del objetivo general se obtiene la siguiente **idea a defender***:* Con el desarrollo de un módulo que permita la detección y registro de los errores en PRIMICIA se logrará conocer el origen y cómo darle solución a los posibles fallos producidos durante la utilización de la plataforma de televisión.

Para dar cumplimiento al objetivo general se trazaron las siguientes **tareas de la investigación**:

1. Caracterización de los posibles errores que puedan ocurrir en el subsistema de transmisión de la Plataforma de Televisión Informativa PRIMICIA.
2. Caracterización de soluciones existentes que puedan contribuir al desarrollo del Módulo de Gestión de Errores.
3. Identificación de los requisitos funcionales y no funcionales vinculados a los procesos de detección y registro de errores en el Módulo de Gestión de Errores.
4. Definición de la estrategia para la detección y registro en caso de errores en el subsistema de transmisión.
5. Argumentación de las tecnologías a utilizar.
6. Implementación de las funcionalidades identificadas.
7. Validación del módulo.

Para el desarrollo de la investigación se utilizaron métodos científicos que permitieron identificar las características del objeto de estudio con el propósito de conocer su esencia, entre ellos están los métodos teóricos y empíricos.

El método **analítico – sintético** se utilizó para analizar el procedimiento de detección de los errores de PRIMICIA, lo cual permitió sintetizar los elementos más importantes de dicho procedimiento.

El **inductivo – deductivo** se empleó para analizar sobre los errores que se producen durante la utilización del canal logrando hacer una generalización de su comportamiento, esto constituyó el punto de partida para definir o confirmar formulaciones teóricas. Además sirvió para comparar soluciones informáticas con características similares a PRIMICIA, analizar cómo se detectaron los errores producidos y deducir la mejor solución para resolver el problema de la investigación.

Los métodos empíricos dieron la posibilidad de extraer de los elementos relacionados con el objeto de estudio la información que se necesitaba de ellos, en el caso de la presente investigación se utilizó la observación.

La **observación** se utilizó durante toda la etapa de identificación y análisis de recopilación de datos e información acerca de los errores que ocurren en el canal y que impiden la transmisión del mismo.

Capítulo 1: Fundamentación teórica del sistema, en este capítulo se abordan aspectos relacionados con el proceso de gestión de errores. Se hace un estudio de los conceptos asociados al problema y al objetivo general de la investigación. Incluye además un análisis de las soluciones existentes a nivel internacional que puedan aportar elementos para el desarrollo del módulo. Se fundamentan las tecnologías y herramientas que se utilizan para el desarrollo del mismo.

Capítulo 2: Características del sistema, en este capítulo se describen los principales conceptos asociados al dominio del problema y se refleja la relación existente entre ellos mediante la elaboración del modelo de dominio. También se especifican los requerimientos funcionales y no funcionales que el sistema debe cumplir. Se definen los actores y casos de uso del sistema así como las relaciones entre ellos mediante el diagrama de casos de uso del sistema y las descripciones generales de cada caso de uso.

Capítulo 3: Análisis y Diseño del sistema, en este capítulo se describe la arquitectura del sistema, utilizándose el patrón Arquitectura en Capas y los patrones de diseño. Además se realiza el diseño del sistema creándose los diagramas de clases como primera aproximación al modelo de implementación.

Capítulo 4: Implementación y Prueba del sistema, este capítulo contiene las principales características de la implementación del sistema. Se muestran los principales componentes y sus relaciones mediante el diagrama de componentes. Además se refleja la validación del sistema mediante la realización de pruebas verificando que los requisitos funcionales fueron cumplidos y asegurando la calidad del mismo.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

1.1 Introducción

En este capítulo se abordan varios temas que conforman la fundamentación teórica del desarrollo del Módulo de Gestión de Errores. De manera general está centrado en la definición de los conceptos asociados al dominio del problema, la fundamentación del objeto de estudio, se analizan algunas soluciones existentes a nivel internacional que se enmarcan en el campo del problema planteado y se fundamentan el uso de la metodología de desarrollo y las tecnologías que se utilizarán para el desarrollo del módulo.

1.2 Conceptos asociados al dominio del problema.

1.2.1 Video

Proviene del verbo latino *videre*, y significa "yo veo". Hace referencia a la captación, procesamiento, transmisión y reconstrucción de una secuencia de imágenes y sonidos que representan escenas en movimiento. (1) Los comienzos del video están relacionados con el intento de cubrir las necesidades que tenía la televisión. Actualmente está asociado a distintos formatos de almacenamiento, ya sea análogos como digitales. Las primeras trasmisiones televisivas se realizaban en vivo y con la posibilidad de grabarlas se facilitaba el trabajo de programación.

1.2.2 Sonido

Del latín *sonĭtus*, un sonido es una sensación que se produce en el oído por el movimiento vibratorio de los cuerpos. Estas vibraciones se transmiten por el aire u otro medio elástico. El sonido audible para los seres humanos está formado por las oscilaciones de la presión del aire, que el oído convierte en ondas mecánicas para que el cerebro pueda percibirlas y procesarlas. Las cuatro cualidades principales del sonido son la altura (grave, agudo o medio según la frecuencia de las ondas), la duración (el tiempo en el cual se mantiene el sonido), el timbre (depende los armónicos) y la intensidad (la cantidad de energía que contiene). (2)

El sonido es una vibración producida por un objeto, que se transmite en forma de ondas, a través del aire. Hay sonidos que no pueden ser captados por el oído humano y sí por los animales. El sonido necesita un medio en el cual propagarse y es por esto que no puede producirse en el vacío.

1.2.3 Televisión

Es un sistema de transmisión de imágenes y sonido a distancia a través de ondas hertzianas. (3)En el caso de la televisión por cable, la transmisión se concreta a través de una red especializada. La noción de televisión surgió con la combinación del vocablo griego *tele* (“distancia”) y el término latino *visio* (“visión”). El concepto permite referirse tanto al sistema de transmisión como al dispositivo que permite la visualización de las imágenes (también llamado televisor), la programación televisiva y la emisora de televisión. (4)

La televisión es un medio de comunicación masiva que tiene la capacidad de transmitir imágenes y sonido. La transmisión de televisión puede ser efectuada a través del aire, cable o por satélite Es un gran instrumento que permite informar a las personas del acontecer mundial y se ha convertido en un elemento esencial en el desarrollo de la vida del hombre y su cultura.

1.2.4 Plataforma de Televisión Informativa PRIMICIA

Es un producto informático desarrollado completamente con software libre. Provee un canal de televisión que permite integrar varios formatos multimedia: imagen, texto, video y audio.

Está estructurado en dos subsistemas:

- Subsistema de administración.
- Subsistema de transmisión.

El subsistema de administración permite la administración del canal y toda la gestión de las noticias y recursos multimedia. El subsistema de transmisión es el encargado de visualizar las noticias y materiales publicados.

El canal informativo muestra de forma automática ciclos de noticias constantes y repetitivos condicionados por las informaciones publicadas para determinados períodos de tiempo. Durante las transmisiones permite la reproducción de un fondo musical que puede ser personalizado según la noticia que se muestra, además es posible la utilización de infocintas o también llamados cintillos informativos que permiten el adelanto o emisión de breves informaciones de carácter relevante o promocional. (5) También muestra Informaciones adicionales como es la fecha y hora, tiempo restante de la noticia o pantalla y titular de la próxima noticia. Además permite la transmisión de señales de video externas que pueden ser provenientes de otro canal de televisión o de una filmación en vivo.

1.2.5 Traza

En sintaxis, la traza es una marca o huella dejada. Un registro de trazas es la conservación de los datos en una estructura definida de un evento sucedido o también de un elemento al haberse desplazado a una nueva posición. (6) Una traza es la evidencia que deja un evento ocurrido la cual puede ser almacenada para tener un registro que permita tomar una acción respecto a dicho evento. Esto posibilita el análisis para determinar un posible comportamiento de lo que ha sucedido.

1.2.6 XML[1]

El XML es considerado como un metalenguaje de definición de documentos estructurados mediante marcas o etiquetas. Se trata de un estándar del W3C[2] cuyo objetivo es crear unas reglas básicas para permitir el intercambio de información estructurada entre aplicaciones y en particular entre aplicaciones web. (7)Permite que los diseñadores creen sus propias etiquetas, permitiendo la definición, transmisión, validación e interpretación de datos entre aplicaciones. Es un sistema estándar de codificación de información creado para enriquecer la estructura de los documentos que pueden ser usados en la web.

1.2.7 Error

Se refiere a algo equivocado o desacertado. Puede ser una acción, un concepto o algo que se hizo erradamente. (8)Es equivalente a la equivocación y puede ser sobre hechos o sobre ideas. También puede ser un efecto o consecuencia no deseada de una determinada acción.

Diferentes empresas que trabajan específicamente con la transmisión televisiva han tenido la necesidad de gestionar los errores en sus aplicaciones para lograr así una mayor eficiencia en el tratamiento de los errores que se podrían dar a la hora de la transmisión.

Al abordar todos estos conceptos se resume que:

- Un video es una secuencia de imágenes y sonidos formando escenas en movimientos.
- El sonido es una sensación capturada por el oído a consecuencia de la vibración de los cuerpos. Se propaga a través del aire u otro medio elástico.
- La televisión es un medio de difusión para transmitir imágenes y sonidos ya sea por cable o por ondas.
- La Plataforma de Televisión Informativa, PRIMICIA es una solución informática capaz de proveer una canal de televisión. Puede combinar formatos de texto, imagen, video y sonido para que la

[1] Extensive Markup Language en español: Lenguaje de marcas extensible
[2] World Wide Web Consortium

información sea amena. PRIMICIA está dividida en dos subsistemas: transmisión y administración.

- Una traza es un indicio de que ha sucedido algo y un registro de trazas es el almacenamiento de estas.
- XML es un estándar definido por la W3C para el intercambio de información estructurada entre diferentes plataformas.
- Un error es algo equivocado. Es algo que se realizó erradamente.

1.3 Objeto de Estudio.

La gestión de errores en la transmisión televisiva es indispensable para el trabajo óptimo de las aplicaciones que se encargan de transmitir información a los diferentes medios, además es la única forma de que los errores sean detectados de forma eficiente. Por ello resulta necesario el uso de algún mecanismo con este propósito.

En el centro de GEySED de la UCI se trabaja durante varios años en el desarrollo de la Plataforma de Televisión Informativa PRIMICIA. Hasta el momento la investigación se ha basado en el perfeccionamiento del proceso de redacción y transmisión de las noticias, sin embargo todo sería en vano si no se logra realizar la detección de los errores que ocurren cuando se transmite el canal informativo de la plataforma de televisión. Para lograr esto es necesario definir un procedimiento que garantice la detección de los errores y la estrategia a seguir para la generación y almacenamiento de las trazas generadas por los mismos.

El procedimiento general para la deteccion de errores es por medio de una técnica o a simple vista hasta identificarlo. En el caso de las aplicaciones informáticas también se utiliza la vista puesto que hay errores que son fáciles de encontrar. También se usan herramientas que automatizan la monitorización de los mismos. El modo realizado hasta el momento por los desarrolladores del producto PRIMICIA para detectar la ocurrencia de un error en la transmisión del canal es ir y buscar por intuición propia de donde podría provenir el error que ha ocasionado el mal funcionamiento o la interrupción del mismo. Se revisa la conexión con los diferentes servidores, la disponibilidad de los datos a los cuales accede la aplicación de transmisión, la estructura interna del XML de configuración y demás factores hasta encontrar que provoca el error.

De manera general el modo de registrar los errores ocurridos es de acuerdo a los detalles del mismo que puede ser la hora y fecha y otras características asociadas a él. Cuando se tienen estos detalles se guardarían en algún dispositivo electrónico o se registraría en papel. En el caso del electrónico que puede ser una computadora se guardaría en un fichero ya sea un XML o documento de texto y también en una base de datos.

Todos los elementos mencionados anteriormente son imprescindibles para lograr un producto con calidad y que se ajuste a las necesidades actuales de PRIMICIA. Por tanto se hace necesario analizar el procedimiento descrito, sirviendo éste de base para la implementación del Módulo de Gestión de Errores.

A continuación se mencionan procesos y situaciones que generan errores:

- **Error de orígenes de datos:** Los recursos multimedia no se reproducen o visualizan porque no está disponible la ruta o dirección donde se encuentran o porque hay desconexión con los servidores.
- **Error de base de datos:** La conexión de la aplicación de transmisión con la base de datos deja de existir porque está caído el servidor, el proceso no se está ejecutando o porque existe error al intentar conectarse.
- **Error de existencia de medias:** Los videos, imágenes y música que se encuentran en base de datos no se hallan físicamente en el servidor de medias.
- **Error en el XML de configuración:** El XML de configuración posee errores de validez en su estructura y también que alguna o algunas etiquetas a las cuales se necesita acceder para recoger su contenido no existan o estén sin éste.
- **Error de alto consumo de RAM y CPU:** El consumo de RAM y CPU de la aplicación de transmisión supera los establecidos por el usuario en el Módulo de Gestión de Errores.

1.4 Análisis de soluciones existentes

En la actualidad no existe casi ninguna vía que ofrezca solución al problema que se pretende solucionar. Las pocas que hay solo están implementadas para sistemas privativos y sus precios son bastante elevados dada la complejidad de este tipo de producto y la gran cantidad de funcionalidades que debe tener disponible para la labor para la cual están diseñados. Además, por regla general solo permiten un trabajo muy limitado en el hecho mismo de la gestión de errores.

1.4.1 GFI Events Manager

En Europa la compañía GFI, desarrollador internacional de software de seguridad de red, seguridad de contenido y mensajería, ha desarrollado el software GFI Events Manager, el cual permite el seguimiento del funcionamiento de los dispositivos de hardware. Cubre dos funciones principales: monitorización, administración y archivo de sucesos. La más reciente compilación de GFI Events Manager mejora el nivel de alertas cuando se detectan en la red sucesos importantes o intrusiones. (9)

La funcionalidad operativa de GFI Events Manager se divide en 2 etapas: Recolección de sucesos y procesamiento de sucesos. Esta solucion permite a los administradores del sitema monitorizar el

estado y seguridad de toda la red, mejorando la disponibilidad. También permite al administrador centralizar los sucesos de múltiples orígenes en varios formatos posibilitando una forma más ágil de detectar errores y proveer trazas. Además puede activarse el envío de alertas a una o varias personas a través de correo electrónico o SMS.

1.4.2 Plataforma Communi.TV

La plataforma Communi.TV (CTV) permite la publicación de contenidos y aplicaciones para web, móviles y televisión digital mediante componentes reutilizables en diferentes tecnologías que facilitan el despliegue de servicios. CTV es una plataforma de desarrollo que está orientada a favorecer la productividad y la creación de nuevos formatos en canales digitales. Una vez instalado el sistema se proporcionarán una serie de scripts de monitorización que interrogan al gestor de contenidos para determinar su disponibilidad y rendimiento. Los scripts están orientados a recoger métricas de funcionamiento de la plataforma. Con estas métricas y las proporcionadas por el cliente se determinará la necesidad de escalabilidad de la plataforma. (10)
La plataforma cuenta con un sistema de registro de trazas y monitorización de errores. Registra los tiempos de respuestas totales y parciales permitiendo detectar, controlar y corregir los cuellos de botella y la generación de alertas.

Al analizar los sistemas informáticos expuestos anteriormente se decide no utilizarlos pues fueron desarrollados con fines privativos. Analizando el costo que conlleva la utilización de alguna de estas aplicaciones a partir de la compra de su licencia y además de que estas no realizarían, al ser soluciones generales, una detección de todos los errores que ocurren en el canal, se decide realizar el Módulo de Gestión de Errores para la Plataforma de Televisión Informativa. La solución a realizar es muy específica puesto que está dirigida a un producto particular.

1.5 Lenguaje de modelado y metodología de desarrollo

El lenguaje de modelado que se utilizará es UML pues es el que emplea el Proceso Unificado de Desarrollo de Software RUP, el cual se usará en todo el ciclo del desarrollo de la solución por política interna del proyecto PRIMICIA. UML servirá para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo del software. RUP posibilitará la generación de los artefactos y la documentación durante todas las fases de la realización del módulo de gestión de errores.

1.5.1 El Lenguaje Unificado de Modelado (UML) como soporte de la modelación de la solución propuesta.

Es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Es una notación y no un proceso o método, es decir, es una herramienta útil para representar los modelos del sistema en desarrollo, pero no ofrece ningún tipo de guía o criterios acerca de cómo obtener esos modelos. Es decir, UML indica qué es lo que teóricamente hará el sistema, pero no cómo lo hará. (11)

UML es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Se puede utlizar con cualquier metodología puesto que es independientemente del ciclo de desarrollo que se vaya a seguir. Incluye un conjunto de técnicas de notación gráfica para crear modelos visuales de programación orientada a objetos de sistemas en desarrollo.

1.5.2 El Proceso Unificado de Desarrollo de Software (RUP) como base en el desarrollo de la solución.

Las metodologías de desarrollo de software ayudan a realizar todo el proceso de desarrollo del software de una forma más rápida. Entre las metodologías más utilizadas por su fortaleza en el proceso de desarrollo del software se destaca el Proceso Unificado de Desarrollo de Software (RUP). Para describir lo que se tiene y lo que se espera del software, RUP se basa en casos de uso. Esto garantiza una buena documentación basándose en el Lenguaje de Modelado Unificado (UML), la cual puede ser utilizada por los nuevos miembros del proyecto PRIMICIA.

Las fases de RUP permiten establecer la oportunidad y alcance del producto, también identifica las entidades externas y los actores con los que se trata, relacionándolos a los casos de uso. Al utilizar RUP quedará garantizada la planificación, desarrollo y mantenimiento del sistema.

Ventajas al utilizar RUP:

- Aplicación del conocimiento adquirido de una iteración a otra.
- Agilidad en la realización del modelado.
- Definir en cada momento del ciclo de vida del software, cuáles artefactos, con qué nivel de detalle y qué roles deben ser creados.
- Identificar problemas y fallos de modo que puedan prevenirse o corregirse a tiempo.

1.6 Tecnologías

El proyecto PRIMICIA tiene definidas cuales son las herramientas apropiadas para el desarrollo de la Plataforma de Televisión Informativa y algunas de estas están definidas en el documento de

Arquitectura de Software como es el caso de la herramienta de modelado y el sistema gestor de base de datos por lo que no se realiza un estudio de las diferentes tecnologías a utilizar.

1.6.1 Fundamentación de la herramienta de modelado:

En la actualidad muchas empresas se han extendido a la adquisición de herramientas CASE[3], con el fin de automatizar los aspectos claves de todo el proceso de desarrollo de un sistema. Visual Paradigm 8.0 es una herramienta fácil de utilizar que emplea las últimas notaciones de UML, ingeniería inversa y generación del código.

Principales ventajas en la utilización de la herramienta:

- Generación de código.
- Desarrollo y refinamiento visual de la solución, mediante la utilización de gráficos.
- Modelado de procesos de negocio, de requerimientos y de base de datos con la misma herramienta.
- Ambiente gráfico agradable e intuitivo.

Visual Paradigm es una herramienta CASE multiplataforma privativa con una versión libre. Utiliza UML 2.1 como lenguaje de modelado. Está diseñada para una amplia gama de usuarios interesados en construir sistemas de software fiables con el uso del paradigma orientado a objetos, incluyendo actividades como ingeniería de software, análisis de sistemas y análisis de negocios. (12)

1.6.2 Fundamentación del Lenguaje de Programación

Un lenguaje de programación es un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Un lenguaje de programación permite a uno o más programadores especificar de manera precisa sobre qué datos debe operar una computadora, cómo estos datos deben ser almacenados o transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias. (13)

C++ es un lenguaje imperativo derivado del C. En realidad un superconjunto de C, que nació para añadirle cualidades y características de las que carecía. El resultado es que como su ancestro, sigue muy ligado al hardware subyacente, manteniendo una considerable potencia para programación a bajo nivel, pero se la han añadido elementos que le permiten también un estilo de programación con alto nivel de abstracción. (14)

[3] Computer Aided Software Engineering, traducido al español significa Ingeniería de Software Asistida por Computación

Ventajas del lenguaje C++:

- Es de propósito general.
- Es un lenguaje multinivel, es decir, se puede usar tanto para programar directamente el hardware (dependiendo del sistema operativo), como para crear aplicaciones de escritorio.
- El sistema de detección de errores de C++ es mucho más robusto que el de C.
- Amplia documentación que ayuda a aclarar disímiles dudas.
- Rápida ejecución de código que permite la creación de los ejecutables más ligeros que se pueden construir para una máquina.
- En C++ se puede manejar la memoria sin ningún tipo de problema.

Además de estas ventajas también se escoge porque en los avances que se han hecho del producto PRIMICIA se ha utilizado C++ aportando gran dinamismo a la aplicación. También porque el subsistema de transmisión ha ganado experiencia con la utilización de este lenguaje en la realización de la versión actual del producto PRIMICIA.

1.6.3 Fundamentación del Framework:

En el desarrollo de software un framework es una estructura de soporte definida, mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros programas para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Para el desarrollo con C++ existen diferentes frameworks, como por ejemplo Crystal Space, .Net framework, Platinum, y Qt.

El framework Qt tiene licencia libre y privativa. Además proporciona una amplia colección de más de 500 clases escritas en C++ que ayudan a realizar un software con calidad y rapidez.

El framework seleccionado para desarrollar el Módulo de Gestión de Errores es Qt principalmente porque está escrito en su mayoría por el lenguaje escogido: C++ y proporciona varias herramientas y clases que permiten reducir el tiempo de desarrollo de una aplicación de escritorio. También porque el proyecto lo está utilizando en las nuevas versiones del subsistema de transmisión, aportando mejoras y dinamismo al producto.

Algunas de sus características: (15)

- Compatibilidad multiplataforma con un solo código fuente.
- Disponibilidad del código fuente.
- Excelente documentación.

- Arquitectura lista para el uso de plugins.

Componentes del framework Qt a utilizar:

- Las librerías Qt (amplia colección de clases escritas en C++)
- Qt Assistant: Acceso rápido a la documentación de Qt.
- QT Designer: Diseñador de interfaces gráficas de usuario.
- qmake: Simplifica el proceso de construcción de proyectos en las diferentes plataformas soportadas.

1.6.4 Fundamentación del Entorno de desarrollo integrado (IDE):

Un entorno de desarrollo integrado o IDE es un programa informático compuesto por un conjunto de herramientas de programación. Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. (16)Es posible que un mismo IDE pueda funcionar con varios lenguajes de programación.

Qt Creator 2.2.1 es un entorno de desarrollo integrado (IDE) multiplataforma adaptado a las necesidades de los desarrolladores de Qt para desarrollar aplicaciones en C++ de manera sencilla y rápida. Este, como el KDevelop, no cuenta con un compilador propio. Como su nombre lo indica, está basado en la biblioteca Qt. Presenta un sofisticado editor de código, integración con sistemas de control de versiones, un diseñador de formularios (GUI) integrado, herramientas para la administración y construcción de proyectos, completamiento de código y depurador visual.

Permite a los desarrolladores crear aplicaciones para ordenadores de escritorio y plataformas de dispositivos móviles. Se integra con la mayoría de los sistemas de control de versiones populares, incluyendo Git, Subversion, Perforce, CVS y Mercurial. Es un avanzado editor de código que provee soporte para edición de C++ y Java Script, ayuda sensible al contexto y completamiento de código. (17)

Por las características tan ventajosas que posee el IDE Qt Creator, la experiencia que se tiene en el proyecto en la utilización del mismo en la construcción del producto PRIMICIA y las ventajas anteriormente mencionadas se escoge para la realización del Módulo de Gestión de Errores del subsistema de transmisión.

1.6.5 Fundamentación del Sistema gestor de base de datos (SGBD):

Un sistema gestor de base de datos (SGBD) se define como el conjunto de programas que administran y gestionan la información contenida en una base de datos. El propósito general de estos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante para una organización. (18)Incluye características de la orientación a objetos pero no es un sistema de gestión de bases de datos puramente orientado a objetos. Soporta diferentes tipos de datos y permite la creacion de tipos propios. Puede ser utlizado en la mayoria de los sitemas operativos utlizados actualmente. Permite conexiones remotas lo cual es muy útil para administrar bases de datos desde otra computadora. Tambien tiene la posiblidad de realizar copias de seguridad para casos de accidente.
Ayuda a realizar las siguientes acciones:

- Definición de los datos.
- Mantenimiento de la integridad de los datos dentro de la base de datos.
- Control de la seguridad, manipulación y privacidad de los datos.

PostgreSQL 9.1 es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) basado en el proyecto POSTGRES de la universidad de Berkeley. Es una derivación libre (Open Source) de este proyecto. (19)
Además de que se debe utilizar por políticas del proyecto se escoge PostgreSQL por ser una herramienta libre y por la experiencia que tiene el grupo de trabajo del proyecto PRIMICIA desde la realización de las primeras personalizaciones del producto.

1.7 Conclusiones parciales

- Con el estudio de los diferentes conceptos asociados al dominio del problema se logró comprender mejor el contenido de la investigación.
- Al analizar el objeto de estudio se evidencia que es necesario realizar un mecanismo para detectar las fallas en el subsistema de transmisión para garantizar que la Plataforma de Televisión Informativa PRIMICIA funcione correctamente.
- Las aplicaciones analizadas permitieron realizar una vista global de los elementos más importantes que podrían aplicarse en el desarrollo del sistema como es el monitoreo de errores y el registro de trazas.

- Con el estudio de los diferentes lenguajes y herramientas se logró especificar los elementos fundamentales de los mismos para lo que serán utilizados. El lenguaje de modelado UML se utilizará pará indicar qué es lo que teóricamente hará el sistema, la metodología de desarrollo RUP para agilizar y guiar el proceso de desarrollo del sistema, la herramienta CASE Visual Paradigm para generar los artefactos durante el desarrollo del software, el lenguaje de programación C++ para que el sistema posea una rápida ejecución, el framework Qt para ahorrar tiempo de desarrollo, el entorno de desarrollo integrado Qt Creator para reducir el tiempo de programación gestor de Base de datos PostgreSQL para almacenar los datos.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA.

2.1 Introducción

En este capítulo se realizará la descripción de las principales definiciones asociadas al dominio del problema. También se especificarán los requisitos funcionales y no funcionales. Se identificarán los actores y casos de uso del sistema así como las relaciones entre ellos mediante el diagrama de casos de uso del sistema y las descripciones generales de cada caso de uso.

2.2 Diagrama de clases del dominio

Un modelo del dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. También se les denomina modelos conceptuales, modelo de objetos del dominio y modelos de objetos de análisis. El modelo del dominio podría considerarse como un diccionario visual de las abstracciones relevantes, vocabulario del dominio e información del dominio. (11)

Se determinó definir un modelo de dominio con el objetivo de capturar los objetos más importantes que existen o los eventos que suceden en el entorno donde estará el sistema, además existe la necesidad de modelar el problema para su mejor comprensión.

2.2.1 Modelo del Dominio

Se realizará un modelo del dominio porque no se tienen claramente definidos los procesos del negocio para realizar un modelado del mismo. También se explicarán los conceptos asociados a éste.

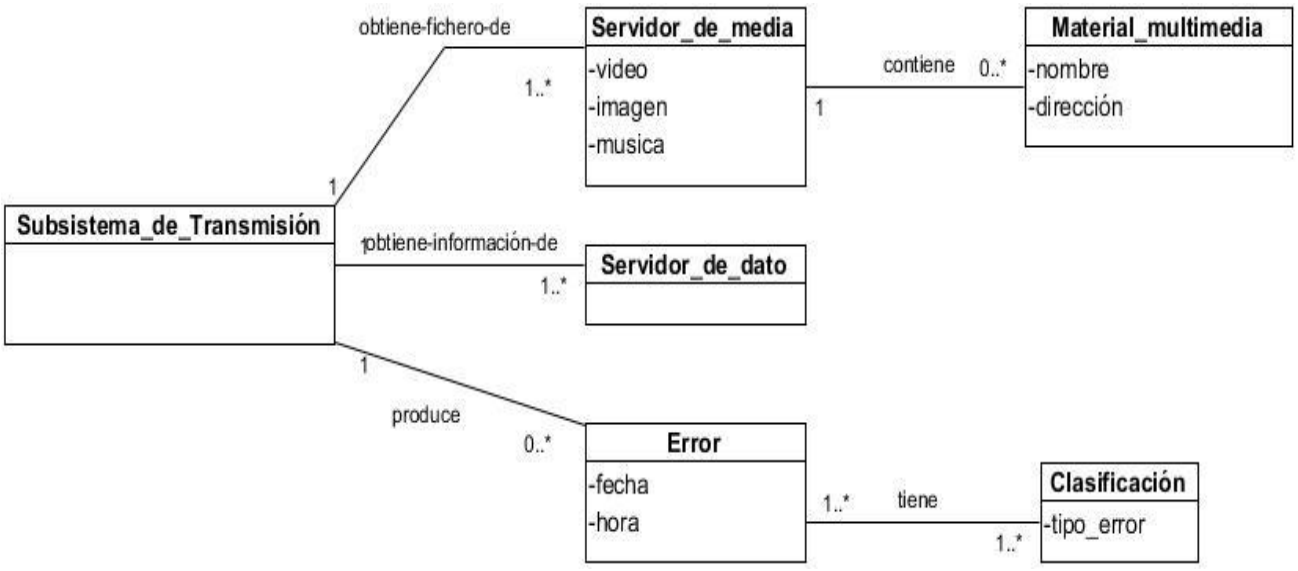


Figura 1: Modelo de Dominio

2.2.2 Definición de clases del modelo del dominio

Subsistema_de_Transmisión

Nombre de la clase:	Subsistema_de_Transmisión
Descripción:	Esta clase transmite la señal del canal informativo. Muestra las noticias que se confeccionan y publican en el Subsistema de Administración.

Tabla 1: Descripción de la clase Subsistema_de_Transmisión.

Servidor_de_media

Nombre de la clase:	Servidor_de_media
Descripción:	Almacena los distintos tipos de medias que se utilizarán en la transmisión de las noticias. Estas pueden ser de tipo imagen, video o música.

Tabla 2: Descripción de la clase Servidor_de_media.

Servidor_de_dato

Nombre de la clase:	Servidor_de_dato
Descripción:	Almacena las configuraciones de las noticias que se crean en el Subsistema de Administración. Estas son utilizadas por el Subsistema de Transmisión para mostrar el ciclo noticioso.

Tabla 3: Descripción de la clase Servidor_de_dato.

Material_multimedia

Nombre de la clase:	Fichero_multimedia
Descripción:	Son los diferentes archivos multimedia que se encuentran guardados en el servidor de medias.

Tabla 4: Descripción de la clase Fichero_multimedia.

Error

Nombre de la clase:	Error
Descripción:	Es el error que puede producirse en el transcurso de la transmisión del

	canal, pudiendo ocasionar un mal funcionamiento o interrupción del canal.

Tabla 5: Descripción de la clase Error.

Clasificación

Nombre de la clase:	Clasificación
Descripción:	Catalogación de los tipos de errores posibles. Estos pueden ser de base de datos, XML, orígenes de datos, existencia de medias y de consumo.

Tabla 6: Descripción de la clase Clasificación.

2.3 Estrategia para la detección y registro de errores ocurridos en el Subsistema de Transmisión.

En los días actuales las aplicaciones de escritorio deben tener el mínimo de errores para que el cliente se sienta confiado y atraído a éste no solo por la fachada. Existen diversas estrategias para manipular errores. Estas estrategias pueden ser desde aplicaciones creadas con el fin de gestionar los errores de otras como librerías que se pueden utilizar en el código programado.

La mejor práctica generalmente aceptada es utilizar un framework para la administración de errores (logging framework) que tiene los conceptos de: (20)

- **Objetos de registro:** Diferentes clases y módulos pueden entrar a los diferentes registradores de mensajes (*loggers*), así se puede optar por aplicar las configuraciones de registro a diferentes partes de la aplicación.
- **Niveles de registro:** Se puede modificar la configuración del registro y sólo registrar los tipos de errores que desee el usuario.
- **Salidas de registro diferentes:** El framework debe permitir configurar el resultado de registro sin necesidad de cambios en el código base. Algunos ejemplos de los diferentes lugares posibles que se desee enviar la salida de registro son archivos, bases de datos y correo electrónico.

El framework de registro de errores que se vaya a utilizar depende de la plataforma. La opción para C++ es log4cxx. Estos poseen diferentes niveles de trazas, envío de los errores a diferentes destinatarios y configuración por ficheros.

2.3.1 Log4cxx

Log4cxx es un logging framework para C++ modelado después de log4j. Apache log4cxx usa Apache Portable Runtime (APR) y se puede usar en cualquier plataforma soportada por APR. **l**og4cxx está

licenciado bajo la Apache License, una licencia de código abierto certificado por la iniciativa de Código Abierto. Tiene tres componentes principales: *loggers, appenders and layouts.* Estos tres tipos de componentes trabajan juntos para permitir a los desarrolladores registrar los mensajes según el tipo de mensaje y el nivel, y para controlar en tiempo de ejecución estos mensajes, el formato y el qué se informó. (21)

Una de las ventajas de la API[4] log4cxx es su capacidad de gestión. Una vez que las declaraciones de registro se han insertado en el código, puede ser controlado con los archivos de configuración. El paquete log4cxx está diseñado de manera que las declaraciones de registro pueden permanecer en el código enviado sin incurrir en un costo de rendimiento pesado. (21)

Sistemas operativos de Windows y las distribuciones de Linux tienen formas de registrar sus errores.

2.3.2 Windows

Las versiones 95 y 98 usan un registro prácticamente idéntico, pero la forma en que corrigen sus errores y lo preservan es muy diferente. El 95 guardaba una sola copia de respaldo del registro con cada arranque exitoso, en dos archivos llamados **user.da0** y **system.da0**. El 98 convoca al programa scanreg.exe en cada inicio para detectar y corregir errores y crear un backup con cada arranque exitoso. Sin embargo, estas copias se guardan ahora como archivos **.cab** en la carpeta Sysbckup de Windows. Todavía más importante, almacena cinco copias buenas de cinco arranques sucesivos (lo que normalmente equivale a cinco días). El primero se llama **rb000.cab** y el más reciente, **rb004.cab**. Los **.cab** son archivos comprimidos, como los .zip, y se pueden ver o extraer su contenido con la interfaz de Windows o con programas como el WinZip. (22)

Los errores registrados se pueden gestionar en el visor de sucesos en administración de equipos. El visor de sucesos permite a los usuarios supervisar los sucesos registrados en los registros de aplicación, de seguridad y del sistema.

2.3.3 Linux

El sistema de *logs* de Linux es un mecanismo estándar que se encarga de recoger los mensajes generados por los programas y aplicaciones y enviarlos a un destino predefinido. En cada mensaje consta la fuente (el programa que generó el mensaje), la prioridad (nivel de importancia del mensaje), la fecha y la hora. Los *logs* se guardan en archivos ubicados en el directorio */var/log*, aunque muchos

[4] Application Programming Interface en español interfaz de programación de aplicaciones

programas manejan sus propios *logs* y los guardan en */var/log/<programa>*. También es posible especificar múltiples destinos para un mismo mensaje. En el entorno gráfico hay varias aplicaciones para monitorizar *logs* como es el caso de **KSystemLog**, la cual es muy utilizada. Es muy habitual monitorizar en una consola el archivo */var/log/messages* con el comando tail -f /var/log/messages. (23)

2.3.4 Kaspersky

Existen aplicaciones de escritorio que poseen un registro de errores. Una de estas es el antivirus Kaspersky Anti-Virus versión 6.0 MP4 el cual permite registrar información acerca de los eventos de aplicaciones en el registro general de eventos de Microsoft Windows o en un registro de eventos específico de Kaspersky Anti-Virus (KasperskyEvent Log). (24)

2.3.5 Sistema Docal

Es una herramienta para el control de los documentos, registros y procesos relacionados con la Gestión de Procesos de Calidad y Medioambiental. (25)El Agente del Sistema Docal es un *software* auxiliar que permite enviar a los usuarios por e-mail una copia de los mensajes generados en el Sistema Docal, a través de una cuenta en cualquier servidor SMTP.

Estas son sus características principales: (25)

- Se instala en un único Servidor o PC de la red.
- Necesita que se le configure una cuenta de correo saliente en un servidor SMTP para utilizarla en el envío de los mensajes.
- Está permanentemente en ejecución.
- Puede conectarse a servidores SMTP con y sin autenticación.
- Un solo Agente puede configurarse para enviar el correo de todas las Bases de Datos que se hayan creado.
- Se puede configurar para inicio automático.
- Registra los errores en el caso de no poder enviar los correos.

En esta investigación se va a realizar una estrategia de registro de errores particular, acorde a lo que demanda el Módulo de Gestión de Errores, que posee algunas características de las estrategias mencionadas anteriormente como es el caso de guardar los errores en un fichero local y en una base de datos.

La estrategia que se va a realizar en esta investigación es que después de detectado el error se registrará sus detalles (hora, fecha, tipo de error y descripción) en una base de datos en caso de que el mismo no esté relacionado con ella; de lo contrario se guardará en un archivo local, el cual se borrará al subirse su información a la base de datos cuando se reanuda la conexión. Este borrado y

envío a la base de datos de las trazas garantizará, en el caso de tener un fichero, que solo se tenga que realizar la lectura a éste. Al borrar el archivo local se logrará que no se acumule información sin necesidad en disco duro y teniéndolos en la base de datos se puede realizar operaciones con estos como es el filtrado de los mismos por tipo y por fecha y hora.

2.4 Requerimientos del sistema.

Los requerimientos pueden ser funcionales y no funcionales. Son las características que el sistema debe cumplir. Estos influyen en el producto realizado puesto que una mala definición y aplicación de los mismos pueden traer como consecuencia un *software* con mala calidad.

2.4.1 Requisitos funcionales.

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir.

RF 1: Identificar error de base de datos: El sistema debe ser capaz de identificar si existe desconexión con la base de datos.

RF 2: Leer fichero local: El sistema debe ser capaz de interpretar el XML local donde se guardan los errores en caso de no existir conexión con la base de datos.

RF 3: Subir error a base de datos a partir de fichero local: El sistema debe ser capaz de guardar en base de datos los errores contenidos en el fichero local al existir conexión con la base de datos.

RF 4: Identificar error de origen de los datos:

RF 4.1: Identificar error de disponibilidad de videos: El sistema debe ser capaz de identificar si la ruta de los videos en el servidor de medias se encuentra disponible.

RF 4.2: Identificar error de disponibilidad de imágenes: El sistema debe ser capaz de identificar si la ruta de las imágenes en el servidor de medias se encuentra disponible.

RF 4.3: Identificar error de disponibilidad de música: El sistema debe ser capaz de identificar si la ruta de la música en el servidor de medias se encuentra disponible.

RF 4.4: Identificar error de disponibilidad de logos: El sistema debe ser capaz de identificar si la ruta de los logos en el servidor de medias se encuentra disponible.

RF 4.5: Identificar error de disponibilidad de videos e imágenes del canal: El sistema debe ser capaz de identificar si la ruta de los videos e imágenes que se encuentra dentro del proyecto de transmisión está disponible.

RF 5: Identificar error de existencia de medias:

RF 5.1 Identificar error de existencia física de videos: El sistema debe ser capaz de identificar si los videos guardados en base de datos coinciden con los físicos en el servidor de medias.

RF 5.2 Identificar error de existencia física de imágenes: El sistema debe ser capaz de identificar si las imágenes guardadas en base de datos coinciden con las físicas en el servidor de medias.

RF 5.3 Identificar error de existencia física de música: El sistema debe ser capaz de identificar si la música guardada en base de datos coinciden con la física en el servidor de medias.

RF 6: Identificar error de consumo: El sistema debe ser capaz de identificar si el consumo de RAM y CPU de la aplicación de transmisión se encuentra por encima del porciento establecido por el usuario.

RF 7: Registrar errores en fichero: El sistema debe ser capaz de registrar los errores identificados en un fichero local en caso de no existir conexión con la base de datos.

RF 8: Guardar error en base de datos: El sistema debe ser capaz de guardar en la base de datos, en caso de haber conexión, los errores que se producen.

RF 9: **Identificar error de XML:** El sistema debe ser capaz de encontrar las deficiencias en la estructura del XML que contiene los datos de configuración del subsistema de transmisión.

RF 10: Buscar error:

RF 10.1: Buscar error por tipo: El sistema debe ser capaz de buscar los errores por base de datos, orígenes de datos, existencia de medias, consumo y XML.

RF 10.2: Buscar error por rango de fecha y hora: El sistema debe ser capaz de buscar los errores de acuerdo a un rango de fecha y hora especificado.

RF 11: Visualizar medias inexistentes: El sistema debe ser capaz de mostrar los nombres de las medias inexistentes detectadas.

RF 12: Visualizar errores del XML: El sistema debe ser capaz de mostrar los errores encontrados en el XML de la configuración del subsistema de transmisión.

RF 13: Modificar porciento de consumo de RAM y CPU: El sistema debe ser capaz de cambiar el o los porcientos máximos de consumo de RAM y CPU especificados por el usuario.

2.4.2 Requisitos no funcionales.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener, así sean restricciones del entorno o de la implementación. Representan las características del producto y son una parte significativa de la especificación de requisitos. Estos tienen gran impacto en el desarrollo y mantenimiento de un sistema.

Usabilidad

- El sistema de forma general debe brindar gran facilidad de uso para personas con poca experiencia con las computadoras pero con nivel calificado. Esto se logrará con un diseño ameno en la interfaz donde todos los elementos mostrados estarán muy organizados.

Soporte

- El soporte y/o mantenimiento del sistema no debe detener el servicio de trasmisión del canal informativo. Esto se logrará al realizarse el módulo en una aplicación aparte.

Restricciones de diseño y la implementación

- El diseño y la implementación deben tener una arquitectura flexible, que permita la fácil integración o desintegración de componentes. Esto se logrará con la aplicación de diferentes patrones de diseño y utilizando el patrón Arquitectura en Capas para una mejor organización de las clases.
- La aplicación no debe impactar ni alterar el funcionamiento de la aplicación del subsistema de transmisión. Para esto el Módulo de Gestión de Errores se ejecutará en una aplicación aparte de la de transmisión.
- Para la modelación del sistema se utilizará como lenguaje de modelado UML y como herramienta CASE Visual Paradigm 8.0.
- El sistema estará implementado en lenguaje C++, utilizando como IDE de desarrollo el Qt Creator.

Software

- Utilizar Qt Creator 2.2.1 como IDE de desarrollo y PostgreSQL 8.4 o superior como sistema gestor de base de datos.
- Utilizar como sistema operativo en los servidores la distribución de Ubuntu 10.04 o superior.

Hardware

A continuación se muestran los requerimientos mínimos y óptimos recomendados que debe cumplir el equipamiento tecnológico de la plataforma:

Servidor	Requerimientos	Procesador	Memoria RAM	Disco Duro	Tarjeta de Red
Transmisión	Mínimos	Pentium IV 2.8 GHz	512 Mb	100 Mb	Ethernet 10/100 Mbps
	Óptimos	Dual-Core Xeon 1.60 GHz	1Gb	500Mb	Ethernet 10/100 Mbps

Tabla 7: Requerimiento de Hardware.

Requisitos Legales, de Derecho de Autor y otros.

Los derechos de autor y otros van estar determinados por la entidad comercializadora de la UCI.

2.5 Modelo de Casos de Uso

El modelo de casos de uso permite visualizar de una forma sencilla las funcionalidades que deberá tener el sistema propuesto. En el mismo se definen los actores y casos de uso correspondientes.

2.5.1 Definición de los actores del Sistema

Actor	Justificación
Reloj	Es el que procede a iniciar todo el proceso de detección de errores que pueden ocurrir en la ejecución del subsistema de transmisión.
Usuario	Es la persona que procede a ejecutar las funcionalidades de filtrar errores, visualizar errores de existencia de medias y de XML y cambiar los valores máximos de consumo de RAM y CPU.

Tabla 8: Definición de los actores del sistema.

2.5.2 Descripciones textuales de Casos de Uso del sistema

Caso de Uso del Negocio: Identificar error de base de datos.

Caso de Uso: CU-1	Identificar error de base de datos
Actores:	Reloj
Resumen:	El caso de uso inicia cuando el actor Reloj inicializa el chequeo del estado de la conexión a la base de datos. El caso de uso finaliza cuando se muestra el estado de la conexión con la base de datos.
Precondiciones:	El Reloj debe activarse
Referencias	RF1, RF2, RF3, RF7
CU asociados:	
Prioridad	Crítico
Flujo Normal de Eventos	

Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el actor inicializa la comprobación de la conexión con la base de datos.	2. El sistema comprueba la conexión con la base de datos.
	3. Si se detecta el error el sistema detalla el mismo poniendo la fecha y hora en que ocurre, el tipo de error y la descripción, que estará en dependencia de por qué no hay conexión con la base de datos y lo guarda en un fichero local.
	4. Muestra en pantalla la ocurrencia del error informando que no hay conexión con la base de datos y finaliza el caso de uso. (Ver prototipo 1)
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
	3a. Si no hay error se comprueba que el fichero local existe, en el caso de existir se suben los errores contenidos en él a la base de datos y luego se elimina. Además se muestra en pantalla que hay conexión con la base de datos y finaliza el caso de uso. (Ver prototipo 2)
Prototipo de Interfaz de Usuario	
 **Prototipo 1** 	

Prototipo 2	
Pos condiciones	Que se actualice el estado de la conexión con la base de datos.

Tabla 9: Descripción textual del CU Identificar error de base de datos

Caso de Uso del Negocio: Identificar error de orígenes de datos.

Caso de Uso: CU-2	Identificar error de orígenes de datos
Actores:	Reloj
Resumen:	El caso de uso inicia cuando el actor Reloj inicializa la acción de comprobar la existencia de los orígenes de datos a los cuales accede el subsistema de transmisión en la ejecución del canal. El caso de uso finaliza cuando se muestra en pantalla si las diferentes rutas se encuentran disponibles o no.
Precondiciones:	El Reloj debe activarse
Referencias	RF4, RF7, RF8
CU asociados:	
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el actor inicializa la comprobación de la existencia del directorio de los orígenes de datos.	2. Comprueba la existencia del directorio de los videos, imágenes, música y los logos a los cuales accede el subsistema de transmisión en la ejecución del canal.
	3. Si existe algún error con la disponibilidad de estas rutas el sistema detalla los mismos poniendo la hora y fecha en que ocurre, el tipo de error y la descripción, que

	estará en dependencia de cual o cuales directorios no están disponibles.
	4. Se comprueba la conexión con la base de datos.
	5. Si no hay error de conexión se guardan los detalles en la base de datos.
	6. Muestra en pantalla la ocurrencia del error de orígenes de datos informando cual o cuales directorios no están disponibles y finaliza el caso de uso. (Ver prototipo 3)
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
	3a. Si no hay error con los orígenes de datos se muestra en pantalla que no hay error con los diferentes directorios y finaliza el caso de uso. (Ver prototipo 4)
	5a. Si hay error de conexión con la base de datos se guardan los detalles del error en un fichero local.
Prototipo de Interfaz de Usuario	

Prototipo 3

Prototipo 4

Pos condiciones	Que se actualice el estado de disponibilidad de los orígenes de datos.

Tabla 10: Descripción textual del CU Identificar error de orígenes de datos.

Caso de Uso del Negocio: Identificar error de existencia de medias.

Caso de Uso: CU-3	Identificar error de existencia de medias
Actores:	Reloj
Resumen:	El caso de uso inicia cuando el actor Reloj inicializa la acción de comprobar que las medias que se encuentran guardadas en la base de datos se encuentran físicamente en el servidor de medias. El caso de uso finaliza cuando se muestra en pantalla si las diferentes medias

	en base de datos se encuentran físicamente o no.
Precondiciones:	El Reloj debe activarse
Referencias	RF5, RF7, RF8
CU asociados:	
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el actor inicializa la comprobación de la coincidencia de las medias que están guardadas en la base de datos con las que están físicamente en el servidor de medias.	2. Comprueba la coincidencia de las medias que están guardadas en la base de datos con las que están físicamente en el servidor de medias.
	3. Si existe algún error con la existencia de las medias el sistema detalla los mismos poniendo la hora y fecha en que ocurre, el tipo de error y la descripción, que estará en dependencia de cual o cuales medias no se encuentran físicamente.
	4. Se comprueba la conexión con la base de datos.
	5. Si no hay error de conexión se guardan los detalles en la base de datos.
	6. Muestra en pantalla la ocurrencia del error de existencia de las medias informando cual o cuales medias no se encuentran físicamente, guarda en un listado los nombres de él o los ficheros inexistentes, se muestra un botón que permite visualizar el listado y finaliza el

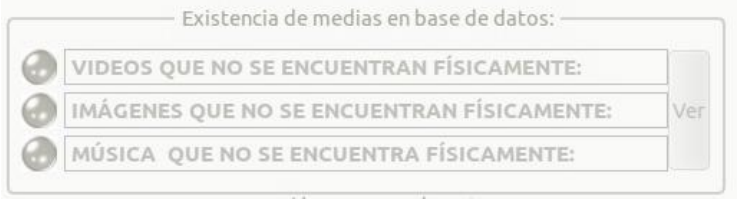
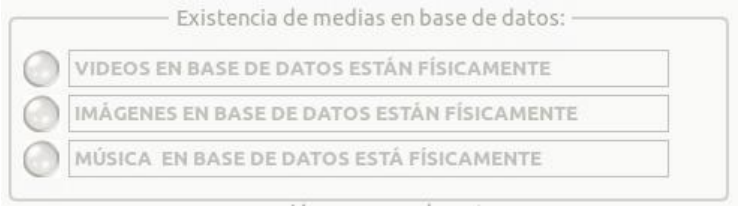
	caso de uso . (Ver prototipo 5)
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
	3a. Si no hay error se muestra en pantalla que las diferentes medias en base de datos están físicamente y finaliza el caso de uso. (Ver prototipo 6)
	5a. Si hay error de conexión con la base de datos se guardan los detalles del error en un fichero local.
Prototipo de Interfaz de Usuario	
 **Prototipo 5**  **Prototipo 6**	
Pos condiciones	Que se actualice el estado de existencia física de las medias guardadas en la base de datos.

Tabla 11: Descripción textual del CU Identificar error de existencia de medias.

Caso de Uso del Negocio: Identificar error de consumo.

Caso de Uso: CU-4	Identificar error de consumo
Actores:	Reloj
Resumen:	El caso de uso inicia cuando el actor Reloj inicializa la acción de comprobar que el consumo de RAM y CPU de la aplicación de transmisión no están por encima del porciento especificado. El caso de uso finaliza cuando se muestra en pantalla que el consumo de RAM y CPU están normales o altos.
Precondiciones:	
Referencias	RF6, RF7, RF8
CU asociados:	El Reloj debe activarse
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el actor inicializa la comprobación de error de consumo de RAM y CPU de la aplicación de transmisión.	2. Comprueba que el consumo de RAM y CPU de la aplicación de transmisión no sobrepasan el porciento especificado.
	3. Si existe algún error al estar el consumo por encima de este porciento el sistema detalla el mismo poniendo la hora y fecha en que ocurre, el tipo de error y la descripción, que estará en dependencia de si es el consumo de RAM, el de CPU o ambos están altos.
	4. Se comprueba la conexión con la base de datos.
	5. Si no hay error de conexión se guardan

	los detalles en la base de datos.
	6. Muestra en pantalla la ocurrencia del error de consumo informando cual o cuales consumos están altos y finaliza el caso de uso. (Ver prototipo 7)
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
	3a. Si no hay error se muestra en pantalla que el consumo de RAM y CPU están normales y finaliza el caso de uso. (Ver prototipo 8)
	5a. Si hay error de conexión con la base de datos se guardan los detalles del error en un fichero local.
Prototipo de Interfaz de Usuario	
 **Prototipo 7**	 **Prototipo 8**
Pos condiciones	Que se actualice el estado del consumo de RAM y CPU de la aplicación de transmisión.

Tabla 12: Descripción textual del CU Identificar error de consumo.

Caso de Uso del Negocio: Identificar error del XML.

Caso de Uso: CU-4	Identificar error del XML
Actores:	Reloj
Resumen:	El caso de uso inicia cuando el actor Reloj inicializa la acción de comprobar el contenido del XML de configuración del subsistema de transmisión. El caso de uso finaliza cuando se muestra en pantalla que el XML tiene error o no.
Precondiciones:	
Referencias	RF7, RF8, RF9
CU asociados:	El Reloj debe activarse
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el actor inicializa la comprobación del contenido del XML de configuración del subsistema de transmisión.	2. Comprueba si el contenido del XML de configuración del subsistema de transmisión es válido.
	3. Si existe algún error con el contenido se detalla el mismo poniendo la hora y fecha en que ocurre, el tipo de error y la descripción, que estará en dependencia de que parte del contenido es el que contiene error.
	4. Se comprueba la conexión con la base de datos.
	5. Si no hay error de conexión se guardan los detalles en la base de datos.

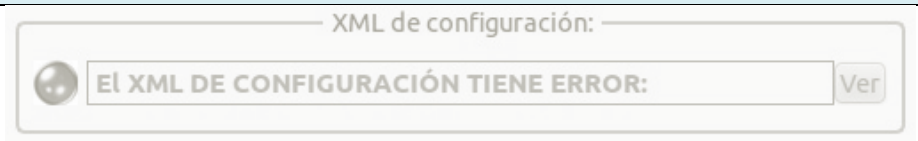
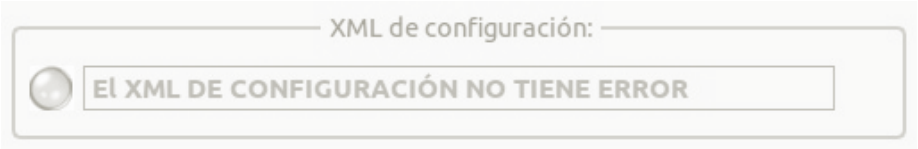
	6. Muestra en pantalla la ocurrencia del error informando que el XML contiene error, guarda en un listado la o las partes que contienen errores, se muestra un botón que permite visualizar el listado y finaliza el caso de uso. (Ver prototipo 9)
Flujos Alternos	
Acción del Actor	Respuesta del Sistema
	3a. Si no hay error en el XML se muestra en pantalla que no hay error con el contenido del mismo y finaliza el caso de uso. (Ver prototipo 10)
	5a. Si hay error de conexión con base de datos se guardan los detalles del error en un fichero local.
Prototipo de Interfaz de Usuario	
 **Prototipo 9**  **Prototipo 10**	
Pos condiciones	

Tabla 13: Descripción textual del CU Identificar error del XML.

Caso de Uso del Negocio: Realizar reportes de errores.

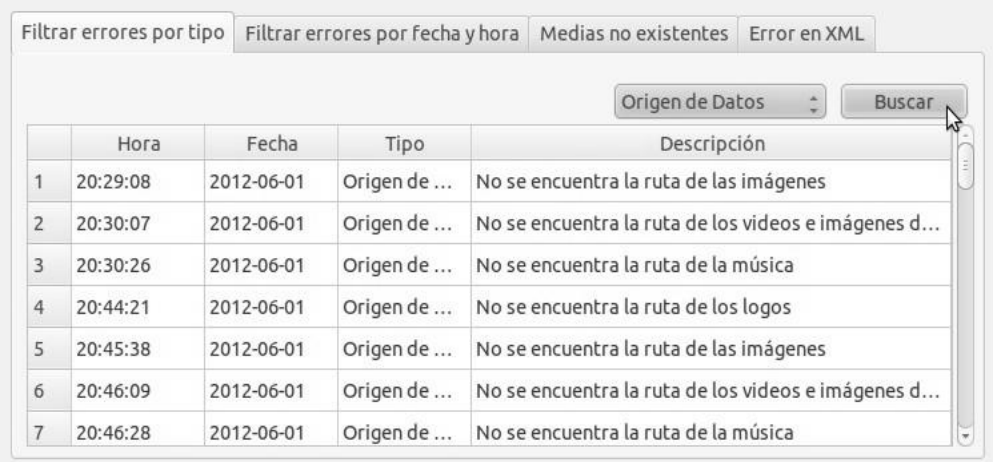
Caso de Uso: CU–6	Realizar reportes de errores
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el actor accede a ver los reportes de errores, dándole la posibilidad de buscar por tipo, fecha y hora. El caso de uso finaliza cuando el sistema muestra el listado con los errores que coinciden con el tipo de búsqueda escogido.
Precondiciones:	
Referencias	RF10
CU asociados:	
Prioridad	Secundario

Flujo Normal de Eventos	
Sección “Buscar error”	
Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el actor selecciona la pestaña de búsqueda de errores por tipo o a la de por fecha y hora.	2. Si elige: ➢ Por tipo de error. Ver sección Buscar por tipo de error. ➢ Por fecha y hora. Ver sección Buscar por fecha y hora.
Sección “Buscar por tipo de error”	
Acción del Actor	Respuesta del Sistema
	1. El sistema muestra un campo para escoger el tipo de error por el que desea buscar, que puede ser de base de datos, orígenes de datos, existencia de medias,

	consumo y de XML.
2. El usuario escoge el tipo de error y presiona el botón Buscar.	3. El sistema busca los errores en la base de datos según el tipo escogido, muestra un listado de los mismos y finaliza el caso de uso. (Ver prototipo 11)

Flujos Alternos	
Acción del Actor	Respuesta del Sistema
	3a. Si no escoge un tipo de error se muestra un mensaje indicando que escoja uno. (Ver prototipo 12)

Prototipo de Interfaz de Usuario



Prototipo 11

Prototipo 12

Sección “Buscar por fecha y hora”

Acción del Actor	Respuesta del Sistema
	1. El sistema muestra dos campos para la fecha y hora de inicio y otros dos para la fecha y hora de fin del rango de tiempo en que desea buscar.
2. El usuario introduce la fecha y hora de inicio y de fin y presiona el botón Buscar.	3. El sistema muestra un listado de los errores comprendidos en el rango de fecha escogido y finaliza el caso de uso. (Ver prototipo 13)

Flujos Alternos	
Acción del Actor	Respuesta del Sistema
	3a. Si la fecha de inicio es mayor que la de fin se muestra un mensaje indicando que el rango no es válido. (Ver prototipo 14)

Prototipo de Interfaz de Usuario

Prototipo 13

Prototipo 14

Pos condiciones	

Tabla 14: Descripción textual del CU Realizar reportes de errores.

Caso de Uso del Negocio: Visualizar medias inexistentes.

Caso de Uso: CU–7	Visualizar medias inexistentes
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el actor accede a ver las medias inexistentes. El caso de uso finaliza cuando el sistema muestra un listado de las medias inexistentes organizadas por tipo.
Precondiciones:	
Referencias	RF11
CU asociados:	
Prioridad	Opcional

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el actor accede a ver las medias inexistentes. (Ver prototipo 15)	2. El sistema muestra un listado de las medias: video, imagen y música que no se encuentran físicamente en el servidor de medias y finaliza el caso de uso. (Ver prototipo 16)

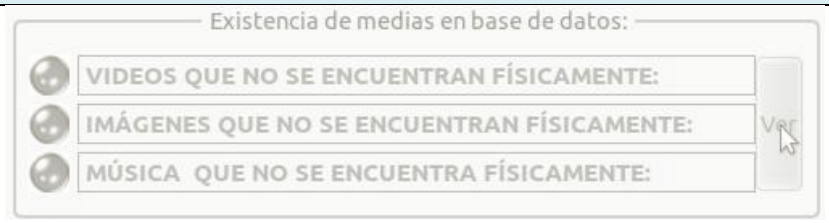
Prototipo de Interfaz de Usuario	
 **Prototipo 15**  **Prototipo 16**	
Pos condiciones	

Tabla 15: Descripción textual del CU Visualizar medias inexistentes.

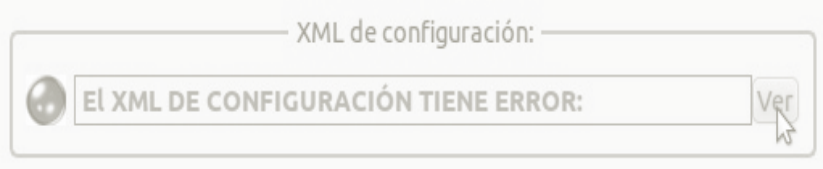
Caso de Uso del Negocio: Visualizar errores del XML.

Caso de Uso: CU–8	Visualizar errores del XML
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el actor accede a ver los errores que tiene el XML de configuración del Subsistema de Transmisión. El caso de uso finaliza cuando el sistema muestra un listado de los errores encontrados en el XML.
Precondiciones:	

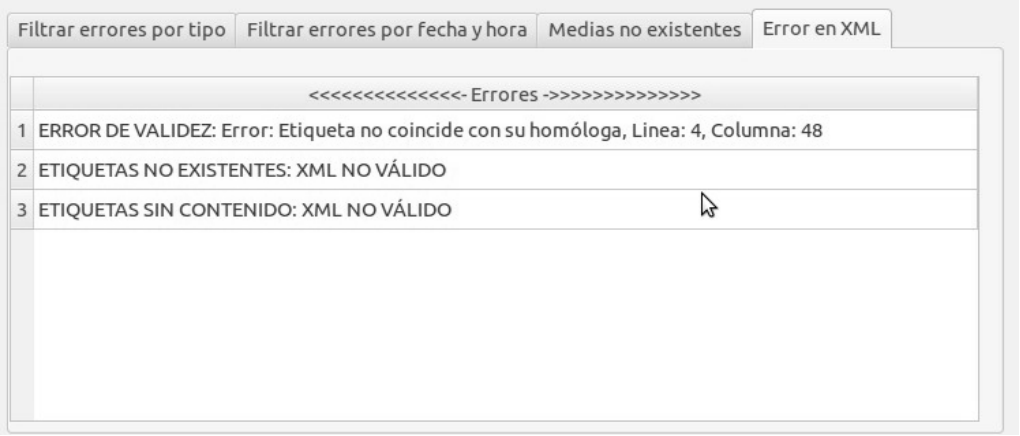
Referencias	RF12
CU asociados:	
Prioridad	Opcional

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el actor accede a ver los errores del XML. (Ver prototipo 17)	2. El sistema muestra un listado de los errores que tiene el XML de configuración del Subsistema de Transmisión y finaliza el caso de uso. (Ver prototipo 18)

Prototipo de Interfaz de Usuario



Prototipo 17



Prototipo 18

Pos condiciones	

Tabla 16: Descripción textual del CU Visualizar errores del XML.

Caso de Uso del Negocio: Modificar porciento de consumo de RAM y CPU.

Caso de Uso: CU–9	Modificar porciento de consumo de RAM y CPU
Actores:	Usuario
Resumen:	El caso de uso inicia cuando el actor accede a cambiar los valores de de RAM y CPU máximos a los que puede llegar la aplicación de transmisión. El caso de uso finaliza cuando se modifican estos valores.
Precondiciones:	
Referencias	RF13
CU asociados:	
Prioridad	Crítico

Flujo Normal de Eventos	
Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el actor accede a cambiar los valores máximos de RAM y CPU.	2. El sistema muestra dos campos para insertar los porcientos de RAM y CPU que el usuario estime conveniente.
3. El usuario inserta los valores y presiona el botón cambiar. (Ver prototipo 19)	4. El sistema muestra un mensaje informando que los valores se han cambiado satisfactoriamente y finaliza el caso de uso. (Ver prototipo 20)

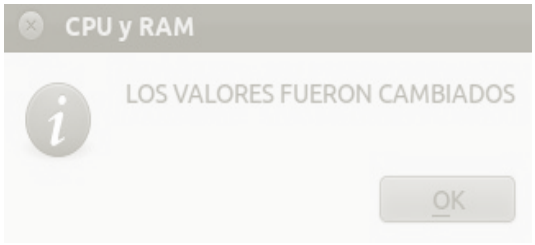
Prototipo de Interfaz de Usuario	
	
Prototipo 19	**Prototipo 20**
Pos condiciones	

Tabla 17: Descripción textual del CU Visualizar errores del XML.

2.5.3 Diagrama de Casos de Uso del Sistema

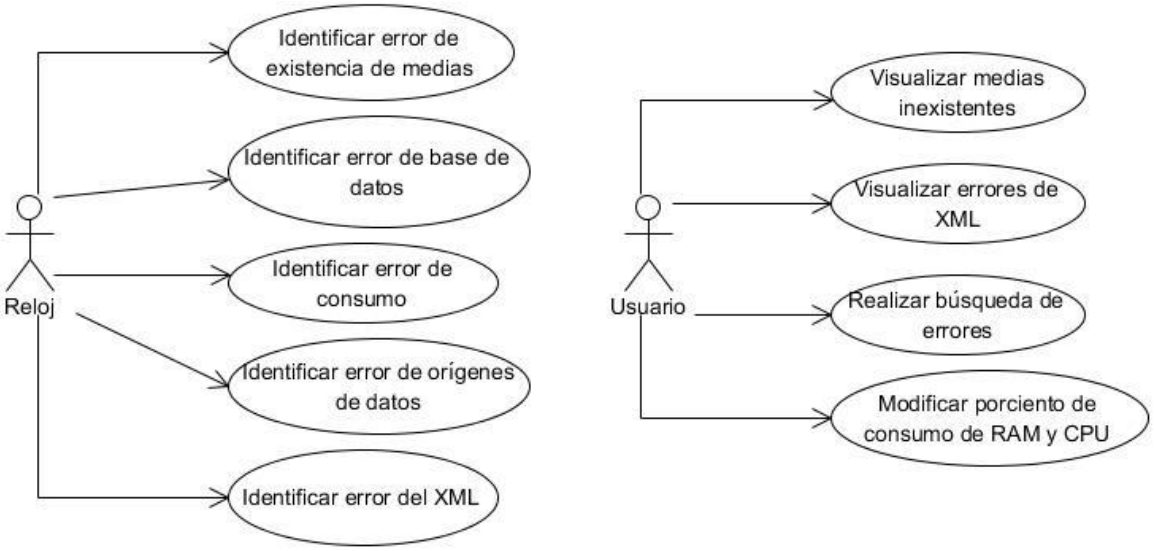


Figura 2: Diagrama general de casos de uso del sistema.

2.6 Conclusiones parciales

- Con la realización del modelo del dominio se mostraron las clases conceptuales lo que permitió una mejor comprensión del dominio del problema.
- El estudio de sistemas que realizan la detección y registro de errores ayudaron a realizar una estrategia en la que se aplican elementos de estas soluciones como es el caso de guardar los errores en un fichero local y en una base de datos.
- La especificación de los requisitos funcionales y no funcionales permitieron entender las características del sistema y lo que este debe hacer.
- El modelado de los casos de uso sirvió para mostrar con más detalles las funcionalidades que tendrá el sistema y su interacción con él o los actores, además posibilitó un mejor entendimiento de los requisitos del sistema.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA.

3.1 Introducción

En este capítulo se describe la arquitectura del sistema, haciendo énfasis en los patrones de diseño utilizados para la realización del mismo. Además se decide no realizar el modelado del análisis y se prioriza en el diseño del sistema, donde se elaboró el modelo de clases correspondientes a este flujo de trabajo.

3.2 Modelo de análisis

El propósito del análisis es lograr un refinamiento y estructuración de los requisitos capturados en fases anteriores para lograr una mayor compresión, reparación y modificación de los mismos. Por su parte el modelo de análisis especifica el comportamiento funcional del sistema, independientemente de los aspectos relativos al ambiente en el que va a ser finalmente implementado.

Un modelo de análisis puede considerarse como una primera aproximación al modelo de diseño, y es por tanto una entrada fundamental cuando se da forma al sistema en el diseño y en la implementación. Ofrece una especificación más precisa de los requisitos. Utilizado fundamentalmente por los desarrolladores para comprender cómo debería darse forma al sistema, es decir, cómo debería ser diseñado e implementado. (26)

¿Por qué no realizar el modelado del análisis?

Aunque el análisis forma parte de uno de los flujos de trabajo de RUP su empleo no es conveniente en todos los casos. RUP es un proceso configurable que puede ser adaptado a las necesidades del desarrollador, por lo que puede obviarse el modelado de análisis para de tal forma reducir el costo de tiempo cuando los requisitos del software permanecen simples, no existen grandes riegos y de realizarse el análisis, el cual constituye una primera aproximación del diseño no aportaría información relevante para este. En el caso particular de la investigación en cuestión, se tomó la decisión de no realizar el análisis por las razones antes expuestas y como tal para priorizar el ahorro de tiempo en actividades de diseño para incrementar el tiempo de implementación y validación del sistema.

3.3 Descripción de la Arquitectura

"Una arquitectura orientada a objetos bien estructurada está llena de patrones. La calidad de un sistema orientado a objetos se mide por la atención que los diseñadores han prestado a las colaboraciones entre sus objetos. Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles". (Grady Booch)

3.3.1 Arquitectura de software

La arquitectura del software desempeña un papel importante durante el desarrollo de un sistema. La misma permite organizar sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución. Expresan un esquema organizativo estructural para sistemas de software y definen las reglas generales de organización, las restricciones en la forma y la estructura de un grupo numeroso de aplicaciones. La selección de un patrón arquitectónico es una decisión básica del diseño en el desarrollo de un sistema. Para el desarrollo de las funcionalidades a implementar se utilizó el patrón Arquitectura en Capas, que define un modelo general de N-capas para la arquitectura lógica. (11) Como variante de este patrón se utilizará el “tres capas”. El mismo es un estilo de programación, cuyo objetivo primordial es la separación de la capa de presentación, capa de negocio y la capa de datos.

La capa de presentación: se encarga de proveer una interfaz entre el sistema y el usuario. Esta capa se comunica únicamente con la capa de negocio. También es conocida como interfaz gráfica y debe tener la característica de ser amigable para el usuario generalmente se presentan como formularios. (27)

La capa de negocio: es donde se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados y con la capa de datos, para solicitar al gestor de base de datos para almacenar o recuperar datos de él. (27)

La capa de datos: es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. (27)

Se selecciona la arquitectura anteriormente mencionada pues la misma facilita la modularidad del sistema, la localización de errores y mejora considerablemente el soporte del mismo. Cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior y las interacciones entre estas ocurren generalmente por invocación de métodos. Además permite el desarrollo paralelo del sistema por cada capa. Facilita el mantenimiento y soporte del proyecto. Mayor flexibilidad (se pueden añadir nuevos módulos para dotar al sistema de nueva funcionalidad). Con el uso del modelo de tres capas se logra en cierta medida obtener una mejor organización durante la realización del software. En dicha arquitectura a cada nivel se le confía

una misión concreta, lo que permite el diseño de una arquitectura escalable (que pueden ser ampliada con facilidad en caso de ser necesario).

El uso de esta arquitectura se evidencia en el sistema propuesto, separando las clases necesarias para el funcionamiento del mismo y estructurándolas de forma que, cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior.

3.3.2 Patrones de diseño

Los patrones de diseño son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Al aplicarse, los diseños serán mucho más flexibles, modulares y reutilizables. Todo patrón de diseño debe ser reusable y, por lo tanto, aplicable a diferentes problemas de diseño en distintas circunstancias. (27)

Patrones GRASP[5]

Los patrones GRASP describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable. (28)

Experto: Se utilizará este patrón para la asignación de responsabilidades indicando que la clase que cuenta con la información necesaria es la responsable de manejar la misma. En la figura 5 se puede ver el uso de este patrón en el diseño de la clase CError, pues esta clase es la única que conoce los detalles de los errores ocurridos.

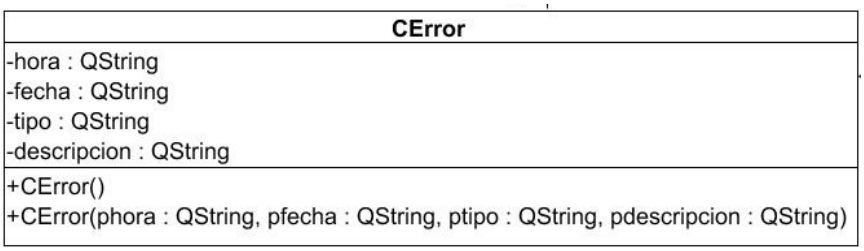


Figura 3: Diagrama donde se evidencia el uso del patrón Experto

[5] General Responsability Assignment Software Patterns, en español Patrones de Software para la Asignación General de Responsabilidad

Creador: Se utilizará este patrón dado las propiedades de todo sistema orientado a objetos. Se aplicará en los casos donde una clase tiene la responsabilidad de crear una nueva instancia de la otra. En la figura 6 se puede ver el uso de este patrón en el diseño de la clase CControladora, pues esta clase es la encargada de crear las instancias de las diferentes clases a utilizar.

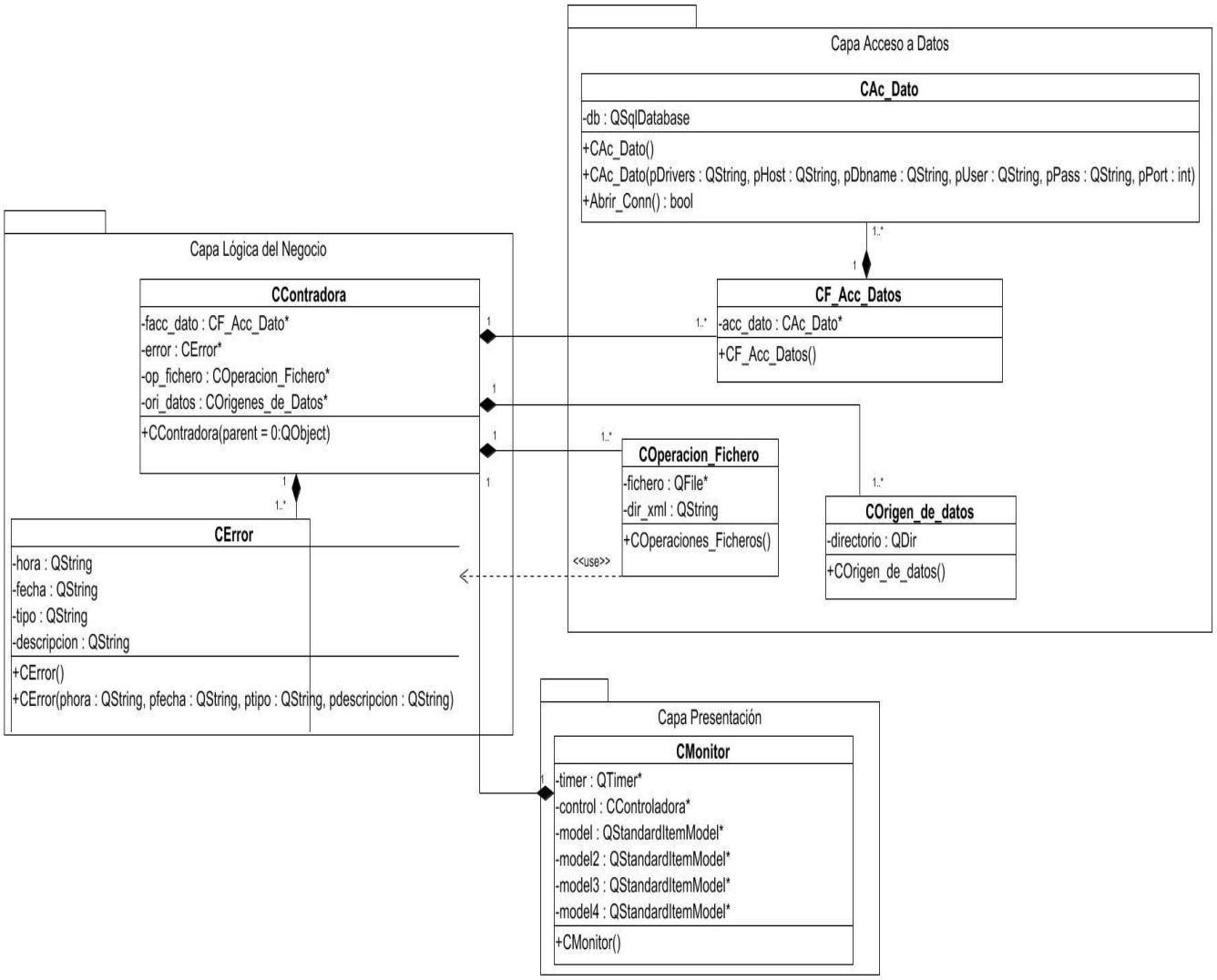


Figura 4: Diagrama donde se evidencia el uso del patrón Creador, Bajo Acoplamiento, Alta Cohesión y Controlador.

Bajo Acoplamiento: Se utilizará este patrón para fomentar la reutilización de código indicando una menor dependencia entre clases.

En la figura anterior, se brinda un ejemplo de cómo se ve evidenciado dicho patrón, ya que las clases están relacionadas de manera que se establecen sólo las dependencias necesarias para cumplir con sus responsabilidades, esto favorece a la flexibilidad del diseño y a la actualización de los cambios del sistema pues las clases son menos dependientes entre sí.

Alta Cohesión: Este patrón se utiliza para asignar responsabilidades a las clases de manera que todos sus métodos tuvieran un comportamiento bien definido. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.

Este patrón fue utilizado en el sistema desarrollado al implementar la clase controladora CControladora.

Patrón Controlador: Es una clase que, para el diseñador representa de alguna manera al sistema global. En la figura 6 se puede ver el uso de este patrón en la clase CControladora, la cual se encarga de los eventos más significativos del sistema.

3.4 Modelo de diseño

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en como los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además sirve de abstracción de la implementación del sistema y es, de ese modo, utilizada como entrada fundamental de las actividades de implementación. (26)

3.4.1 Diagrama de clases del diseño

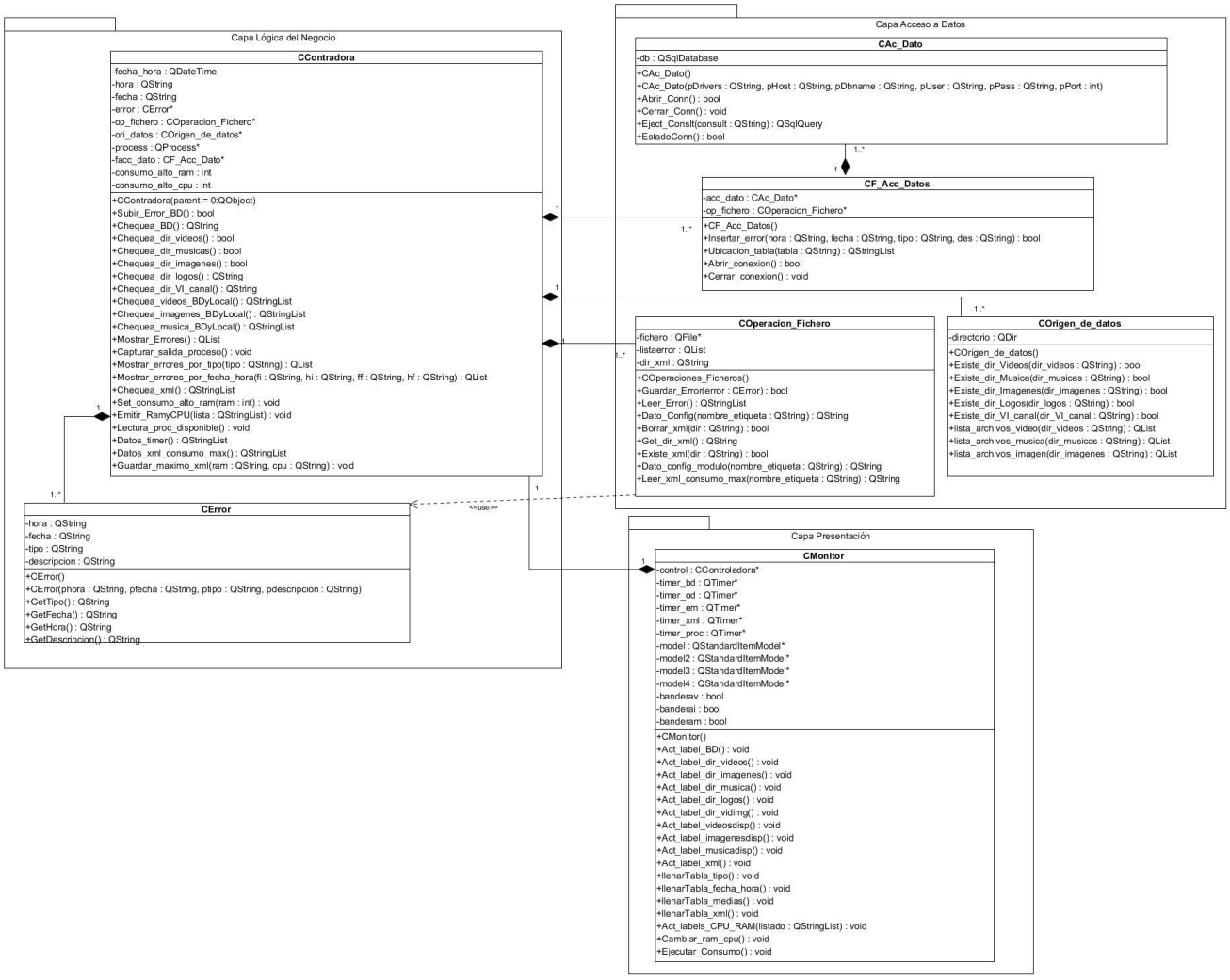


Figura 5: Diagrama de clases del diseño.

3.5 Diagrama de clases persistentes

El diagrama de clases persistentes muestra las clases capaces de mantener su valor en el espacio y en el tiempo.

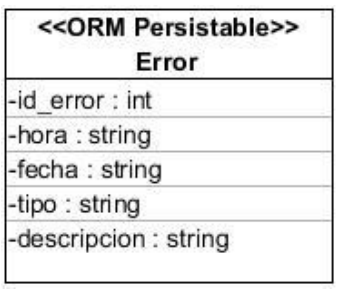


Figura 6: Diagrama de clases persistentes

3.6 Modelo de datos

En el Modelo de datos se describen las tablas que representan las distintas entidades que pertenecen al dominio del problema y serán almacenadas en la base de datos, en este caso se tiene solamente una tabla.

3.6.1 Diagrama Entidad- Relación

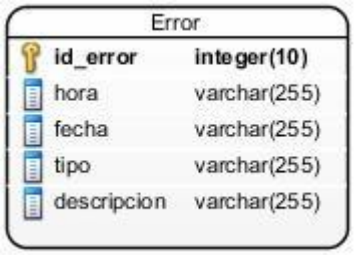


Figura 7: Diagrama Entidad - Relación.

3.7 Modelo de despliegue

El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. (26)

3.7.1 Diagrama de despliegue

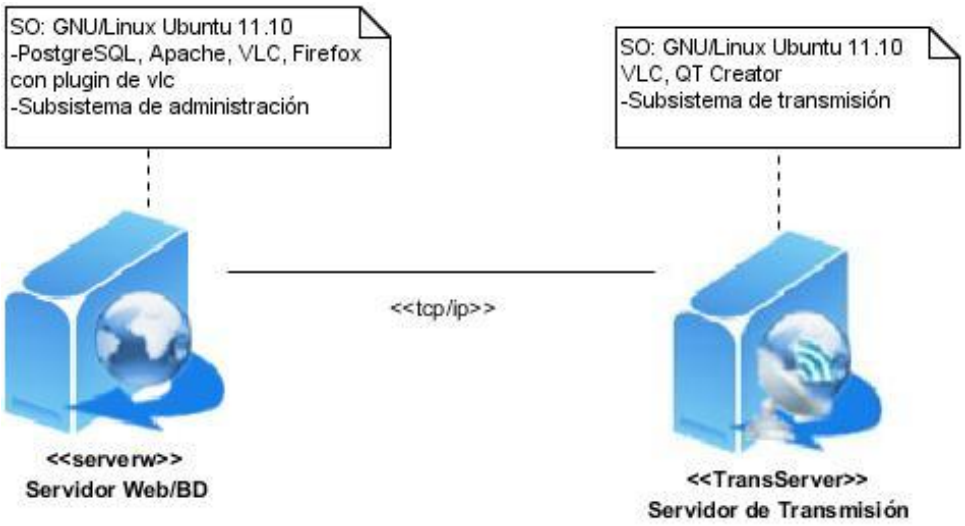


Figura 8: Diagrama de despliegue.

3.8 Conclusiones

- Al aplicar los diferentes patrones de diseño se elaboró un diagrama de clases flexible y reutilizable.
- El diagrama de clases del diseño sirvió para mostrar la realización física de los requisitos funcionales y no funcionales. Mediante este diagrama se definieron los principales métodos y atributos a desarrollar durante el proceso de implementación del sistema.
- A partir de las clases persistentes se construyó el modelo de datos para describir la información que será almacenada. Estas clases serán usadas posteriormente en la creación de la base de datos a utilizar en la implementación del sistema.
- La realización del diagrama de despliegue sirvió para modelar las relaciones físicas de los distintos nodos que componen el sistema.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA DEL SISTEMA

4.1 Introducción

Este capítulo contiene las principales características de la implementación del sistema. Muestra los principales componentes y sus relaciones, haciendo uso del diagrama de componentes. Además se desarrollan varias pruebas para conocer la calidad del producto, para verificar que los requisitos funcionales fueron cumplidos y para asegurar el correcto funcionamiento de la aplicación.

4.2 Modelo de componentes

El modelo de implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes. Describe cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y de modularización disponibles en el entorno de implementación y en el lenguaje de programación utilizados, y cómo dependen los componentes unos de otros. (26)

4.2.1 Diagrama de componentes

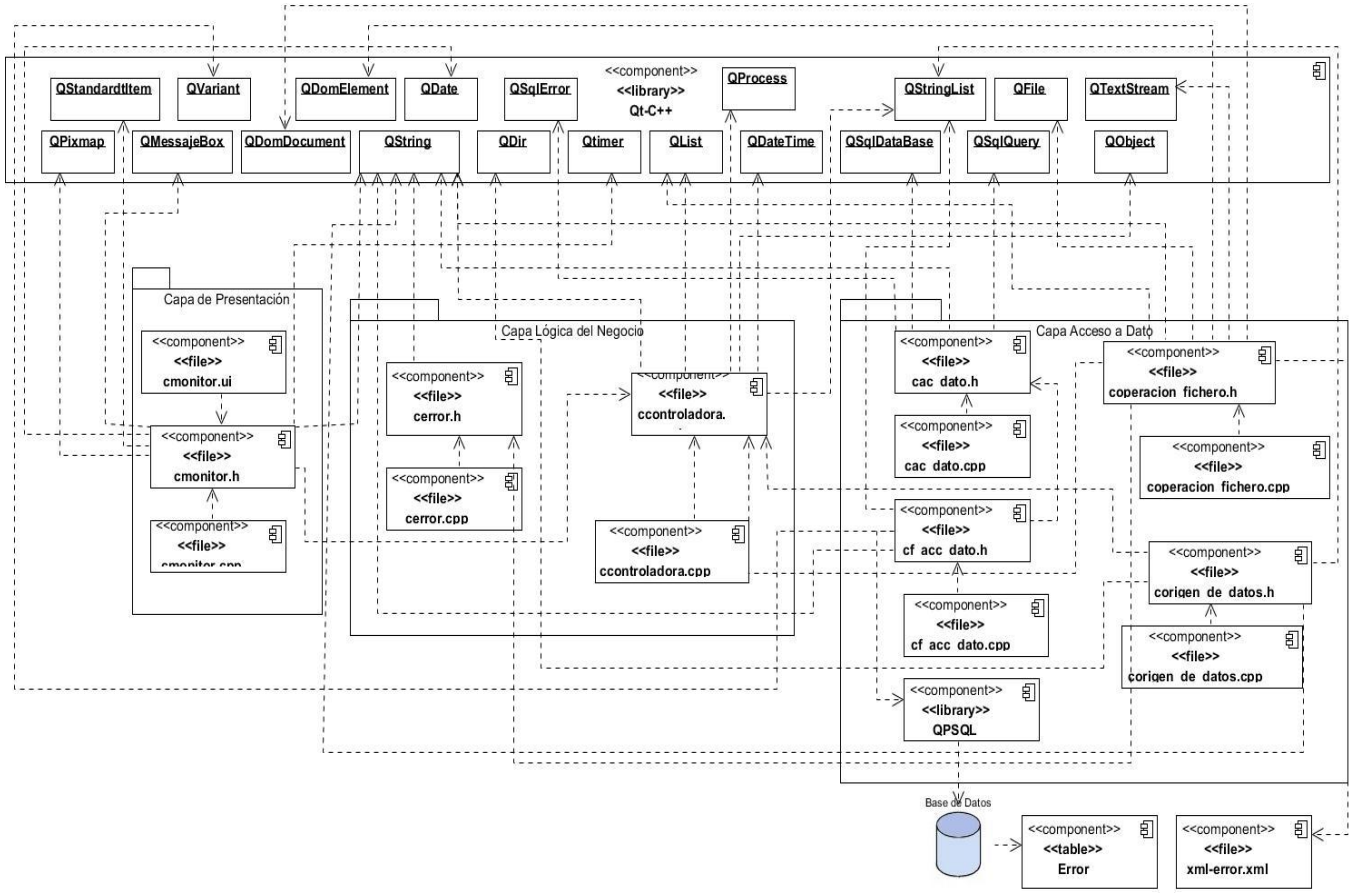


Figura 9: Diagrama de componentes.

4.3 Pruebas del sistema

En el desarrollo de cualquier software las pruebas son de vital importancia pues permiten verificar y revelar la calidad del mismo. Son utilizadas para identificar posibles fallos que pueden estar relacionados con la implementación, calidad o usabilidad del sistema. Cada prueba tiene su estrategia y propósito. Con el objetivo de validar las funcionalidades del sistema se realizarán pruebas funcionales para lograr buena calidad en el producto.

Se denominan pruebas funcionales a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para los cuales han sido creados. A este tipo de pruebas se les denomina también pruebas de comportamiento o pruebas de caja negra. Al realizar estas pruebas lo que se pretende es ponerse en los pies del usuario, usar el sistema como él lo usaría. (29)

El tipo de prueba que se va a realizar es el de caja negra y dentro de esta se utilizará el método de prueba de partición equivalente. Este método de prueba de caja negra divide el dominio de entrada de un programa en clases de datos, a partir de las cuales deriva los casos de prueba. Cada una de estas clases de equivalencia representa a un conjunto de estados válidos o inválidos. Se identificaron un conjunto de casos de pruebas que permiten detallar la forma en la que se va a validar las funcionalidades y algoritmos utilizados para el desarrollo del sistema. Es un conjunto de entradas de pruebas, condiciones de ejecución y resultados esperados, desarrollados para cumplir un objetivo en particular o una función.

4.3.1 Casos de prueba

A continuación se muestran las pruebas realizadas a los casos de uso significativos del sistema:

Caso de uso: Identificar error de base de datos.

Descripción General.

Comienza cuando el actor reloj inicializa el chequeo de la conexión con la base de datos en búsqueda de algún error. El caso de uso finaliza cuando termina la comprobación mencionada anteriormente y retorna si existe o no conexión.

Condiciones de ejecución.

Solo se ejecutará esta acción cuando el reloj inicialice la comprobación de la conexión con la base de datos.

Secciones a probar en el Caso de Uso.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Identificar error de conexión con la base de datos.	EC 1.1: Se identifica error de conexión con la base de datos.	Se comprueba la conexión con la base de datos y se identifica error con la misma. Se detalla el error poniendo la fecha y hora en que ocurre, el tipo de error y la descripción y se guarda en un fichero local. Luego el sistema muestra un mensaje indicando que no existe conexión con base de datos.
	EC 1.2: No se identifica error de conexión con la base de datos.	Se comprueba la conexión con la base de datos y no se identifica error con la misma. Se comprueba si existe el fichero local, en el caso de existir se suben los errores contenidos en él a la base de datos y luego se elimina. Luego el sistema muestra un mensaje indicando que existe conexión con base de datos.

Tabla 18: Secciones a probar en el CU Identificar error de base de datos.

Descripción de variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	conexión	boolean	No	Permite conocer si hay conexión o no con la base de datos. Puede tomar valor true o false.

Tabla 19: Descripción de variables del CU Identificar error de base de datos.

Matriz de Datos.

SC 1: Identificar error de conexión con la base de datos.

Escenario	Variable 1 conexión	Respuesta del Sistema	Resultado de la Prueba
EC 1.1: Se identifica error de conexión con la base de datos.	V/false	El sistema muestra un mensaje indicando que no existe conexión con la base de datos.	Satisfactorio
EC 1.2: No se identifica error de conexión con la base de datos.	V/true	El sistema muestra un mensaje indicando que existe conexión con la base de datos.	Satisfactorio

Tabla 20: Matriz de datos de la SC1 Identificar error de base de datos.

Caso de uso: Identificar error de orígenes de datos.

Descripción General.

Comienza cuando el actor reloj inicializa el chequeo de la existencia de los directorios de los orígenes de datos. El caso de uso finaliza cuando termina chequeo mencionado anteriormente y retorna si están disponibles o no los diferentes directorios.

Condiciones de ejecución.

Solo se ejecutará esta acción cuando el reloj inicialice el chequeo de la existencia de los orígenes de datos.

Secciones a probar en el Caso de Uso.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Identificar error de orígenes	EC 1.1: Se identifica error de orígenes de datos.	Se comprueba la existencia del directorio de los videos, imágenes, música y los logos

de datos.		a los cuales accede el subsistema de transmisión en la ejecución del canal y existe error de disponibilidad con uno o más. Se detalla el mismo poniendo la fecha y hora en que ocurre, el tipo de error y la descripción y se guarda en base de datos si hay conexión, de lo contrario se guarda en un fichero local. Luego el sistema muestra un mensaje indicando cuales directorios no están disponibles.
	EC 1.2: No se identifica error de orígenes de datos.	Se comprueba la existencia del directorio de los videos, imágenes, música y los logos a los cuales accede el subsistema de transmisión en la ejecución del canal y no existe error de disponibilidad con estos. Luego el sistema muestra un mensaje indicando que los directorios se encuentran disponibles.

Tabla 21: Secciones a probar en el CU Identificar error de orígenes de datos.

Descripción de variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	directorio	boolean	No	Permite conocer si los diferentes directorios están o no disponibles. Puede tomar valor true o false.

Tabla 22: Descripción de variables del CU Identificar error de orígenes de datos.

Matriz de Datos.

SC 1: Identificar error de orígenes de datos.

Escenario	Variable 1 directorio	Respuesta del Sistema	Resultado de la Prueba
EC 1.1: Se identifica error de orígenes de datos.	V/false	El sistema muestra un mensaje indicando cuales directorios no están disponibles.	Satisfactorio
EC 1.2: No se identifica error de orígenes de datos.	V/true	El sistema muestra un mensaje indicando que los directorios se encuentran disponibles.	Satisfactorio

Tabla 23: Matriz de datos de la SC1 Identificar error de orígenes de datos.

Caso de uso: Identificar error de existencia de medias.

Descripción General.

Comienza cuando el actor reloj inicializa la comprobación de la coincidencia de las medias que están guardadas en la base de datos con las que están físicamente en el servidor de medias. El caso de uso finaliza cuando termina la comprobación mencionada anteriormente y retorna si existe error o no con la existencia de las medias.

Condiciones de ejecución.

Solo se ejecutará esta acción cuando el reloj inicialice la comprobación de la existencia de medias.

Secciones a probar en el Caso de Uso.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Identificar error de existencia	EC 1.1 Se identifica error de existencia de medias.	Se comprueba la coincidencia de las medias que están guardadas en la base de

de medias.		datos con las que están físicamente en el servidor de medias y existe error al no hallarse una o varias medias. Se detalla el mismo poniendo la fecha y hora en que ocurre, el tipo de error y la descripción y se guarda en base de datos si hay conexión, de lo contrario se guarda en un fichero local. Luego el sistema muestra un mensaje indicando que las medias no se encuentran físicamente y un botón para visualizar el nombre del fichero de las mismas.
	EC 1.2: No se identifica error de existencia de medias.	Se comprueba la coincidencia de las medias que están guardadas en la base de datos con las que están físicamente en el servidor de medias y no existe error. Luego el sistema muestra un mensaje indicando que las medias se encuentran físicamente.

Tabla 24: Secciones a probar en el CU Identificar error de existencia de medias**.**

Descripción de variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	existe	boolean	No	Permite conocer si las diferentes medias en base de datos están o no físicamente en el servidor de medias. Puede tomar valor true o false.

Tabla 25: Descripción de variables del CU Identificar error de existencia de medias.

Matriz de Datos.

SC 1: Identificar error de existencia de medias.

Escenario	Variable 1 directorio	Respuesta del Sistema	Resultado de la Prueba
EC 1.1 Se identifica error de existencia de medias.	V/false	El sistema muestra un mensaje indicando que las medias no se encuentran físicamente y un botón para visualizar el nombre del fichero de las mismas.	Satisfactorio
EC 1.2: No se identifica error de existencia de medias.	V/true	El sistema muestra un mensaje indicando que las medias se encuentran físicamente	Satisfactorio

Tabla 26: Matriz de datos de la SC1 Identificar error de existencia de medias.

Caso de uso: Identificar error de consumo.

Descripción General.

Comienza cuando el actor reloj inicializa la comprobación de error de consumo de RAM y CPU de la aplicación de transmisión. El caso de uso finaliza cuando termina la comprobación mencionada anteriormente y retorna si está alto o normal el consumo.

Condiciones de ejecución.

Solo se ejecutará esta acción cuando el reloj inicialice la comprobación de error de consumo de RAM y CPU de la aplicación de transmisión

Secciones a probar en el Caso de Uso.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC 1: Identificar	EC 1.1 Se identifica error	Se comprueba que el consumo de RAM y

error de consumo.	de consumo.	CPU de la aplicación de transmisión no sobrepasan los porcientos especificados existiendo error al estar alto uno o ambos. Se detalla el mismo poniendo la fecha y hora en que ocurre, el tipo de error y la descripción y se guarda en base de datos si hay conexión, de lo contrario se guarda en un fichero local. Luego el sistema muestra un mensaje indicando que el consumo es alto.
	EC 1.2: No se identifica error de consumo.	Se comprueba que el consumo de RAM y CPU de la aplicación de transmisión no sobrepasan los porcientos especificados existiendo error al estar alto uno o ambos. Luego el sistema muestra un mensaje indicando que el consumo es normal.

Tabla 27: Secciones a probar en el CU Identificar error de consumo.

Descripción de variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	alto	boolean	No	Permite conocer si el consumo de RAM y CPU de la aplicación de transmisión es alto o no. Puede tomar valor true o false.

Tabla 28: Descripción de variables del CU Identificar error de consumo.

Matriz de Datos.

SC 1: Identificar error de consumo.

Escenario	Variable 1 alto	Respuesta del Sistema	Resultado de la Prueba
EC 1.1 Se identifica error de consumo.	V/false	El sistema muestra un mensaje indicando que el consumo se encuentra alto.	Satisfactorio
EC 1.2: No se identifica error de consumo.	V/true	El sistema muestra un mensaje indicando que el consumo es normal.	Satisfactorio

Tabla 29: Matriz de datos de la SC1 Identificar error de consumo.

El caso de prueba Identificar error del XML puede ser consultado en el anexo 1.

Al realizar la primera iteración de las pruebas de caja negra se arroja como resultado un total de 15 no conformidades de las cuales 5 se clasifican como significativas, relacionadas problemas de funcionalidad de la aplicación. Las restantes 10 no conformidades son de tipo no significativas y son detectadas principalmente por problemas ortográficos o de redacción. En la segunda iteración de pruebas se reducen las no conformidades a un total de 5, siendo 2 de estas de tipo no significativa relacionada con problemas de ortografía y 3 significativas pertenecientes a funcionalidades del sistema. En la tercera iteración de pruebas no son detectadas no conformidades.

El resultado de las pruebas realizadas al software fue satisfactorio. Se evidenció que las respuestas del sistema son las esperadas y que el mismo cumple con los requisitos funcionales y a su vez coinciden con las descripciones de los casos de uso. Con las pruebas aplicadas se comprobó que el sistema cumple con lo objetivos propuestos.

4.4 Conclusiones parciales

- El modelo de componente realizado ayudó a entender la distribución física de los mismos y la dependencia entre ellos. También permitió tener una visión de la estructura del código fuente del sistema.

- Al realizar las pruebas al software se arroja como resultado que la aplicación cumple de manera satisfactoria con las funcionalidades identificadas durante el proceso de desarrollo del software. Con las iteraciones de las pruebas se logra refinar las no conformidades encontradas, permitiendo que el sistema cuente con la calidad requerida.

CONCLUSIONES GENERALES

Con la culminación del desarrollo del Módulo de Gestión de Errores para el subsistema de transmisión de la Plataforma de Televisión Informativa PRIMICIA se pudo arribar a las siguientes conclusiones:

- Con el análisis del objeto de estudio se identificaron los posibles tipos de errores que impiden el buen funcionamiento del canal informativo.
- Se realizó una estrategia para la detección y almacenamiento en base de datos de errores ocurridos durante la transmisión del canal informativo.
- Mediante el uso de la metodología de desarrollo de software RUP se generaron los artefactos y la documentación necesaria para llevar a cabo el desarrollo del módulo; estos servirán como base de futuras actualizaciones al mismo.
- Con la aplicación de los casos de pruebas sobre las funcionalidades se logró validar la solución, comprobándose la calidad del software pues se obtuvo resultados positivos.
- Se logró solucionar el problema de la investigación logrando el desarrollo del Módulo de Gestión de Errores para el subsistema de transmisión de la Plataforma de Televisión Informativa PRIMICIA. (Ver anexo 2).

RECOMENDACIONES

- Dar seguimiento a la aplicación de transmisión con el objetivo de detectar si surgen nuevos tipos de errores que causen el mal funcionamiento del canal informativo.
- A futuras actualizaciones del módulo se le pudiera agregar la funcionalidad de avisar a los administradores mediante correo electrónico que ha ocurrido un error.

REFERENCIAS BIBLIOGRÁFICAS

1. Jack, Keith. Dictionary of Video and Television Technology. United State of America : s.n.

2. Definición de . [Online] [Cited: 14 11 , 2010.] http://definicion.de/sonido.

3. Diccionario Manual de la Lengua Española. s.l. : Larousse Editorial, 2007.

4. Definición.de. [Online] [Cited: 11 21 , 2010.] http://definicion.de/television/.2007.

5. PRIMICIA, Plataforma de Televisión Informativa. UCI, Ciudad de La Habana : s.n.

6. Alvarez, Elaine Morales. Módulo de Gestión de Errores de la Plataforma de Televisión Informativa PRIMICIA. UCI, Ciudad de la Habana : s.n., 2010.

7. Pérez, Iván Nieto. El código. [Online] [Cited: 11 21, 2010.] http://www.elcodigo.net/tutoriales/diccionario.html.

8. Definición.de. [Online] [Cited: 11 2010, 11.] http://definicion.de/error/.2007.

9. Calzado, Frank Benitez. Procedimiento para el seguimiento y tratamiento de los errores de PRIMICIA. UCI,Ciudad de la Habana : s.n., 2002.

10. Plataforma Communi. TV. Gestión de contenidos multicanal y servicios interactivos. 2006.

11. Larman, Craig. UML y Patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado. s.l. : PEARSON EDUCACION, 2002.

12. freedownloadmanager. [Online] [Cited: 11 2010, 15.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(Iglesia_Anglicana)_% 5BMac_OS_X_cuenta_14717_p.

13. Blog – Ramip. [Online] 8 18, 2010. [Cited: 11 25, 2010.] http://www.ramip.net/lenguajes-de-programacion/que-son-los-lenguajes-de-programacion.htm.

14. Millán, A. J. Zator Systems. [Online] [Cited: 11 15, 2010.] http://www.zator.com/Cpp/E1_2.htm.

15. TecHerald.com. [Online] 03 23, 2009. [Cited: 11 15, 2010.] http://techerald.com/post.view?que-es-qt-ejemplo-clasico-hola-mundo--230320090143.html.

16. Software Libre. [Online] [Cited: 11 25, 2010.] http://www.softwarelibre.ec/site/index.php?option=com_content&view=article&id=198&Itemid=165.

17. Ricardovs. Geeks & Linux Atelie. [Online] 05 15, 2009. [Cited: 11 15, 2010.] http://www.glatelier.org/2009/05/qt-creator-desarrollando-aplicaciones-rapidamente/.

18. desarrolloweb.com. [Online] 07 31, 2007. [Cited: 11 15, 2010.]] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.

19. Pecos, Daniel. Daniel Pecos. [Online] [Cited: 11 25, 2010.] http://danielpecos.com/docs/mysql_postgres/x15.html.

20. w3support. [Online] [Cited: 01 18, 2011.] http://es.w3support.net/index.php?db=so&id=296150.

21. Logging Services. [Online] [Cited: 01 18, 2011.] http://logging.apache.org/log4cxx/index.html .

22. varelaenred. [Online] [Cited: 01 28, 2011.] http://www.varelaenred.com.ar/explicacion%20registro%20de%20win.htm.

23. estrellateyarde. [Online] [Cited: 01 28, 2011.] http://www.estrellateyarde.org/so/logs-en-linux.

24. support kaspersky. [Online] [Cited: 01 28, 2011.] http://support.kaspersky.com/sp/sos6mp4/error?qid=208280747.

25. idt. [Online] 04 06, 2009. [Cited: 01 28, 2011.] http://www.idt.es/cddocal/web/agenteaplicacion.htm.

26. . Jacobson, Ivar , Booch, Grady y Rumbaugh, James. El Proceso Unificado de Desarrollo de Software. Madrid : Pearson Educación. S.A, 2000.

27. Slideshare. [Online] [Cited: 05 10, 2011.] http://www.slideshare.net/Décimo/arquitectura-3-capas.

28. Saavedra, Jorge. [Online] 05 07, 2007. [Cited: 02 25, 2011.] http://jorgesaavedra.wordpress.com/category/patrones-grasp/.

29. B, Ing. Alexander Oré. CalidadySoftware.com. [Online] 2009. [Cited: 05 28, 2012.] http://www.calidadysoftware.com/testing/pruebas_funcionales.php.