**Universidad de las Ciencias Informáticas**

**Facultad 6**

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

# Módulo para el Monitoreo de recursos y la Gestión de los errores en PRIMICIA

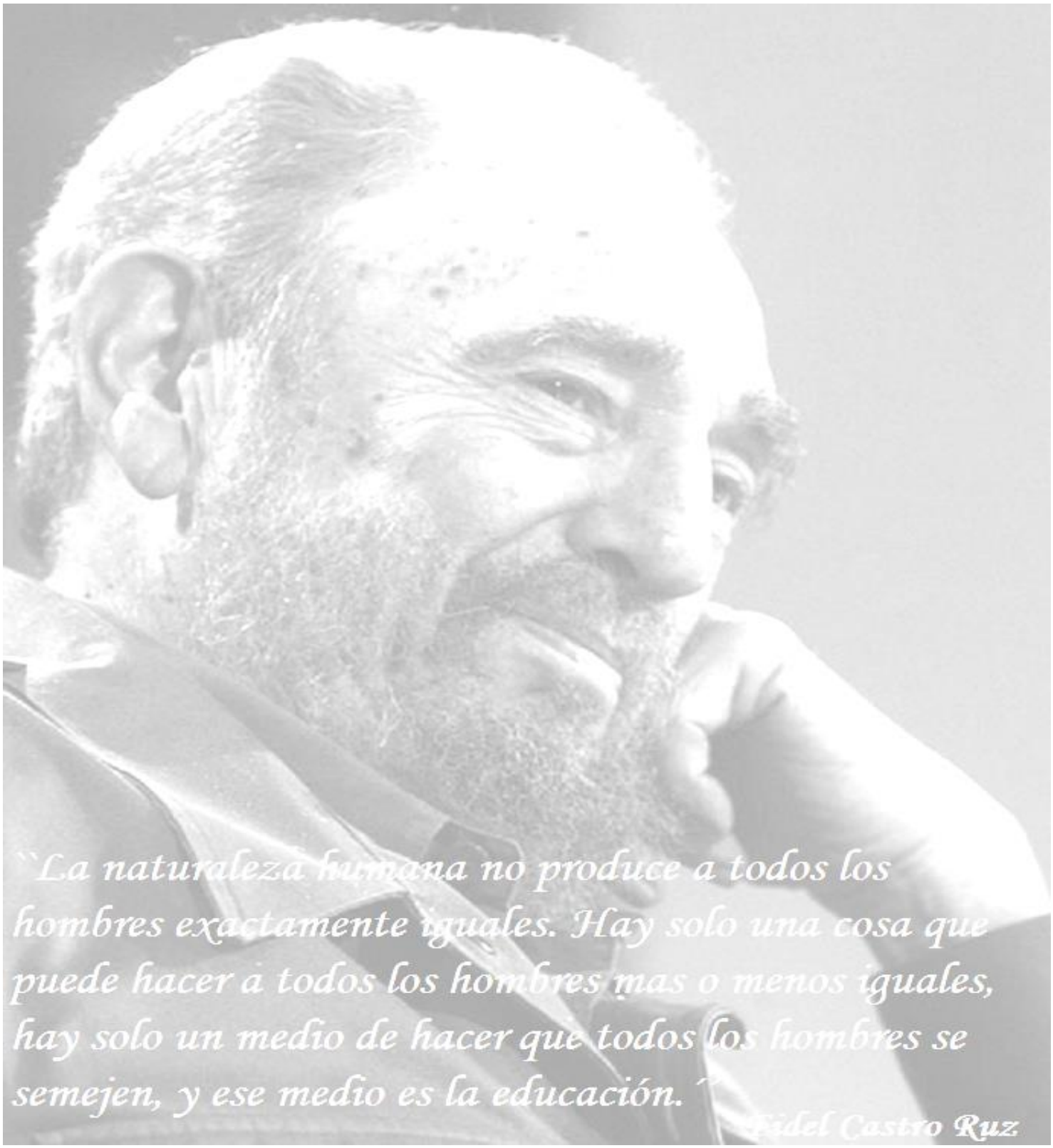
***Autor:*** *Osmani Martínez López*

***Tutores:*** *Ing. Yanary Hernández Sosa*

*Ing. Yoilan Rios Rosales*

*La Habana, junio de 2016*

## *Pensamiento*



## Dedicatoria

Esencialmente a mis padres que me apoyaron desde el inicio de la carrera dándome todo lo que estaba a su alcance para que yo me sintiera a gusto. A mis hermanos que siempre me dieron todo lo que le pedí y hasta lo que no, a todas mis amistades en general. A todos los que confiaron en mí y me apoyaron siempre para la realización de este, mi mayor triunfo.

# *Agradecimientos*

Por contribuir en la formación de la persona que soy hoy, le doy las gracias a:

Mi familia en general, especialmente a la mujer y el hombre que más amo en este mundo, ”**mis padres**”, por haber hecho de mi la persona que hoy soy; por consagrar su vida a mí en los momentos más difíciles de mi vida, por hacer hasta lo imposible para satisfacer mis antojos, Por aguantarme tanta malacrianza desde champolito, en fin, por ser quienes son.

Los **amigos** del PRE y mis vecinos; Luisito Nelson, Arturo, Dalita, Yary, Carlos, Yoito, Roxana y la nueva integrante del grupo “Melisa” Todos ellos a pesar de la distancia mantienen comunicación conmigo, y a pesar de llevar varios años por distintos caminos, cada vez que nos encontramos se evidencia una vez más que lo único que nos falta para ser familia es ser de la misma sangre.

También a los míos de MOA que conocí en la UCI, en especial a Dairon, que tuve que esperarlo otro curso más porque sin mi está perdido aquí en la universidad, yo me quedé para repasarlo, y para darle clases de FIFA; al Niche, que me mantuvo como si fuera mi padre. Siempre se lo dije, y le estoy agradecido por eso y por ser tan buen chamaco, a Jose “el pillo”, que fue otro ejemplo a seguir de lo que es tener buena conducta en la UCI, a mi gente del Palenque Record´s y los Supper Niggers y a la banda del hacha: Yoyo, Aliu etc.., a mis amigas que a todas las quiero por igual y doy las gracias a dios por haberlas conocido.

Los **profesores** y **directivos** de la UCI que conocí en el transcurso de la carrera, es importante destacar a Yanelis Benítez, gracias por ser como eres, y al Jim que es el mejor profesor que puede tener un estudiante.

Mis **tutores**, que a pesar de yo ser desorientado e incumplidor, me ayudaron sin molestarse cada vez que fui con alguna pregunta o a buscar lo que necesitaba.

Las **piedras** que encontré en el camino, que me hicieron valorar mis actitudes desde otro punto de vista, y que me enseñaron a no conformarme con la victoria de una batalla, sino que es necesario desear más allá del triunfo.

## DECLARACIÓN DE AUTORÍA

Declaro ser el único autor de la presente investigación que tiene por título: Módulo para el Monitoreo de recursos y la Gestión de los errores en PRIMICIA y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

**Autor**

_____

Osmani Martínez López

**Tutores**

_____ _____

Ing. Yanary Hernández Sosa Ing. Yoilan Rios Rosales

## DATOS DE CONTACTO

Tutores:

Yanary Hernández Sosa

Correo electrónico: yanary@uci.cu

Año de graduación: 2012

Yoilan Rios Rosales

Correo electrónico: yoilan@uci.cu

Año de graduación: 2014

Institución: Universidad de las Ciencias Informáticas.

Dirección de la institución: Carretera a San Antonio de los Baños, Km. 2 1⁄2, Reparto: Torrens, Municipio: Boyeros, Provincia: La Habana.

## Resumen

Producto de los diferentes resultados obtenidos de las investigaciones realizadas para el desarrollo de la Plataforma de Televisión Informativa surgió PRIMICIA. PRIMICIA es un producto informático que da respuesta a la transmisión automatizada de televisión informativa. En el transcurso de los años de transmisiones se han venido generando diferentes errores que han impedido que el funcionamiento de PRIMICIA tenga la calidad requerida, ya que no es identificada en el momento la causa que lo provocó y no se lleva un control del porciento de consumo de algunos procesos. El presente trabajo constituye una solución para el monitoreo de esos recursos y la detección, clasificación y almacenamiento de errores en la plataforma PRIMICIA. En éste se exponen las herramientas, metodologías, lenguaje de programación y framework que se llevaron a cabo para el desarrollo del prototipo funcional para el monitoreo de recursos y la gestión de los errores en PRIMICIA. Además se detectan los problemas más críticos que afectan el buen funcionamiento del sistema y traza una estrategia para el envío de alertas al administrador cada vez que ocurra alguna de estas fallas.

**Palabras clave:** errores, módulo, PRIMICIA.

# *Abstract*

## Abstract:

Product of the different results of research conducted to develop Informative TV Platform emerged PRIMICIA. PRIMICIA is a software product that responds to the automated transmission of television information.During the years of broadcasting have been generating various errors that have prevented PRIMICIA operation has the required quality because it is not identified at the time the cause that caused it, and not have a percent of consumption control of some processes. This work is a solution for monitoring those resources and the detection, classification and storage of errors PRIMICIA platform. In this document exposed tools, methodologies, programming language and framework that set used for the development of functional prototype for resource monitoring and management errors PRIMICIA. Also they detected most critical problems affecting the good performance of system and it draw a strategy for sending alerts to the administrator for each occurrence of these faults.

**Keywords:** Errors, Module, PRIMICIA.

# Índice de Contenido

Introducción ..... 1

Capítulo 1. Fundamentación Teórica ..... 6

1.1 Introducción ..... 6

1.2 Conceptos asociados al dominio del problema ..... 6

Proceso de Administración ..... 6

Transmisión ..... 6

Error ..... 6

Base de datos ..... 7

Registro de traza ..... 8

1.4 Objeto de estudio ..... 8

1.5 Análisis de soluciones existentes ..... 9

1.5.1 Soluciones existentes a nivel internacional ..... 10

1.5.2 Soluciones existentes a nivel nacional ..... 11

1.6 Estrategias para la detección de errores ..... 12

Logs de errores en Windows 7 ..... 12

1.7 Propuesta de solución ..... 12

SmtpClient ..... 13

1.8 Metodología ..... 14

RUP ..... 14

1.9 Lenguaje Unificado de Modelado (UML) ..... 17

1.10 Tecnologías ..... 17

1.10.1 Visual Paradigm para UML ..... 18

1.11 Lenguajes de programación ..... 18

C++ ..... 18

1.12 Framework o marco de trabajo ..... 19

QT ..... 19

# Índice de Contenido

1.13 Entorno de Desarrollo Integrado (IDE)....19
QT Creator 3.1.1....19
1.14 Sistema Gestor de Base de *Datos*....20
PostgreSQL....20
Conclusiones del capítulo....20
Capítulo 2: Análisis y Diseño de la Solución Propuesta....21
2.2 Modelo del dominio....21
2.2.1 Conceptos más relevantes del diagrama del modelo de dominio.....22
2.3 Requisitos del sistema....23
2.3.1 Requisitos funcionales.....23
2.3.2 Requisitos no funcionales.....25
2.4 Modelo de Casos de Uso....26
Descripción textual de los Casos de Uso....27
2.5 Modelo de Análisis....32
2.6 Arquitectura del sistema....32
2.8 Patrones de diseño....33
2.8.1 Patrones GRASP utilizados....34
2.8.2 Patrones GOF utilizados....35
2.9 Modelo de diseño....35
2.10 Diagrama de clases persistentes.....36
2.11 Conclusiones del capítulo....37
Capítulo 3: Implementación y Prueba de la Solución Propuesta....38
3.1 Introducción....38
3.2 Modelo de Implementación....38
3.4 Diagrama de componentes....38
3.3 Diagrama de Despliegue....39

# Índice de Contenido

3.6 Pruebas del sistema ..... 40

3.7 Casos de prueba ..... 41

Conclusiones del capítulo ..... 45

CONCLUSIONES ..... 46

RECOMENDACIONES ..... 47

Referencias ..... 48

# Índice de Tablas

Índice de Tablas
Requisitos no funcionales de hardware.....25
Actores del Sistema. .....26
Descripción del CU Cargar Fichero.....27
Descripción del CU Listar error. .....28
Descripción del CU Configurar recursos. .....29
Descripción del CU Detectar error. .....30
Descripción del CU Enviar Alerta. .....31
Escenarios CUS Detectar error. .....41
Variables CUS Detectar error.....42
Matriz de datos de la SC 1 Detectar error. .....42
Escenarios CUS Configurar recurso. .....43
Variables CUS Configurar recurso. .....43
Matriz de datos de la SC 1 Configurar recurso.....44

# Índice de Figuras

Características de RUP.....15

Proceso de desarrollo Iterativo de la Metodología RUP. .....16

Diagrama del modelo de dominio.....22

Diagrama de Casos de Uso del Sistema.....27

Arquitectura en capas (3 capas). .....33

Diagrama de clases del diseño .....36

Diagrama de clases persistentes .....37

Diagrama de componentes .....39

Diagrama de despliegue .....40

Interpretación gráfica de Caja negra .....41

Pruebas de Iteración.....44

# Índice de Figuras

## Introducción

Los medios de comunicación visual surgieron con el objetivo de satisfacer las necesidades existentes entre los miembros de la sociedad con respecto a la carencia de obtención de información. La televisión, desde sus primeras emisiones se destacó como uno de los más importantes por la capacidad que tiene de hacer llegar la información a diferentes partes del mundo de manera simultánea. Con su progresiva evolución permite a la humanidad la visualización de imágenes y la escucha de sonidos asociados a estas, los cuales abordan sobre sucesos acontecidos o que están aconteciendo en ese instante. Su integración con la informática y las telecomunicaciones dio lugar a grandes avances en el mundo de las noticias, ya que estas tecnologías garantizaron en la televisión una mayor capacidad y explotación de los medios de comunicación.

A pesar de estos progresos, la televisión aún cuenta con algunos problemas a la hora de la emisión, porque durante el proceso de transmisión o administración de la información ocurren algunas fallas que provocan que la recepción por parte de la comunidad sea menos satisfactoria.

Hoy en día se ha hecho muy popular internacionalmente el uso de canales corporativos con el fin de mantener informados a un determinado público de las principales noticias de una empresa específica. Cuba no se ha quedado fuera de lo que acontece en el mundo sobre el desarrollo de tecnologías en el campo de la transmisión de señales digitales. Sectores como la salud, la educación, servicios de hotelería u organismos que tengan una red televisiva se benefician de las ventajas que brindan dichas transmisiones.

La Universidad de las Ciencias Informáticas (UCI) desde su creación en el 2002 ha constituido una entidad clave en la aplicación de estas tecnologías en el país. Actualmente cuenta con un Centro de desarrollo de Geoinformática y Señales Digitales (GEYSED) ubicado en la Facultad 6, donde se desarrollan sistemas informáticos con este propósito. Uno de ellos es la Plataforma de Televisión Informativa PRIMICIA, la cual surge por la necesidad de mantener a clientes o trabajadores informados de una manera clara, precisa y oportuna. Esta aplicación es capaz de proveer un canal televisivo para la transmisión automática y constante de informaciones en distintos formatos como: texto, imagen, audio y video a través de una red de televisión. El sistema mantiene un constante desarrollo concentrándose en mejorar sus funcionalidades y en buscar otras nuevas que le permitan introducirse en el mercado internacional que cuenta de gran competitividad.

Con el paso de los años han creado varias aplicaciones informáticas que tienen un objetivo similar al de PRIMICIA, a pesar de que cada una de ellas es una mejora adaptada al ámbito donde se aplica. Entre ellas se encuentra

# *Introducción*

Señal 3 desarrollada 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, surgió TVEnergía, a solicitud del Ministerio de Energía y Petróleo de Venezuela (MENPET), a la que hubo que adaptarle funcionalidades acordes a las necesidades del cliente pues querían que estuviera desarrollada con herramientas libres.

Finalmente se crea la Plataforma PRIMICIA la cual está concebida para transmitir información de manera constante e inmediata, puede ser utilizada para brindar información sobre los vuelos en los aeropuertos, o en terminales de ómnibus para mostrar información relacionada con los pasajeros, o en cualquier institución que necesite brindar servicios relacionados con las información de los viajes, de los transitorios o noticias de relevancia tanto nacional como internacional.

Para su funcionamiento, PRIMICIA cuenta con dos subsistemas que están estrechamente relacionados, el de Administración donde se gestionan los usuarios del sistema, las secciones temáticas, las infocintas, la redacción de noticias, el almacenamiento y reproducción de los recursos multimedia así como los cambios entre las señales de canales televisivos. En cambio, el de Transmisión genera una cartelera, visualiza las noticias y las infocintas, reproduce un fondo musical y visualiza las señales de canales televisivos en vivo.

Las versiones anteriores realizadas a PRIMICIA contenían errores en la transmisión del canal a causa de fallas tanto internas como externas; ejemplo de estas lo constituye la falta de conexión a la base de datos o al servidor de medias, teniendo como resultado la interrupción del proceso de visualización de las noticias o el mal funcionamiento del canal. Otro de los errores que presentan estas versiones es que se detenga el servició de replicación. Este servicio se encarga de enviar todos los datos al servidor de transmisión para que estos sean utilizados y transmitidos. Si el volumen de los datos crece rápidamente y existe una gran concurrencia de transacciones en el sistema lo mejor sería hacer uso de la replicación. Mediante esta acción se mantiene actualizado el servidor de transmisión en cuanto a las noticias, los videos u otros materiales nuevos que se adicionen.

Un fallo en esta operación trae graves afectaciones en el sistema, no se actualiza el servidor de transmisión, el cual estará mostrando las mismas noticias continuamente, y en el peor de los casos se cae la transmisión.

# *Introducción*

Una versión mejorada de PRIMICIA es la 2.0, pero también trae consigo errores en la transmisión del canal ya que presenta errores de consumo de recursos en el funcionamiento del canal; este error se presenta cuando el consumo de los recursos de RAM y CPU de la aplicación sobrepasa un determinado valor de RAM o CPU definidos por el administrador como los valores normales entre los que se debe mantener la aplicación. Además cuenta con errores de inexistencia de medias provocados cuando los videos guardados en la base de datos no coinciden con los físicos en el servidor de medias, teniendo como resultado que en ocasiones no se escuche el fondo musical o no se visualice algún video o imagen asociado a la noticia. También se les hace difícil a los responsables de la administración de dicha plataforma obtener información específica de los procesos y recursos que se manejan en la misma y necesitan tener controlada la información sobre el rendimiento del servidor en el cual se encuentra desplegada la aplicación. De los problemas con los errores que ocurren en el Subsistema de Administración, solo es posible ver su efecto y no lo que lo origina, lo que impide llevar a cabo acciones correctivas.

Además esta versión actual tampoco cuenta con un mecanismo de control que alerte al administrador cada vez que ocurra una de estas fallas en el funcionamiento del canal por lo que se hace necesario el desarrollo de una estrategia que permita solucionar estos problemas.

Dada la situación problemática expuesta anteriormente, el presente trabajo de diploma plantea como **problema de la investigación**: en la plataforma PRIMICIA los procesos de transmisión y administración generan errores a la hora de emitir la información provocando el mal funcionamiento del canal definiéndose como **objeto de estudio** la monitorización de los recursos y la gestión de errores, presentando como **campo de acción** la monitorización de los recursos y la gestión de los errores utilizando tecnologías de escritorio.

Para solucionar el problema planteado se define como **objetivo general** desarrollar un módulo que permita monitorizar los recursos y la gestión de errores en PRIMICIA.

Para guiar la investigación se definen las siguientes **preguntas de investigación:**

1. ¿Cuáles son los fundamentos teórico-metodológicos para desarrollar un módulo para el monitoreo de recursos y la gestión de los errores en plataformas de televisión informativas?
2. ¿Cuáles son las similitudes del proceso de gestión de errores asociadas a la Plataforma de Televisión Informativa PRIMICIA?
3. ¿Qué se conoce sobre el consumo de los recursos en la Plataforma de Televisión Informativa PRIMICIA?

4. ¿Cómo organizar el proceso de desarrollo del Módulo para el Monitoreo de recursos y la Gestión de los errores asociados a la Plataforma de Televisión Informativa PRIMICIA?

En función del cumplimiento del problema y para dar solución al objetivo general se plantean las siguientes **tareas para la investigación:**

1. Caracterización de soluciones existentes que permiten la gestión de errores y el monitoreo de recursos.
2. Caracterización de los posibles errores que puedan ocurrir en la Plataforma de Televisión Informativa PRIMICIA.
3. Definición y caracterización de las herramientas y metodología a utilizar durante el desarrollo del sistema.
4. Definición de los requisitos funcionales y no funcionales del Módulo para el Monitoreo de recursos y la Gestión de errores en PRIMICIA.
5. Elaboración del Análisis y Diseño del sistema haciendo uso de una herramienta CASE.
6. Implementación del Módulo para el Monitoreo de recursos y la Gestión de errores en PRIMICIA.
7. Ejecución de las pruebas funcionales necesarias a la solución propuesta.

Al finalizar el trabajo de diploma se espera obtener como posibles resultados la documentación de la investigación realizada, así como los artefactos vinculados con la metodología de desarrollo de software utilizada y el código fuente de la aplicación.

## Métodos científicos de la investigación

Para lograr un adecuado desarrollo del objetivo planteado y para darle cumplimiento a las tareas de investigación se aplicaron varios métodos científicos existentes, tanto teóricos como empíricos, entre los cuales se encuentran los siguientes:

***Métodos teóricos***

Analítico–Sintético: Se utilizó para analizar las diferentes soluciones existentes tanto a nivel nacional como internacional que están relacionadas a la problemática planteada, permitiendo sintetizar lo fundamental para el desarrollo de la investigación.

Modelación: Se utilizó para crear una representación o modelo de los artefactos que se generaron a lo largo de todo el proceso de desarrollo del módulo para la monitorización de recursos y la gestión de los errores de PRIMICIA.

Inductivo-Deductivo: Se utilizó para hacer una generalización del comportamiento de los errores que ocurren durante la utilización del canal, esto confirmó las formulaciones teóricas y permitió comparar soluciones informáticas cuyos objetivos son similares a PRIMICIA. Además permitió deducir cual es la mejor solución para resolver el problema de la investigación.

***Métodos empíricos***

Observación Científica: La observación científica es la percepción planificada dirigida a un fin y relativamente prolongada de un hacho o fenómeno. Es el instrumento universal del científico, se realiza de forma consciente y orientada a un objetivo determinado (Hernández, 2002).

Para llevar a cabo este método se utilizó el tipo de observación externa, y se trazó una planificación de la observación donde se precisa parámetros como el tipo de observación, se define cuando se van a observar y los resultados que se esperan obtener al observar el comportamiento de los errores en PRIMICIA para la recopilación de información relacionada con los mismos.

## El presente trabajo de diploma se estructura de la siguiente manera:

**Capítulo 1 Fundamentos Teóricos de la Solución Propuesta**

Se realiza un estudio del estado del arte, abordando cómo se lleva a cabo la gestión de errores como proceso en plataformas de televisión informativas. Se definen los conceptos claves a tener en cuenta durante la investigación, constituyendo las bases teóricas bajo la cual se desarrolla la misma.

Además, se fundamenta la selección de la metodología de desarrollo, así como las herramientas y tecnologías a utilizar.

**Capítulo 2 Análisis y Diseño de la Solución Propuesta**

En este capítulo se definen las características del sistema, a partir de la descripción de los requisitos funcionales y no funcionales de la aplicación. Se realiza el diagrama y la descripción de los casos de uso del sistema. Además se representa la solución a partir de su diseño donde se especifican las responsabilidades de las clases y sus relaciones. Conjuntamente se exponen los elementos de la arquitectura: patrones de diseño y de arquitectura que se aplican.

**Capítulo 3 Implementación y Pruebas de la Solución Propuesta**

En este capítulo se modela la implementación del sistema a través del diagrama de componentes, se diseñan los casos de prueba y se ejecutan las pruebas para detectar y corregir los errores.

# Capítulo 1. Fundamentación Teórica

## 1.1 Introducción

Este capítulo comprende el estado del arte sobre el tema de investigación tratado y se profundiza en las tendencias nacionales e internacionales. Se establece la metodología y herramientas que se utilizan para dar solución al problema planteado y la justificación de por qué fueron escogidas.

## 1.2 Conceptos asociados al dominio del problema

***Proceso de Administración***

Proceso administración es el flujo continuo e interrelacionado de las actividades de planeación, organización, dirección y control, desarrolladas para lograr un objetivo común: aprovechar los recursos humanos, técnicos, materiales y de cualquier otro tipo, con los que cuenta la organización para hacerla efectiva, para la sociedad (GestioPolis, 2003).

El Proceso de Administración es el flujo continuo e interrelacionado de las actividades de planeación, organización, dirección y control, con el objetivo de aprovechar los recursos humanos, técnicos, materiales y de cualquier otro tipo con los que cuenta la organización para hacerla efectiva para la sociedad.

***Transmisión***

La transmisión de información tiene carácter unidireccional y unilateral, lo que altera la esencia interactiva del proceso comunicativo, o sea, el proceso comunicativo deja de ser tal si no garantiza interacción social, intercambios, interinfluencias, retroalimentación. Así, el proceso humano a través del cual se transmiten ideas, opiniones, puntos de vista de un emisor a un receptor, sin que medie la interacción social, la interinfluencia y la retroalimentación se considera **transmisión de información** (Navarro, y otros, 2012)**.**

De forma resumida se puede decir que la Transmisión es el proceso a través del cual se transmiten ideas, opiniones e información de un emisor a un receptor sin que medie la interacción social, la interinfluencia y la retroalimentación.

***Error***

De modo general es una creencia o estado o estado mental que no se ajusta a la realidad objetiva

Aplicado a la ley es una falla en la base, procedimiento, juicio o la ejecución de una acción juridicial, la redacción de un documento, o la grabación de un comunicado.

En estadísticas es la desviación de un modelo, norma, o especificación que no se debe a una falta de conocimiento (Murcko, 2015).

# Capítulo 1: Fundamentación Teórica

En el caso de la investigación planteada se refiere a una debilidad o defecto en un sistema informático pprovocando un resultado indeseado o el mal funcionamiento del mismo.

**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.

- Subsistema de Administración: permite la administración de los recursos del canal
- Subsistema de Transmisión: es el encargado de visualizar las noticias, las infocintas y las señales publicadas

El canal informativo muestra de forma automática ciclos de noticias constantes y repetitivos condicionado por las informaciones publicadas para determinados periodos 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 (Romero, y otros, 2012).

Por lo antes planteado se puede resumir que PRIMICIA es una Plataforma de Televisión Informativa capaz de integrar formatos como: imagen, audio, texto y video.
Está estructurado en dos subsistemas:

- Subsistema de Transmisión: visualiza las noticias, las infocintas y las señales publicadas
- Subsistema de Administración: permite la administración de los recursos del canal

El Subsistema de Transmisión visualiza las noticias y los materiales multimedia y el de administración gestiona todo lo que se transmite.

Además muestra informaciones adicionales como la fecha y hora, el tiempo restante de la noticia, o el titular de la próxima. Además permite la transmisión de señales de video externas provenientes de otro canal de televisión o de una filmación en vivo.

***Base de datos***

El término de bases de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California, USA. Una base de datos se puede definir como un conjunto de información relacionada que se encuentra agrupada o estructurada. Desde el punto de vista informático, la base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un

conjunto de programas que manipulen ese conjunto de datos. Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que queramos guardar en la tabla, cada fila de la tabla conforma un registro (Pérez, 2007).

A modo de resumen se puede decir que una base de datos no es más que un sistema formado por un conjunto de datos almacenados y está compuesta por tablas, las cuales pueden tener una o más filas o columnas.

***Registro de 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 (Alvarez, 2010). 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 (Blanco, 2012).

Resumiendo lo antes planteado se puede decir que un registro de traza es la conservación de los datos en una estructura definida de la evidencia que deja un evento o elemento al ocurrir o desplazarse, indicando la secuencia de las instrucciones de su ejecución.

## 1.4 Objeto de estudio

Gestionar los errores en la transmisión televisiva es imprescindible para el buen trabajo de las aplicaciones que hacen llegar información a los diferentes medios, ya que es la única vía de que los errores sean detectados en su mayoría. Para lograrlo es necesario el uso de algún mecanismo que se encargue de realizarla.

En el centro de GEYSED de la Universidad de las Ciencias Informáticas se trabaja en el perfeccionamiento de la Plataforma de Televisión Informativa PRIMICIA, mejorando el proceso de transmisión de las noticias, sin embargo estas mejoras serían en vano si no se realizara el proceso de detección de errores cuando se transmite o se administra 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 almacenar las trazas generadas por los mismos. El procedimiento general para la detección 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 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 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 para el monitoreo de recursos y la Gestión de los Errores en PRIMICIA.

A continuación se mencionan procesos y situaciones que generan errores en la Plataforma de Televisión Informativa PRIMICIA:

- **Ruta no encontrada**: La dirección o archivo al que se intentó acceder desde el Subsistema de Administración no existe u ocurrió un error al intentar hacer uso del mismo.
- **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 encuentran físicamente en el servidor de medias.
- **Error en el XML de configuración:** El XML de configuración posee errores de validez en su estructura o algunas de las etiquetas a las cuales se necesita acceder para recoger su contenido no existan o estén sin éste.

## 1.5 Análisis de soluciones existentes

En la actualidad la transmisión de noticias se ha relacionado estrechamente con la televisión, permitiendo visualizar noticias relevantes en el mundo a una gran cantidad de personas. Diversas televisoras utilizan sistemas informativos automatizados para procesar con mayor velocidad y mejor eficiencia la información. Los sistemas seleccionados serán analizados para obtener elementos que sirvan de base para el desarrollo de una futura solución, así como para comprender su importancia y la no existencia de una similar en el mundo.

### *1.5.1 Soluciones existentes a nivel internacional*

**GFI Events Manager**

En Europa se ha desarrollado un software llamado 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 primera, ayuda a los administradores a monitorizar el estado y seguridad de toda la red, mejorando la disponibilidad, mientras que la función de administración y archivo permite al administrador centralizar los sucesos de múltiples orígenes en varios formatos de manera que sea más sencillo identificar deficiencias, proporcionar trazas detalladas de auditoría y cumplir diversas regulaciones (Benitez, 2009).

GFI Events Manager permite activar acciones tales como: ejecución de secuencias de comandos o envío de alertas a una o más personas por correo electrónico, mensajes de red y notificaciones SMS (Alvarez, 2010)

El estudio de la solución antes planteada permitió analizar la forma de administrar y archivar los materiales sin importar sus formatos para proporcionar las trazas correspondientes y realizar el envío de alertas. Pero no puede ser utilizada como solución para la problemática planteada porque GFI Events Manager fue desarrollado específicamente para darle seguimiento a dispositivos de hardware y no a Plataformas Televisivas como tal.

**Plataforma Communi.TV (CTV)**

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. Como plataforma de desarrollo multicanal, CTV está orientada a favorecer la productividad y la creación de nuevos formatos en canales digitales.

Cuenta con un sistema de registro de trazas y monitorización de errores. Registra los tiempos de respuesta totales y parciales (transiciones de estados) permitiendo detectar, controlar y corregir los cuellos de botella y la generación de alertas.

Una vez instalado el sistema se proporcionan 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 (Esteban, 2006).

Estos dos sistemas analizados anteriormente contienen una gran cantidad de ventajas que le serían de mucha utilidad a la Plataforma de Televisión Informativa PRIMICIA, el estudio de ellos permitió organizar

el desarrollo de una estrategia que permita el almacenamiento de las trazas de los errores existentes en PRIMICIA y el envío de alertas por correo electrónico al administrador del canal, pero al igual que GFI Events Manager, la plataforma Communi.TV tampoco se puede tomar como solución a utilizar ya que su funcionamiento es muy general y la solución a realizar es específica porque está dirigida a un producto en particular, por lo que no realizarían una detección de todos los errores que ocurren en el canal. Además, las dos soluciones fueron desarrolladas con y para herramientas privativas, esto conlleva a que acudiendo a la compra de sus licencias el costo sería demasiado alto.

### *1.5.2 Soluciones existentes a nivel nacional*

En el año 2010 Elaine Morales Álvarez desarrolló el Módulo de Gestión de Errores de la Plataforma de Televisión Informativa PRIMICIA, el cuál era capaz de detectar los errores más críticos existentes en la plataforma en ese momento. Este sistema fue desarrollado utilizando tecnologías Web, por lo que ha quedado obsoleto con las nuevas versiones de PRIMICIA. Además, a causa del avance que ha tenido la plataforma y de las funcionalidades que se le han añadido durante el desarrollo de las diferentes versiones, se hizo necesario ampliar la cantidad de errores que pueda detectar el Módulo para el Monitoreo y Gestión de los errores en PRIMICIA. No obstante, esta investigación permitió profundizar sobre los conocimientos relacionados con el procedimiento de seguimiento de los errores en la Plataforma de Televisión Informativa PRIMICIA sirviendo así como base para el desarrollo de la presente investigación.

En junio del 2012 Alejandro Blanco Peña desarrolló el Módulo de Gestión de Errores para el Subsistema de Transmisión de PRIMICIA. El mismo permite identificar los posibles tipos de errores que impiden el buen funcionamiento del canal informativo y desarrolló una estrategia que permite la detección y almacenamiento en base de datos de errores ocurridos durante la transmisión del canal. Esta solución fue un gran avance para PRIMICIA ya que el estudio de la misma permitió la detección y registro de una gran cantidad de errores siendo de mucha ayuda para el desarrollo del Módulo para el Monitoreo de recursos y la Gestión de los errores en PRIMICIA, porque se puede hacer uso de las funcionalidades implementadas en la misma y añadirles nuevas de tal forma que amplíe su rendimiento hasta alcanzar los dos subsistemas. El inconveniente que tiene este módulo es que solo funciona para el Subsistema de Transmisión y de los errores que ocurren en los procesos de administración prevalece el problema de que solo es posible ver su efecto y no lo que causa el error, impidiendo así su corrección.

Durante el estudio de las soluciones existentes, también se analizó la estrategia de detección de errores que presenta el sistema operativo Windows 7 y la aplicación EventLog Analyzer.

## 1.6 Estrategias para la detección de errores

***Logs de errores en Windows 7***

Cuando el sistema operativo Windows 7 falla, los errores son almacenados en logs ubicados en un directorio donde se dispone de un desplegable que muestra donde verás los logs de programas, de seguridad, de instalación y de sistema. Estos logs brindan la información necesaria para conocer si el problema es del sistema, software o de seguridad y encauzarnos para buscar la posible solución (Alebentele, 2014).

Esta estrategia aportó ideas para el almacenamiento de los errores una vez que no se pueda acceder a la base de datos por falta de conexión.

**EventLog Analyzer**

EventLog Analyzer incluye más de 500 criterios de alerta predefinidos para Windows, Linux, Unix, aplicaciones y dispositivos de red que ocurren por los registros de eventos que notifican a los administradores cuando se genera un evento que coincida con un criterio específico predefinido. Estas alertas ayudan a los administradores a controlar los servidores y procesos críticos de la red. Además viene con otra versátil característica; la correlación de eventos en tiempo real y las notificaciones de alertas instantáneas. Se pueden configurar alertas para correlacionar eventos basados en condiciones de umbral o eventos anómalos y notificar en tiempo real para cualquier violación de los umbrales establecidos o anomalías en la red. Se puede obtener una notificación inmediata por correo electrónico y/o SMS. También puede ejecutar un script o un programa personalizado sobre la generación de alertas y tomar medidas correctivas en forma rápida para asegurar sus activos de red (IDRIC, 2016).

Esta solución permitió establecer que para identificar uno de los tipos de errores existentes en PRIMICIA era necesario utilizar la estrategia con la que cuenta este sistema de establecer un umbral, para especificar entre que valores debe mantenerse el consumo de los recursos del Subsistema de Transmisión.

## 1.7 Propuesta de solución

Una vez analizadas las soluciones anteriores se trazó una estrategia capaz de integrarse a la plataforma PRIMICIA que cuente con características como:

- El almacenamiento en la base de datos de los detalles de los errores como son: fecha, hora, tipo de error y descripción del mismo.

- En caso de no haber conexión con la base de datos, los detalles se guardaran en un archivo local para luego de reestablecida la conexión, hacer una lectura del fichero, subir la información que contiene a la base de datos y eliminar el archivo local.
- Una vez que el sistema detecta algún tipo de error, automáticamente le envía un correo electrónico al usuario especificado, informándole sobre la falla ocurrida durante el funcionamiento del canal.

Para darle inicio a su desarrollo fue necesario utilizar algunas bibliotecas que facilitaron el manejo de distintos tipos de ficheros y el envío de alertas por correo electrónico.

**boost::log**

boost::log pertenece a un conjunto de bibliotecas desarrolladas para ser utilizadas por el lenguaje de programación C ++ que proporcionan apoyo a la hora del trabajo con ficheros log.

La misma presenta características como (Torjo, 2007).

- Se adapta a gran cantidad de escenarios: desde muy simples (almacenando todo a un registro) a muy complejas (varios registros, algunos habilitados / otros no).
- Permite elegir la forma de utilizar los registros en su código
- Permite utilizar los niveles de registro (depuración, error, fatal).
- Cuenta con el filtrado de los mensajes de registro, es decir, si un registro está apagado, el mensaje no se procesa en absoluto.

Esta biblioteca se utilizó en la solución propuesta para el manejo de los ficheros log donde se almacenan los errores causados por el Subsistema de Administración.

***SmtpClient***

La clase SmtpClient es responsable de enviar o transportar el correo electrónico. El SmtpClient puede transportar el contenido del correo electrónico en la red. Los correos electrónicos pueden enviarse de forma sincrónica o asincrónica. El SmtpClient también es compatible con el envío de correo electrónico a través de SSL (Secure Socket Layer =Capa de Conexión Segura) por motivos de seguridad (Wanta, 2004).

También fue importante tener conocimiento de términos de relevancia, los cuales están relacionados con los tipos de errores que existen en el funcionamiento de la plataforma PRIMICIA.

**XML**

XML son las siglas del Lenguaje de Etiquetado Extensible. La expresión se forma a partir del acrónimo de la expresión inglesa *Etensible Markup Language*. Con la palabra "Extensible" se alude a la no limitación

en el número de etiquetas, ya que permite crear aquellas que sean necesarias. XML está orientado a los datos en sí mismos, por lo que cualquier software informático puede hacer uso de XML (Lamarca, 2013).

En la presente propuesta de solución se realiza el trabajo con ficheros XML, ya que el Subsistema de Transmisión tiene un archivo de configuración en el cuál están almacenados datos como el nombre de la base de datos a la que accede, el usuario y contraseña para el acceso a la base de datos y la dirección del servidor multimedia a donde accede para transmitir.

### 1.7.4 Ficheros Logs

Los archivos de registro (o archivos de log) son archivos que contienen mensajes sobre el sistema, incluyendo los servicios y las aplicaciones que se ejecutan en dicho sistema. Existen diferentes tipos de archivos de log dependiendo de la información (Linux, 2003).
En el caso de la solución propuesta se hace uso de los ficheros logs para darle lectura a los errores generados por el Subsistema de Administración, ya que estos son almacenados en ficheros log a los cuales hay que acceder para clasificar los tipos de errores que contienen y almacenarlos en la base de datos.

## 1.8 Metodología

Un proceso de desarrollo de software puede hacerse tan complejo como el producto lo requiera y muchas veces el control de dicho proceso se va de las manos de los desarrolladores debido a la no utilización de alguna metodología que guíe de manera coherente todas sus actividades.

Debido a que la solución propuesta debe integrarse un sistema ya desarrollado, se consideró utilizar la misma metodología de desarrollo que presenta el sistema. Por este motivo no se realiza la comparación con otras metodologías, sino que se describen las características específicas de RUP.

***RUP***

Es una metodología tradicional de desarrollo de software guiado por los casos de uso, centrado en la arquitectura, iterativo e incremental diseñado como un marco para los métodos y herramientas de UML.

# Capítulo 1: Fundamentación Teórica


*Fig. 1: Características de RUP*

**Dirigido por Casos de Uso**

Un sistema de software debe brindar servicios a sus usuarios, por lo que se debe conocer qué se necesita y desea para el futuro. Un caso de uso es una fracción de la funcionalidad del sistema. Estos representan los requisitos funcionales (¿qué debe hacer el sistema?) y especifican una secuencia de acciones que el sistema puede llevar a cabo. Tomando el modelo de casos de uso como base, los desarrolladores crean una serie de modelos de diseño e implementación que llevan a cabo los casos de uso (Gil, 2008).

Es importante que los casos de uso representen los conjuntos de procesos que intervienen en la confección de cierta funcionalidad o funcionalidades con la que cuenta el sistema. Los casos de uso no sólo inician el proceso de desarrollo sino que proporcionan un hilo conductor, permitiendo establecer trazabilidad entre los artefactos que son generados en las diferentes actividades del proceso de desarrollo.

**Centrado en la Arquitectura**

La arquitectura en un sistema de software es descrita por medio de diferentes vistas del sistema en proceso de desarrollo. La arquitectura surge de las necesidades de la organización, de cómo la perciben los usuarios y de cómo se reflejan en los casos de uso. Pero hay otros factores que también influyen tales como la plataforma en la que funcionará el software, los bloques de construcción de que se dispondrá, consideraciones de implementación, sistemas heredados y requisitos no funcionales (Gil, 2008).

En un sistema la arquitectura constituye la estructura fundamental del mismo, define los componentes más relevantes y conforma una visión más amena del sistema en cuestión.

La arquitectura involucra los aspectos estáticos y dinámicos más significativos del sistema, está relacionada con la toma de

decisiones que indican cómo tiene que ser construido el sistema y ayuda a determinar en qué orden. Además la definición de la arquitectura debe tomar en consideración elementos de calidad del sistema, rendimiento, reutilización y capacidad de evolución por lo que debe ser flexible durante todo el proceso de desarrollo. La arquitectura se ve influenciada por la plataforma de software, sistema operativo, gestor de bases de datos, protocolos, consideraciones de desarrollo como sistemas heredados.
Muchas de estas restricciones constituyen requisitos no funcionales del sistema (Romero, y otros, 2012).

**Iterativo e Incremental**

Se recomienda dividir el trabajo en partes más pequeñas o subsistemas haciendo uso del principio divide y vencerás; donde cada subsistema es una iteración que resulta en un incremento. Las iteraciones hacen referencia a pasos en el flujo de trabajo, y los incrementos, al crecimiento del producto. En cada iteración, los desarrolladores identifican y especifican los casos de uso relevantes, crean un diseño utilizando la arquitectura seleccionada como guía, implementan el diseño mediante componentes y verifican que los componentes satisfagan los casos de uso (Romero, y otros, 2012).

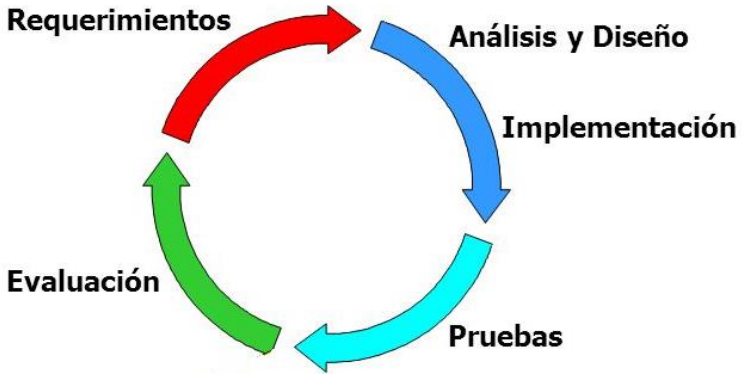



*Fig. 2: Proceso de desarrollo Iterativo de la Metodología RUP.*

Además con los artefactos que generó dicha metodología en cada una de las fases, permitió tener la documentación necesaria sobre la investigación realizada. Esto brinda una mejor comprensión del análisis y diseño del sistema en caso de que la aplicación necesite ser renovada en algún momento. Las fases también permiten establecer la oportunidad y alcance del producto e identifican las entidades externas y los actores con los que se trata, relacionándolos a los casos de uso.
Al utilizar RUP se tuvo una planificación y desarrollo del sistema.

Ventajas al utilizar RUP:

- Mitigación temprana de posibles riesgos altos

- El conocimiento adquirido en una iteración puede aplicarse de iteración a iteración.
- Identificar problemas y fallos de modo que puedan prevenirse o corregirse a tiempo.
- 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.

### *1.9 Lenguaje Unificado de Modelado (UML)*

Para facilitar la integración del Módulo para el monitoreo de recursos y la gestión de los errores de PRIMICIA con la plataforma, se decidió utilizar el lenguaje de modelado UML ya que es el que emplea el Proceso Unificado de Desarrollo de Software RUP el cuál se utilizará en todo el ciclo de desarrollo de la solución por política interna del proyecto PRIMICIA.

UML es un lenguaje estándar de modelado visual que se usa para especificar, construir, documentar y visualizar artefactos de un sistema de software. UML permite a los desarrolladores visualizar el resultado de su trabajo en esquemas o diagramas estandarizados (Pressman, 2007). Por ejemplo, símbolos o íconos característicos utilizados para capturar los requisitos. Estos íconos no son más que una notación gráfica, es decir, una síntesis; sin embargo detrás de esta notación gráfica, UML especifica un significado o semántica. Dicho lenguaje proporciona un vocabulario que incluye tres categorías:

- Elementos
- Relaciones
- Diagramas

Conteniendo aspectos conceptuales tales como procesos de negocios, funciones del sistema y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

### 1.10 Tecnologías

El proyecto PRIMICIA tiene definidas cuales son las herramientas apropiadas para el desarrollo de la Plataforma de Televisión Informativa por lo que no se realiza un estudio de las diferentes tecnologías a utilizar.

### *1.10.1 Visual Paradigm para UML*

En la selección de la herramienta UML a utilizar se escogió Visual Paradigm para UML porque soporta todos los diagramas UML y ayuda para la gestión de requisitos de software, presentando estabilidad de ejecución en diferentes sistemas operativos y la facilidad de abrir y trabajar con un modelo UML utilizando el mismo programa sin importar el sistema operativo y sin afectar en absoluto el trabajo hecho; además destacar que esta herramienta guarda todo el modelo en un solo fichero.

Visual Paradigm 8.0 es una herramienta CASE (Ingeniería de Software Asistida por Computadora) que soporta el ciclo completo de desarrollo de un sistema a través de la representación de diagramas. Es una herramienta libre y multiplataforma que se basa en UML como lenguaje de modelado. Presenta licencia comercial y gratuita. Permite la realización tanto de ingeniería directa como inversa además de un diseño enfocado al negocio que genera un software de mayor calidad (Menéndez, 2009).

Esta herramienta permite aumentar la calidad del software, a través de la mejora de la productividad en el desarrollo y mantenimiento del software. Aumenta el conocimiento informático de una empresa ayudando así a la búsqueda de soluciones para los requisitos. También permite la reutilización del software, portabilidad y estandarización de la documentación.

## 1.11 Lenguajes de programación

***C++***

C++ es un lenguaje de programación de utilidad general diseñado para que el programador serio haga su trabajo de modo más agradable. Salvo algunos detalles menores, C++ es un súper conjunto del lenguaje de programación C. Además de los recursos que ofrece C, C++ proporciona mecanismos flexibles y eficientes para definir nuevos tipos de datos.

Este lenguaje de programación está considerado por muchos como uno de los lenguajes más potentes, debido a que permite trabajar tanto a alto nivel como a bajo nivel. Se puede decir que C++ abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos, dicho lenguaje gana mucho en potencia ya que tiene la posibilidad de sobrecargar operadores. Las principales características de C++ son las facilidades que proporciona para la programación orientada a objetos y para el uso de plantillas o programación genérica (STROUSTRUP, 2013).

Este lenguaje de programación se utilizó con el fin de satisfacer las necesidades de PRIMICIA, ya que la solución que necesita debe ser desarrollada utilizando tecnologías de escritorio y no web.

## 1.12 Framework o marco de trabajo

Un *framework* es una infraestructura para la creación de otros programas porque funciona como base para desarrollar programas futuros de forma rápida. El framework contiene bibliotecas de código y módulos ya listos que resumen las tareas de creación de elementos recurrentes en el desarrollo de aplicaciones, a la vez que define una arquitectura para el desarrollo de software (Cumare, 2014).

***QT***

Qt es un framework multiplataforma, que se utiliza para el desarrollo de aplicaciones, el cual está escrito en C++. Qt ofrece bibliotecas de código para: creación de interfaces gráficas de usuario, acceso a bases de datos, manipulación de contenido XML y comunicación en red. Además, extiende el lenguaje de programación C++, a través de macros y meta-información, agregando nuevas características como: el bucle *foreach*, la sentencia *forever* e introspección (Digia, 2012).

Componentes del framework Qt a utilizar:

- Las bibliotecas de Qt.
- Qt Designer: Diseñador de interfaces gráficas de usuario.
- Qt Assistant: Acceso rápido a la documentación de Qt.

## 1.13 Entorno de Desarrollo Integrado (IDE)

Un IDE puede incluir varios o un solo lenguaje de programación. Estos entornos de programación contienen un editor de código, compilador, depurador y construyen las interfaces gráficas. (JACOBSON, 2013).

***QT Creator 3.1.1***

Qt Creator ofrece un entorno de desarrollo integrado (IDE) multiplataforma, para que los desarrolladores de aplicaciones puedan crear aplicaciones para múltiples plataformas de escritorio y dispositivos móviles, como Android y iOS. Está disponible para Linux, OS X y sistemas operativos de Windows. Qt Creator fue creado por la compañía Trolltech para el desarrollo de aplicaciones con las bibliotecas Qt (Digia, 2012).

En la solución propuesta QT Creator permitió desarrollar una aplicación haciendo uso del asistente de proyectos así como el acceso a proyectos recientes, además de que permitió el trabajo con el editor de código avanzado de C++ que provee características para completar recortes de código.

# Capítulo 1: Fundamentación Teórica

## 1.14 Sistema Gestor de Base de *Datos*

***PostgreSQL***

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD (Berkeley Software Distribution = Distribución de Software Berkeley) y con su código fuente disponible libremente. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Funciona correctamente con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema (Martínez, 2010).

Para el desarrollo de la siguiente investigación se utilizó la versión 9.3 de PostgresSQL ya que incluye algunas mejoras en las características con respecto a algunas versiones anteriores como son:

- Robustez
- Optimizador de consultas.
- Multiplataforma (Linux, Windows, Mac OS X, Solaris, y otros más).

## Conclusiones del capítulo

Como resultado del análisis de las principales tecnologías y herramientas que se emplean para la construcción de la solución, se seleccionan las que se consideran más adecuadas teniendo en cuenta los objetivos del presente trabajo. Como entorno de desarrollo se escoge Qt Creator aprovechando las potencialidades a la hora de implementar mediante el marco de trabajo Qt. El lenguaje de modelado UML se utilizó para indicar teóricamente qué es lo que debe hacer el sistema, como herramienta CASE se utilizó Visual Paradigm para generar los artefactos durante el desarrollo del software, el lenguaje de programación C++ para que el sistema posea una rápida ejecución y para almacenar los datos se utilizó el sistema gestor de Base de datos PostgreSQL.

# Capítulo 2: Análisis y Diseño de la Solución Propuesta

## 2.1 Introducción

En el presente capítulo se realiza el análisis y diseño de la solución propuesta. Se presenta el modelo de dominio y se describen los procesos reflejados en él, permitiendo un mejor entendimiento de la solución. Además se enumeran y describen los requisitos funcionales y no funcionales que debe tener el sistema, para conocer las condiciones y cualidades deseadas por los usuarios. Luego se realiza el diagrama de casos de uso del sistema, los diagramas de clases de análisis y diseño para un mejor entendimiento de las funcionalidades del módulo. Además se describe la arquitectura haciendo énfasis en los patrones de arquitectura y diseño utilizados para la realización del sistema. Se realizan los diagramas de clases del diseño y se elaboró el modelo de clases correspondiente al sistema.

## 2.2 Modelo 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. Por tanto, el modelo del dominio, como muestra la Figura 1 podría considerarse como un diccionario visual de las abstracciones relevantes, vocabulario del dominio e información del dominio. Un modelo del dominio es una representación de las cosas del mundo real del dominio de interés, no de componentes software, como una clase Java o C+ + u objetos software con responsabilidades (Larman, 2003).

A continuación se muestra el modelo de dominio de la presente investigación:

# Capítulo 2. Análisis y diseño de la solución propuesta

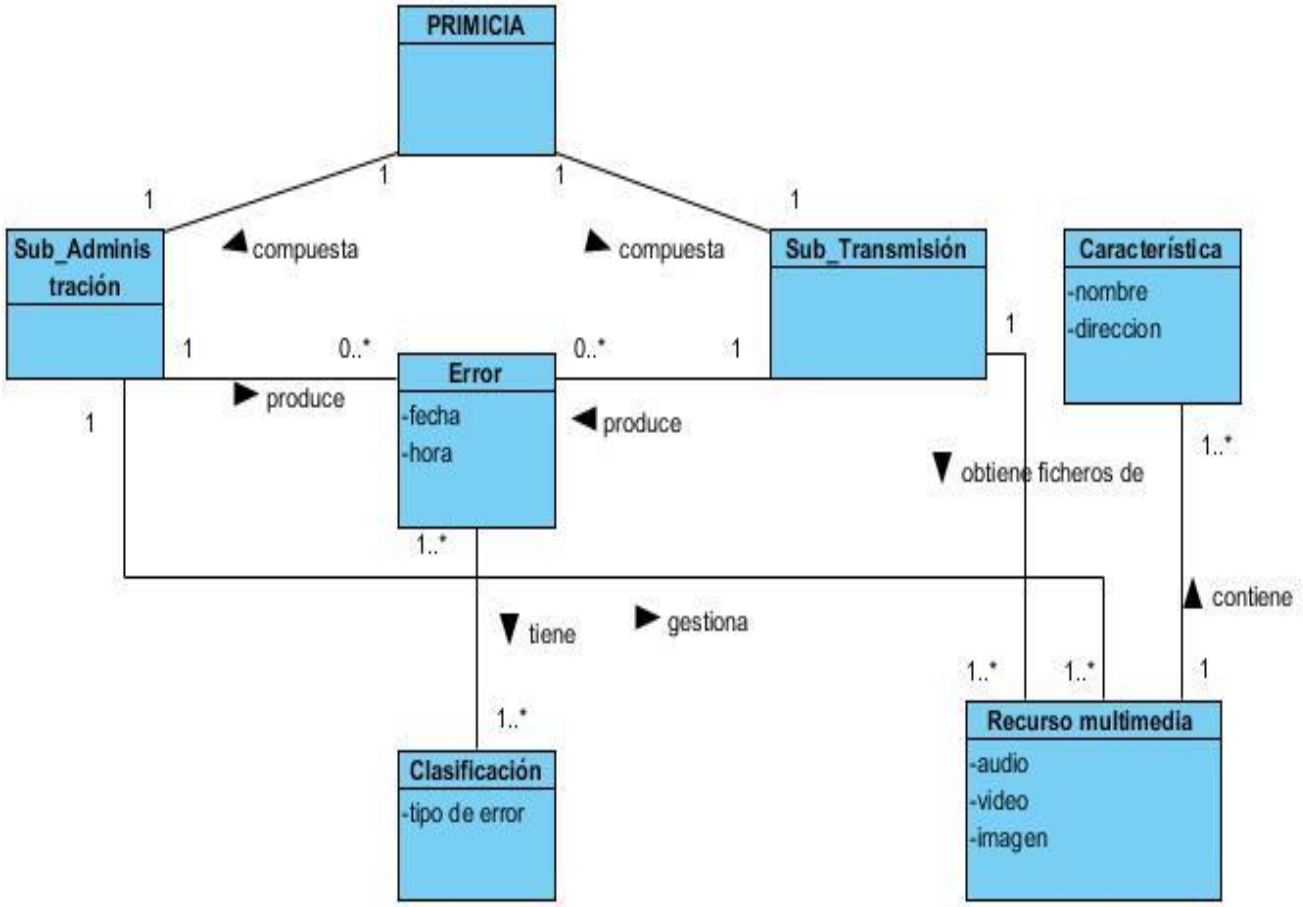



*Fig. 3 Diagrama del modelo de dominio.*

### *2.2.1 Conceptos más relevantes del diagrama del modelo de dominio.*

**PRIMICIA:** es una solución integrada que provee un canal de televisión capaz de presentar informaciones haciendo uso de formatos como: texto, video y audio con el objetivo de mostrar la información lo más sugerente posible.

**Subsistema de Administración**: permite la administración del canal y toda la gestión de las noticias, señales, infocintas y recursos multimedia.

**Subsistema de Transmisión:** es el encargado de visualizar las noticias y materiales publicados.

**Error:** Tanto en el Subsistema de Transmisión como en el de Administración de la Plataforma de Televisión Informativa PRIMICIA ocurren varios errores que impiden el buen funcionamiento del canal. Ejemplos de estos errores son:

1. Se interrumpe o detiene el proceso de transmisión:

En la aplicación de Transmisión se encuentran ejecutándose un grupo de herramientas que son las responsables de transmitir la combinación de las imágenes, los videos y textos, ya que esta combinación

forma parte de las noticias y al ocurrir una falla en este proceso se afecta la transmisión de las noticias provocando que la plataforma quede fuera de servicio al detenerse la aplicación y no se le brinde la información requerida a los televidentes.

2. Interrupción del servicio de PostgresSQL:

La interrupción de este servicio trae como consecuencia la caída de la aplicación, ya que varias de las funcionalidades que brinda la plataforma dependen del mismo por ser uno de los motores principales del funcionamiento de PRIMICIA.

3. Caída de la red entre servidores:

Un problema de este tipo cuando los servidores están separados provoca la afectación de la transferencia de información entre los servidores y como consecuencia no se tiene acceso a todos los servicios que presenta la plataforma.

## 2.3 Requisitos del sistema

Los requisitos son declaraciones que identifican atributos, capacidades, características y/o cualidades que necesita cumplir un sistema (o un sistema de software) para que tenga valor y utilidad para el usuario. En otras palabras, los requisitos muestran qué elementos y funciones son necesarias para un proyecto (Alegsa, 2009).

### *2.3.1 Requisitos funcionales.*

Los requisitos funcionales son capacidades o condiciones con las cuales el sistema debe cumplir e indican su comportamiento, es decir, son las funciones que el sistema debe realizar (Quesada, 2007). A continuación se listan y describen los requisitos funcionales que debe cumplir el sistema para la solución planteada:

**RF1: Cargar fichero de configuración del Subsistema de Transmisión:** el sistema debe ser capaz de permitirle al usuario cargar el fichero de configuración de transmisión.

**RF1.2: Identificar error en el XML de configuración de Transmisión:** El sistema debe ser capaz de identificar los errores en la estructura del XML de configuración del Subsistema de Transmisión.

**RF2: Identificar error de origen de los datos:**

**RF2.1: 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.

**RF2.2: 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.

# Capítulo 2. Análisis y diseño de la solución propuesta

**RF2.3: Identificar error de disponibilidad de audios:** El sistema debe ser capaz de identificar si la ruta del audio en el servidor de medias se encuentra disponible

**RF3: Identificar error de existencia de medias:**

**RF3.1: Identificar error de existencia física de videos:** El sistema debe ser capaz de identificar si los la dirección almacenada en la base de datos de los videos coinciden con los físicos en el servidor de medias.

**RF3.2: Identificar error de existencia física de imágenes:** El sistema debe ser capaz de identificar si la dirección almacenada en la base de datos de las imágenes coinciden con las físicas en el servidor de medias.

**RF3.3: Identificar error de existencia física de audio** El sistema debe ser capaz de identificar si la dirección almacenada en la base de datos de los audios coinciden con la física en el servidor de medias.

**RF4: Establecer límite de consumo de RAM y CPU:** el sistema debe ser capaz de permitirle al usuario establecer un límite por el cuál debe medirse el consumo de RAM y CPU para no ser identificados como errores.

**RF4.1: Identificar alto consumo de RAM y CPU:** El sistema debe ser capaz de identificar si el consumo de RAM y CPU del Subsistema de Transmisión se encuentra por encima del porciento establecido por el usuario.

**RF5: Cargar fichero de errores del Subsistema de Administración:** El sistema debe ser capaz de obtener los errores contenidos en los ficheros logs del Subsistema de Administración.

**RF6: Identificar error de Ruta no encontrada en el Subsistema de Administración:** El sistema debe ser capaz de identificar cuando ocurre un error al acceder a un directorio o archivo en el subsistema de administración.

**RF7: Identificar error crítico en la administración:** El sistema debe ser capaz de identificar los errores de carácter crítico almacenados en el log de errores del Subsistema de Administración.

**RF8: Almacenar error en base de datos:** el sistema debe ser capaz de guardar los errores encontrados en la base de datos.

**RF9: Identificar error de base de datos:** El Sistema debe ser capaz de identificar si no existe conexión con la base de datos.

**RF9.1 Guardar errores en fichero local:** En caso de no existir conexión con la base de datos, el sistema debe ser capaz de guardar los errores en un fichero local.

# Capítulo 2. Análisis y diseño de la solución propuesta

**RF9.2: Subir error de 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 reanudarse la conexión con la base de datos.

**RF10: Listar errores:** el sistema debe ser capaz de mostrar un listado de los errores encontrados clasificados por tipo.

**RF11: Enviar alerta:** Cuando ocurre un error el sistema debe mandar un correo electrónico de forma automática al correo especificado informando que ocurrió una falla.

**RF11.1: Configurar correo:** El sistema debe ser capaz de permitirle al usuario establecer la dirección de correo a donde se van a enviar las alertas.

### *2.3.2 Requisitos no funcionales.*

Los requerimientos no funcionales especifican las propiedades o cualidades que debe tener la solución a desarrollar. Representan las características que hacen al producto atractivo, usable, rápido o confiable. Estos requisitos pueden marcar la diferencia entre un producto bien aceptado y otro con poca aceptación. A continuación se listan estas características:

**Software:**

Utilizar como sistema operativo en los servidores la distribución de Ubuntu 15.04 o superior.

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

*Tabla 1: Requisitos no funcionales de hardware*

**Hardware**:

| Servidor | Procesador | Memoria RAM | Disco Duro | Tarjeta de Red |
|---|---|---|---|---|
| **Transmisión** | Intel Pentium Dual Core G630 2.7GHz | 1GB | 500Gb | Ethernet 10/100 Mbps |
| **Administración** | Pentium IV 2.8 GHz | 1 GB | 120 Gb | Ethernet 10/100 Mbps |

**Soporte:**

# Capítulo 2. Análisis y diseño de la solución propuesta

El Módulo debe estar documentado y proveer el código fuente entendible para posibles modificaciones futuras en el mismo, con el fin de potenciar su alcance en caso de que se produzcan nuevos tipos de errores.

**Restricciones de diseño y la implementación**

1. Utilizar Qt Creator 3.1.1 como IDE de desarrollo y PostgreSQL 9.3 o superior como sistema gestor de base de datos.
2. El sistema estará implementado en lenguaje C++ utilizando como lenguaje de modelado UML y como herramienta CASE Visual Paradigm 8.0 para la modelación del sistema.

## 2.4 Modelo de Casos de Uso

Los componentes primarios de un modelo de casos de uso son los casos de uso, los actores y el sistema modelado. Un caso de uso es una técnica de modelado usada para describir lo que debería hacer un sistema nuevo o lo que hace un sistema que ya existe. En el modelado de casos de uso, el sistema se observa como una caja negra que proporciona casos de uso.

**Definición de Actores**

*Tabla 2: Actores del Sistema.*

| Actor | Descripción |
|---|---|
| **Sistema** | Es el encargado de darle inicio al proceso de monitorización de recursos y detección de errores que pueden ocurrir durante el funcionamiento de la plataforma. |
| **Usuario** | Es aquella persona que recibirá notificaciones cada vez que ocurra un error, además podrá cargar los ficheros de registro de errores de cada subsistema para visualizar los errores de forma clasificada y modificar el estándar de porciento de consumo de RAM y CPU por el que debe medirse la transmisión. |

# Capítulo 2. Análisis y diseño de la solución propuesta

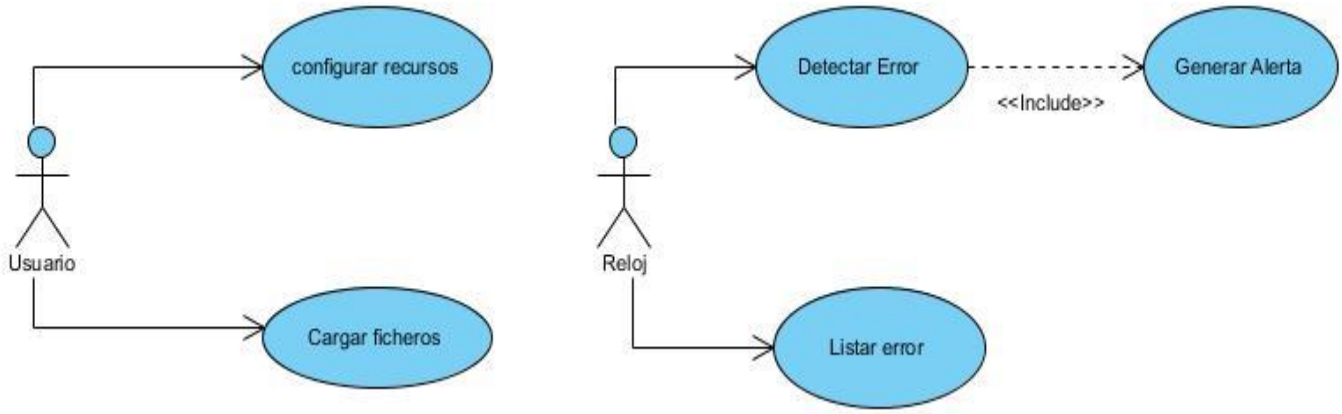



*Fig. 4 Diagrama de Casos de Uso del Sistema.*

### *Descripción textual de los Casos de Uso*

*Tabla 3: Descripción del CU Cargar Fichero*

| | |
|---|---|
| **Caso de Uso** | Cargar fichero |
| **Actores** | Usuario |
| **Resumen** | El caso de uso inicia cuando el actor desea cargar el fichero de configuración del Subsistema de Transmisión y el fichero log donde se almacenan los errores del Subsistema de Administración para comenzar el proceso de detección de errores en ambos subsistemas. El caso de uso finaliza cuando el sistema muestra la información de la búsqueda de errores realizada. |
| **Precondiciones** | Deben existir el fichero de configuración de transmisión y los logs de errores de la administración. |
| **Referencias** | RF1,RF6 |
| **Prioridad** | Baja |
| **Flujo Normal de Eventos** | |

# Capítulo 2. Análisis y diseño de la solución propuesta

| Acción del actor | Respuesta del sistema |
|---|---|
| **1.** El caso de uso comienza cuando el actor accede a la funcionalidad de cargar algún fichero, ya sea el de transmisión o el de administración. | |
| | 2. Después de leer el fichero, el sistema accede a los datos que contiene verificando su estructura y los errores que contiene y finaliza el caso de uso. |
| **Flujos Alternos** | |
| **Acción del Actor** | **Respuesta del Sistema** |
| | 1. En caso de no haber sido cargado ningún fichero, el sistema muestra una alerta informando sobre la necesidad de cargarlos para darle inicio al proceso de detección de errores. |

*Tabla 4: Descripción del CU Listar error.*

| | |
|---|---|
| **Caso de Uso** | Listar error |
| **Actores** | Sistema |
| **Resumen** | El caso de uso inicia cuando el actor detecta algún error durante el funcionamiento de la plataforma. El caso de uso finaliza cuando se detenga el sistema o el funcionamiento del canal. |
| **Precondición** | El canal debe haber sido puesto en funcionamiento. |
| **Referencias** | RF11 |
| **Prioridad** | Baja |

# Capítulo 2. Análisis y diseño de la solución propuesta

| Flujo Normal de Eventos | |
|---|---|
| **Acción del actor** | **Respuesta del sistema** |
| **1.** El caso de uso comienza cuando el actor accede a la base de datos, al servidor de medias y a los ficheros logs en busca de errores. | |
| | 2. El sistema muestra un listado con los errores encontrados clasificados por tipo y finaliza el caso de uso. |

*Tabla 5: Descripción del CU Configurar recursos.*

| | |
|---|---|
| **Caso de Uso** | Configurar recursos |
| **Actores** | Usuario |
| **Resumen** | El caso de uso inicializa cuando el Usuario accede a especificar los porcientos de RAM y CPU por los que debe medirse el consumo del Subsistema de Transmisión. El caso de uso finaliza cuando se modifican estos valores satisfactoriamente. |
| **Precondiciones** | El sistema debe de estar en funcionamiento. |
| **Referencias** | RF5 |
| **Prioridad** | Alta |
| **Flujo Normal de Eventos** | |
| **Acción del actor** | **Respuesta del sistema** |
| 1. El caso de uso inicia cuando el actor accede a modificar el porciento de consumo de RAM y CPU. | |

# Capítulo 2. Análisis y diseño de la solución propuesta

| | |
|---|---|
| | 2. El sistema muestra los campos para insertar los porcientos de RAM y CPU deseados por el usuario. |
| 3. El usuario inserta los valores y presiona el botón actualizar | |
| | 4. El sistema informa si la actualización fue correcta y finaliza el caso de uso. |

*Tabla 6: Descripción del CU Detectar error.*

| | |
|---|---|
| **Caso de Uso** | Detectar error |
| **Actores** | Sistema |
| **Resumen** | El caso de uso comienza cuando se le da inicio al funcionamiento del canal. |
| **Precondiciones** | El canal debe de estar en funcionamiento. |
| **Referencias** | RF1.2, RF2, RF3, RF4.1, RF7,RF9 |
| **Prioridad** | Alta |
| **Flujo Normal de Eventos** | |
| **Acción del actor** | **Respuesta del sistema** |
| 1. El caso de uso inicia cuando el actor inicializa la acción de comprobar el consumo de RAM y CPU del Subsistema de Transmisión y a su vez acude a la identificación de errores que ocurren durante | |

# Capítulo 2. Análisis y diseño de la solución propuesta

| | |
|---|---|
| el funcionamiento del canal. | |
| | 2. El sistema realiza el proceso de identificación, clasificación y almacenamiento de los errores detectados. |
| **Flujos Alternos** | |
| **Acción del Actor** | **Respuesta del Sistema** |
| | 1. En caso de haber error en la conexión a la base de datos se guardan los detalles de los errores en un fichero local. En caso de no haber errores finaliza el caso de uso. |

*Tabla 7: Descripción del CU Enviar Alerta.*

| | |
|---|---|
| **Caso de uso** | Enviar Alerta |
| **Actores** | Sistema |
| **Resumen** | El caso de uso comienza cuando el actor detecta algún error |
| **Precondiciones** | El sistema tiene que estar en funcionamiento |
| **Referencias** | RF11 |
| **Prioridad** | Media |
| **Flujo normal de eventos** | |
| **Acción del actor** | **Respuesta del sistema** |
| 1. El caso de uso inicia cuando el actor detecta algún error durante el funcionamiento de la plataforma | |
| | 2. El sistema envía un correo de forma automática al usuario designado para recibir las notificaciones indicándole el tipo de error y la hora a la que se detectó. |

# Capítulo 2. Análisis y diseño de la solución propuesta

## 2.5 Modelo de Análisis

El modelo de análisis es la primera representación técnica de un sistema. Utiliza una mezcla de formatos en texto y diagramas para representar los requisitos del software, las funciones y el comportamiento. De esta manera se hace mucho más fácil de comprender dicha representación, ya que es posible examinar los requisitos desde diferentes puntos de vista aumentando la probabilidad de encontrar errores, de que surjan debilidades y de que se descubran descuidos.

### 2.5.1 ¿Por qué no realizar el modelado de análisis?

Aunque RUP constituye la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos, no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización y del desarrollador que la utilice, por lo que puede omitirse el modelado de análisis para emplear el tiempo en el resto del desarrollo de la investigación. El resultado de esta personalización se reflejará en el proceso específico del proyecto y no existirían grandes riesgos, por esa razón y para ahorrar tiempo en el diseño incrementando de esta forma el tiempo de implementación y validación del sistema, se decidió no realizar el modelado de análisis.

## 2.6 Arquitectura del sistema

La arquitectura de software de un sistema de programa o computaci6n es la estructura de las estructuras del sistema, la cual comprende 10 componentes del software, las propiedades de esos componentes visibles externamente, y las relaciones entre ellos (Pressman, 2005).

Para la presente investigación se tuvo en cuenta que las arquitecturas de n-capas proporcionan una gran cantidad de beneficios para las entidades que necesitan soluciones flexibles y fiables para resolver complejos problemas inmersos en cambios constantes y como PRIMICIA es un producto que está en constante desarrollo, se escogió esta arquitectura teniendo en cuenta algunas de las ventajas que presenta como por ejemplo:

- Desarrollos paralelos (en cada capa)
- Mantenimiento y soporte más sencillo (es más sencillo cambiar un componente que modificar una aplicación monolítica)
- Mayor flexibilidad (se pueden añadir nuevos módulos para dotar al sistema de nueva funcionalidad)
- Alta escalabilidad. La principal ventaja de una aplicación distribuida bien diseñada es su buen escalado, es decir, que puede manejar muchas peticiones con el mismo rendimiento simplemente

añadiendo más hardware. El crecimiento es casi lineal y no es necesario añadir más código para conseguir esta escalabilidad.

**Arquitectura en Capas (3 capas).**


*Fig. 5 Arquitectura en capas (3 capas).*

**Capa de presentación:** esta capa se encarga de proveer interacción entre el sistema y el usuario a través de las interfaces, además captura y valida los datos del usuario, comunicándose únicamente con la capa de negocio.

**Capa Lógica de Negocio:** Esta capa se encarga de hacer la invocación a la capa de acceso a datos cada vez que el sistema hace una petición de los datos ubicados en la misma, encargándose así de la recepción de solicitudes de la capa de presentación.

**Capa de Almacenamiento o Acceso a datos:** Se encarga del almacenamiento y recuperación de los datos del sistema. Está formada por el gestor de base de datos PostgresSQL que atiende las solicitudes de almacenamiento o recuperación de información provenientes de la capa lógica de negocio.

## 2.8 Patrones de diseño

Los patrones de diseño se aplican a un elemento específico del diseño como un agregado de componentes para resolver algún problema de diseño, relaciones entre los componentes o los mecanismos para efectuar la comunicación de componente a componente (PRESSMAN, 2005).

Los patrones del diseño tratan los problemas del diseño que se repiten y que se presentan en situaciones particulares del diseño, con el fin de proponer soluciones a ellas. Por lo tanto, los patrones de diseño son soluciones exitosas a problemas comunes. Existen muchas formas de implementar patrones de diseño (Villa, 2007).

# Capítulo 2. Análisis y diseño de la solución propuesta

Para el desarrollo de la solución se aplicaron los siguientes patrones de diseño:

- Patrones Generales de Asignación de Responsabilidad de Software (GRASP por sus siglas en ingles)
- grupo de los Cuatro (GoF por sus siglas en ingles)

**Patrones GRASP:** 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 (GROSSO, 2011).

**Patrones GOF:** Los patrones GOF (Gang Of Four) están dirigidos al desarrollo de sistemas orientados a objetos y se encuentran divididos en tres grupos: los de creación utilizados en la abstracción de cómo es creado un objeto, los de estructura que indican cómo se encuentran compuestas las clases y los de comportamiento utilizados en la asignación de responsabilidades en las clases (PRESSMAN, 2005).

### *2.8.1 Patrones GRASP utilizados*

- **Creador:** Este patrón se utiliza para guiar la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. Está destinado a encontrar un creador que necesite estar conectado al objeto creado en un evento en particular, Separando la construcción de un objeto complejo de su representación, de forma que el mismo proceso de construcción pueda crear diferentes representaciones. Por lo tanto se aplica en la clase **CControlador_Error**.

- **Alta Cohesión:** Este patrón permite asignar una responsabilidad de modo que la cohesión siga siendo alta. Es un principio que debemos tener presente en todas las decisiones de diseño, es la meta principal que ha de buscarse en todo momento. Es un patrón evaluativo que el desarrollador aplica al valorar sus decisiones de diseño. Este patrón fue utilizado al desarrollar la clase **CControlador_Error**.

- **Experto:** Este patrón se utiliza para asignarle responsabilidad a la clase que cuenta con la información necesaria para cumplir la responsabilidad. Es un patrón que se usa más que cualquier otro al asignar responsabilidades. Y en la presente investigación se utiliza en la clase **CError**, ya que esta clase es la única que conoce los detalles de los errores ocurridos.

- **Bajo Acoplamiento:** Estimula asignar una responsabilidad, de modo que su colocación no incremente el acoplamiento tanto que produzca los resultados negativos propios de un alto acoplamiento. Además este patrón permitirá soportar el diseño de clases más independientes, que

reducen el impacto de los cambios y también más reutilizables, que acrecientan la oportunidad de una mayor productividad. Se incluye como uno de los principios del diseño que influyen en la decisión de asignar responsabilidades. Este patrón se evidencia en la clase **CControlador_Error** favoreciendo la flexibilidad del diseño.

- **Controlador:** Este patrón se utiliza en la clase **CControlador_Error** para responder la pregunta ¿Quién gestiona los eventos del sistema? Funcionando como controladora de los eventos del sistema recibiendo los datos de una clase y enviándolos hacia otras clases según el método llamado.

### *2.8.2 Patrones GOF utilizados*

- **Fachada**: El patrón Fachada es del tipo estructural. Define una interfaz de alto nivel que hace que el sistema sea más fácil de usar. Simplifica el acceso a un conjunto de clases proporcionando una única clase que los programadores utilicen para comunicarse con dicho conjunto de clases. En el contexto del sistema la clase **CErrorPrimicia** funciona como fachada, ya que a través de ella el cliente puede acceder a las funcionalidades de las demás clases con las que esta se relaciona sin necesidad de que el cliente acceda directamente a las mismas.

- **Observador:** El patrón Observador es del tipo comportamiento. Define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él. Se pone de manifiesto en la clase **CControlador_error**.

- **Singleton:** El patrón Singleton según su propósito es del tipo Creación. Se pone de manifiesto en la clase **CAcc_Dato** con el objetivo de proveer un mecanismo para limitar el número de instancias de una clase. Por lo tanto el mismo objeto es siempre compartido por distintas partes del código.

## 2.9 Modelo de diseño

Para poder realizar un sistema que soporte todos los requisitos especificados, tanto funcionales como no funcionales, se necesita modelarlo, lo cual se hace a través del diseño. El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar.

# Capítulo 2. Análisis y diseño de la solución propuesta

**Diagrama de clases del diseño**.

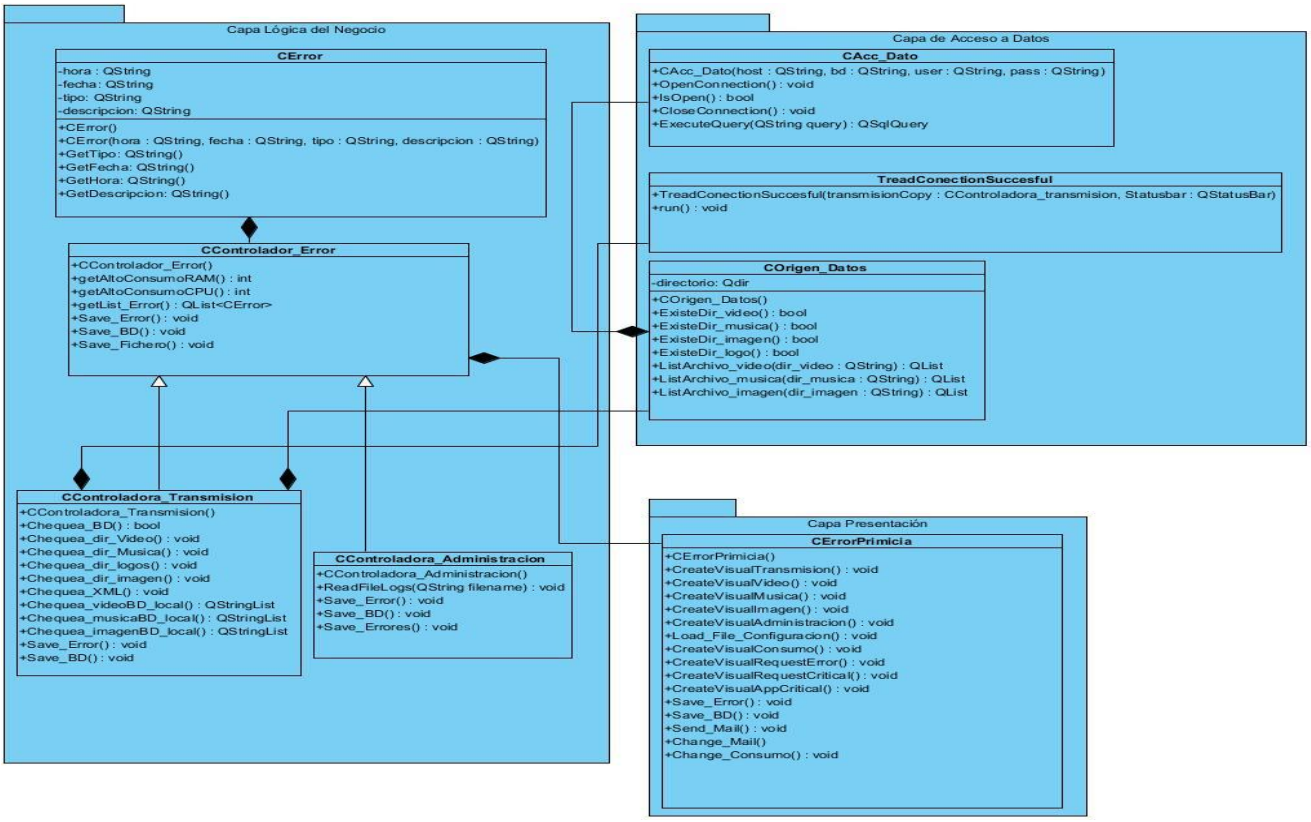



*Fig. 6 Diagrama de clases del diseño*

## 2.10 Diagrama de clases persistentes.

La persistencia de una clase está definida por la propiedad de los objetos de trascender su estado en el tiempo y el espacio: una clase persistente existirá durante la ejecución de un programa, y deberá sobrevivir incluso a la eliminación o colapso del mismo. Lo contrario a las clases persistentes son las clases temporales que son manejadas y almacenadas por el sistema en tiempo de ejecución, por lo que dejan de existir cuando termina el programa (Quintana, y otros, 2011).

El diagrama de Clase Persistente de la presente investigación es el siguiente:

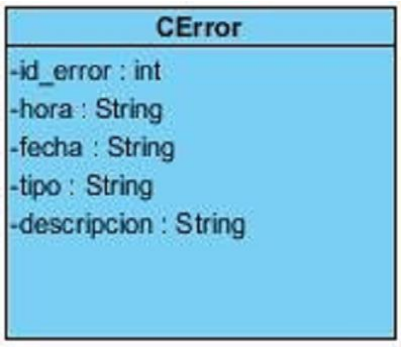



*Fig. 7 Diagrama de clases persistentes*

## 2.11 Conclusiones del capítulo

Al culminar esta fase puede decirse que, se elaboró el modelo de dominio para una idea más clara y definida del funcionamiento del sistema. Se definieron los requisitos funcionales y no funcionales del sistema donde quedaron plasmados 11 requisitos funcionales agrupados en 5 casos de uso. Además se concluyó que los diagramas expuestos permiten garantizar la presentación, la lógica del negocio y el acceso a los datos que conforman el módulo para luego de realizadas las pruebas, comprobar que se cumpla con las especificaciones trazadas al desarrollar los requisitos definidos en el transcurso de la investigación.

# Capítulo 3: Implementación y Prueba de la Solución Propuesta

## 3.1 Introducción

En el capítulo se hace una descripción sobre los procesos de implementación y pruebas realizadas al sistema para verificar que los requisitos funcionales fueron cumplidos y para asegurar el correcto funcionamiento de la aplicación. Se muestra el diagrama de componentes donde se describen los elementos físicos del sistema y sus relaciones. Finalmente se especifican las pruebas realizadas y los resultados alcanzados.

## 3.2 Modelo de Implementación

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

## 3.4 Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo código fuente, binario y ejecutable (joshell, 2010).

Un componente es una unidad de despliegue independiente. Un componente podría ser utilizado por cualquier organización sin necesidad de ningún conocimiento previo. Un componente no tiene estado persistente en sí mismo, lo tienen (opcionalmente) sus objetos. Implementa una serie de interfaces y utiliza otras. Encierran otros elementos de modelado como por ejemplo clases, permitiendo a otros componentes acceder a ellos (Carlos, 2009).

**Diagrama de Componentes**

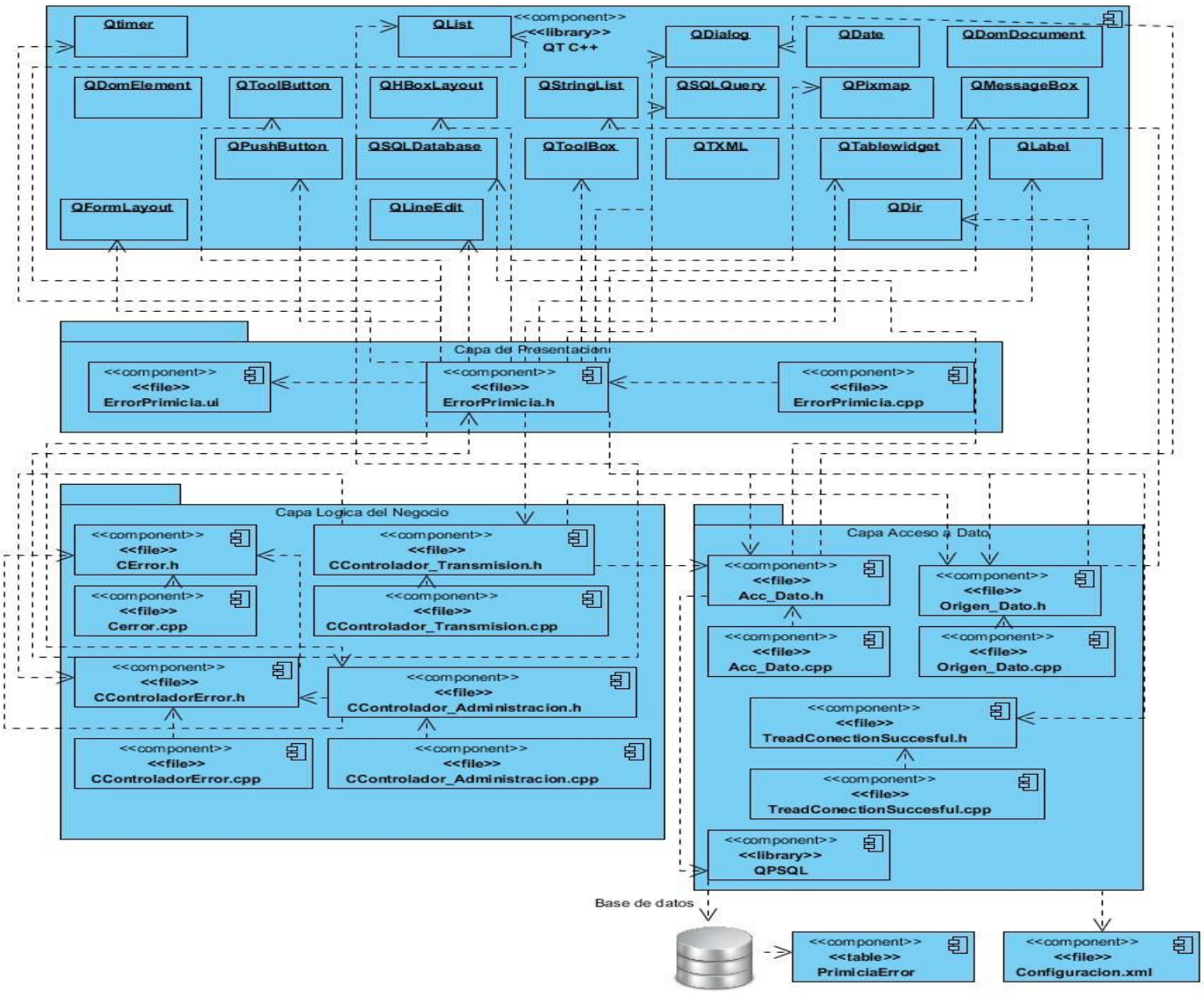



*Fig. 8: Diagrama de componentes*

## 3.3 Diagrama de Despliegue

El diagrama de despliegue describe la distribución física del sistema, muestra cómo están distribuidos los componentes de software entre los distintos nodos de cómputo. Los nodos son los elementos que participan en la ejecución de un sistema representando el empaquetamiento físico de los elementos lógicos (Ferré, y otros, 2011).

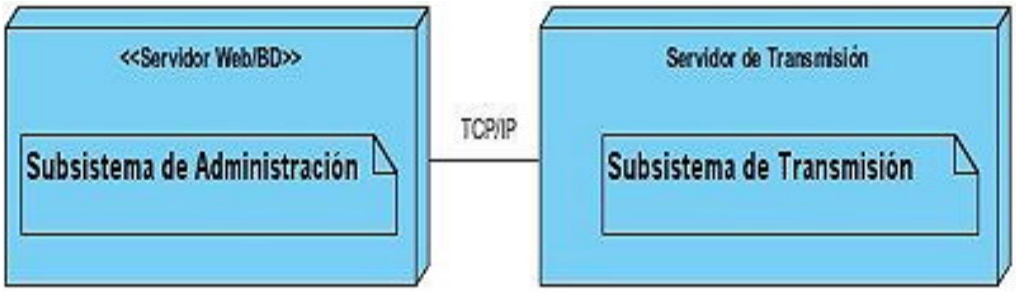



*Fig. 9 Diagrama de despliegue*

**Servidor Web/BD:** Este nodo representa el servidor donde está alojado el Subsistema de Administración y la base de datos.

**Servidor de Transmisión:** Este nodo representa el subsistema encargado de transmitir todas las noticias conteniendo además la solución propuesta para poder realizar la lectura del consumo de RAM y CPU de este servidor de transmisión.

### 3.6 Pruebas del sistema

La realización de las pruebas es una de las etapas principales durante todo el ciclo de vida del software. Las mismas constituyen una serie de actividades donde cada componente, o parte de la aplicación a comprobar, son ejecutados bajo determinadas condiciones. Su objetivo principal es evaluar y elevar la calidad del producto final.

El tipo de prueba que se va a realizar es la prueba de caja negra se refiere a las pruebas que se llevan a cabo en la interfaz del software. Las pruebas de caja negra son diseñadas para validar 10 requisitos funcionales sin fijarse en el funcionamiento interno de un programa. Las técnicas de prueba de caja negra se centran en el ámbito de información de un programa, de forma que se proporcione una cobertura completa de prueba. Los métodos de prueba basados en grafos exploran las relaciones entre 10s objetos del programa y su comportamiento. La partición equivalente divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. El análisis de valores limites prueba la habilidad del programa para manejar datos que se encuentran en 10 límites aceptables. (Pressman, 2005).

Este tipo de prueba no incluye revisión de codificación interna del software ya que se especializa en la comprobación de interfaces, contando con el apoyo de casos de prueba de cada funcionalidad. Se realiza con el objetivo de detectar errores de las siguientes categorías:

- Errores de funciones incorrectas o ausentes.
- Errores de interfaz.

- Errores en estructura de datos o en accesos de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación

Dentro de los métodos de caja negra se utilizó la técnica de partición de equivalencia. Esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.


*Fig. 10: Interpretación gráfica de Caja negra*

### *3.7 Casos de prueba*

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

**Caso de uso:** Detectar error

**Descripción General:**

El caso de uso inicia cuando el actor Sistema inicia la acción de comprobar el consumo de RAM y CPU de la aplicación y a su vez acude a la identificación de errores que ocurren durante el funcionamiento del canal. En caso de no haber errores finaliza el caso de uso.

**Condiciones de ejecución:**

Solo se ejecuta esta acción cuando el canal esté en funcionamiento y el actor Sistema inicialice el chequeo en busca de errores.

**Escenarios a Probar en el CUS:**

*Tabla 8: Escenarios CUS Detectar error.*

| Nombre de la sección | Escenarios de la sección | Descripción de la funcionalidad | Flujo central |
|---|---|---|---|
| SC 1: Detectar error | EC 1.1: Se identifica alguno de los tipos de errores | El sistema comprueba la existencia de algún error durante el funcionamiento de la Plataforma PRIMICIA, | Clic en “Cargar fichero xml de configuración”<br>Clic en “Cargar fichero log de |

| | | | |
|---|---|---|---|
| | | genera un listado con los mismos y los almacena en la Base de Datos | Administración” |
| | EC 1.2: No se identifica ningún error | El sistema comprueba que no existen errores durante el funcionamiento del canal. | Clic en “Cargar fichero xml de configuración”<br>Clic en “Cargar fichero log de Administración” |

**Descripción de las variables:**

*Tabla 9: Variables CUS Detectar error.*

| No | Nombre de Campo | Clasificación | Valor Nulo | Descripción |
|---|---|---|---|---|
| 1 | Errores | boolean | No | Permite conocer si ocurrió algún error durante el funcionamiento del canal. Puede tomar valor true o false. |

**Matriz de Datos:**

*Tabla 10: Matriz de datos de la SC 1 Detectar error.*

| ID del Escenario | Escenario | Variable 1<br>Error | Respuesta del Sistema | Resultado de la Prueba |
|---|---|---|---|---|
| EC 1.1 | Se detecta alguno de los tipos de errores. | True | El sistema muestra un listado con los errores detectados clasificados por tipo, estos errores son almacenados en la base de datos. El sistema envía una alerta por correo electrónico informando de los errores. | Satisfactorio. |
| EC 1.2 | No se identifica ningún error. | True | N/A | Satisfactorio. |

**Caso de uso:** Configurar recurso

# Capítulo 3.Implementación y Prueba

**Descripción General:**

El caso de uso inicia cuando el actor Usuario accede a la funcionalidad “Establecer límites de consumo de RAM y CPU”, el sistema muestra una interfaz con dos campos de entrada para que el usuario establezca los valores.

**Condiciones de ejecución:**

Solo se ejecuta esta acción cuando el sistema esté en funcionamiento y el actor Usuario desee establecer los límites de consumo de RAM y CPU de la Transmisión.

**Escenarios a Probar en el CUS:**

*Tabla 11: Escenarios CUS Configurar recurso.*

| Nombre de la sección | Escenarios de la sección | Descripción de la funcionalidad | Flujo central |
|---|---|---|---|
| SC 1: Configurar recurso | EC 1.1: Se establecen límites para medir el consumo de los recursos de la Transmisión. | El sistema le brinda al usuario la opción de establecer límites por el cual deben medirse el consumo de RAM y CPU de la Transmisión. | Clic en “Establecer límites de consumo de RAM Y CPU”<br>Definir valor límite para el consumo de RAM.<br>Definir valor límite para el consumo de CPU.<br>Clic en “Aceptar” |
| | EC 1.2: No se establecen los límites para medir el consumo de los recursos de la Transmisión | El sistema continúa midiendo el consumo de los recursos de la Transmisión por los últimos valores establecidos. | Clic en “Establecer límites de consumo de RAM Y CPU”<br>Definir valor límite para el consumo de RAM.<br>Definir valor límite para el consumo de CPU.<br>Clic en “Cancelar” |

**Descripción de las variables:**

*Tabla 12: Variables CUS Configurar recurso.*

| No | Nombre de Campo | Clasificación | Valor Nulo | Descripción |
|---|---|---|---|---|
| 1 | Recursos | numérico | No | Permite establecer un límite para medir el consumo de recursos de la Transmisión. |

**Matriz de Datos:**

*Tabla 13: Matriz de datos de la SC 1 Configurar recurso.*

| ID del Escenario | Escenario | Variable 1<br><br>Error | Respuesta del Sistema | Resultado de la Prueba |
|---|---|---|---|---|
| EC 1.1 | Se establecen límites para medir el consumo de los recursos de la Transmisión. | True | El sistema muestra un listado con los errores detectados clasificados por tipo, estos errores son almacenados en la base de datos. El sistema envía una alerta por correo electrónico informando de los errores. | Satisfactorio. |
| EC 1.2 | EC 1.2: No se establecen los límites para medir el consumo de los recursos de la Transmisión | True | N/A | Satisfactorio. |

Después de realizar las pruebas al sistema en la primera iteración se encontraron una serie de no conformidades que fueron corregidas, luego se procedió a realizar una segunda iteración, la cual arrojo en algunos casos no conformidades y en otros resultados satisfactorios, luego se realizó una tercera iteración donde todos los resultados fueron satisfactorios, por lo que no se necesitó la realización de una cuarta iteración. En la Fig. 11 se representa una gráfica con una comparación de los resultados arribados en la primera, la segunda y tercera iteración de pruebas.

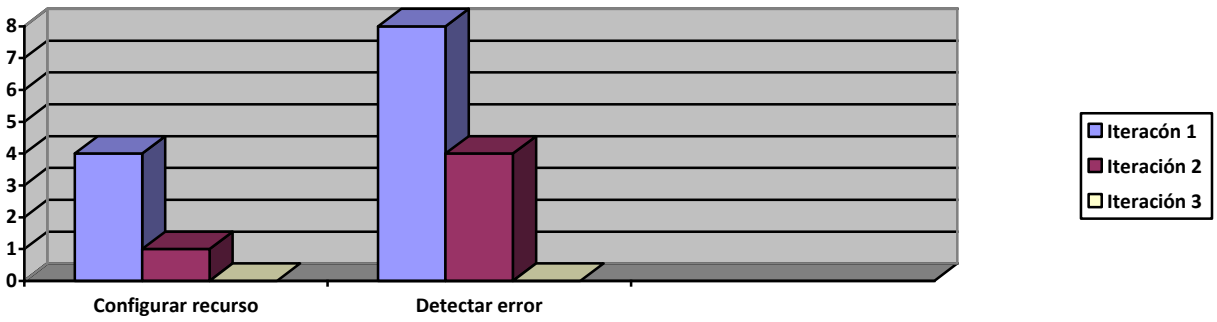



*Fig. 11: Pruebas de Iteración.*

# Capítulo 3.Implementación y Prueba

**Conclusiones del capítulo.**

En el presente capítulo se muestra el diseño detallado de la aplicación, en el cual se hicieron uso de patrones de diseño, la estructura de la arquitectura y los diagramas correspondientes, contribuyendo a un mejor entendimiento y comprensión en la implementación.

# Conclusiones Generales

## CONCLUSIONES

El Módulo para el Monitoreo de recursos y la Gestión de Errores permitirá que PRIMICIA resulte un producto más confiable pues los clientes tendrán control de los errores que se puedan producir durante la utilización de la plataforma. Por todo lo antes expuesto se puede concluir que:

- Se profundizó sobre los conocimientos relacionados con el procedimiento de seguimiento y tratamiento tanto de los errores como de los recursos en la Plataforma de Televisión Informativa PRIMICIA.
- Se modeló adecuadamente el sistema que posibilitó la automatización de las funcionalidades requeridas para el desarrollo del Módulo para el Monitoreo de los recursos y la Gestión de los errores de PRIMICIA.
- Se seleccionaron las tecnologías que sirvieron de apoyo para implementar el sistema que permite la detección de errores y el monitoreo de recursos en la plataforma PRIMICIA.
- Se desarrolló el prototipo funcional de un sistema libre para la gestión de los errores y el monitoreo de los recursos que participan en el funcionamiento de PRIMICIA.
- Se le realizaron los diferentes tipos de pruebas necesarias para comprobar el buen funcionamiento del sistema.

Después de lo visto anteriormente se puede afirmar que el objetivo de la investigación fue cumplido pues se logró desarrollar el Módulo para el Monitoreo de recursos y la Gestión de los Errores ampliando de esta forma las funcionalidades que brinda hoy la Plataforma de Televisión Informativa PRIMICIA.

## RECOMENDACIONES

- Para garantizar la seguridad, se recomienda desarrollar una interfaz que permita al usuario autenticarse para luego acceder a las funcionalidades que presenta el sistema.

- Se recomienda desarrollar una funcionalidad que le permita al usuario realizar una búsqueda de los errores ocurridos por tipo y fecha.

# Referencias bibliográficas

## Referencias

**Alebentele. 2014.** Alebentele.com. [En línea] 15 de 7 de 2014. [Citado el: 11 de 12 de 2015.] http://www.alebentelecom.es/servicios-informaticos/faqs/reparar-errores-de-arranque-windows-7-chkdsk.

**Alegsa, Leandro. 2009.** alegsa.com. [En línea] 2009. [Citado el: 7 de 4 de 2016.] http://www.alegsa.com.ar/Dic/requisitos.php..

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

**Benitez, Frank Calzado. 2009.** *Procedimiento para el seguimiento y tratamiento de los errores de PRIMICIA.* Ciudad de la Habana : s.n., 2009.

**Blanco, Alejandro Peña. 2012.** *Desarrollo del Módulo de Gestión de Errores para el Subsistema de Transmisión de PRIMICIA.* Ciudad de la Habana : s.n., 2012.

**Carlos, Universidad Carlos lll de madrid. 2009.** *Diseño Basado en Componentes.* Madrid : Engineering Information, 2009.

**Cumare. 2014.** wordpress.com. [En línea] 2014. [Citado el: 23 de 3 de 2016.] https://ingsoftwarei2014.wordpress.com/category/framework-de-desarrollo-de-software/..

**Digia. 2012.** digia.com. [En línea] 25 de 2 de 2012. [Citado el: 3 de 25 de 2016.] http://qt.digia.com/Product/.

**Esteban, Luis Grifoll. 2006.** *Plataforma Communi. TV. Gestión de contenidos multicanal y servicos interactivos.* 2006.

**Ferré, Xavier Grau y Sanchez, María Isabel Segura. 2011.** *Desarrollo Orientado a Objetos con UML.* 2011.

**GestioPolis. 2003.** GestioPolis.com. [En línea] 13 de 3 de 2003. [Citado el: 13 de 12 de 2015.] http://www.gestiopolis.com/que-es-proceso-administrativo/.

**Gil, Aros Celio. 2008.** *RUP: Metodología en los sistemas y aplicaciones basadas en la web.* 2008.

**Hernández, Rolando Alfredo León. 2002.** *Metodología de la Investigación Científica.* Ciudad de la Habana : s.n., 2002. ISBN: 959-16-0343-6.

# Referencias bibliográficas

**IDRIC. 2016.** [En línea] 2016. [Citado el: 11 de 1 de 2016.] 2016 http://www.idric.com.mx/log-siem/alertas-eventos-tiempo-real.html .

**joshell. 2010.** es.slideshare.ne. [En línea] 26 de 9 de 2010. [Citado el: 14 de 3 de 2016.] http://es.slideshare.net/joshell/diagramas-uml-componentes-y-despliegue..

**Lamarca, María Jesús Lapuente. 2013.** [En línea] 8 de 12 de 2013. [Citado el: 17 de 12 de 2015.] http://www.hipertexto.info/documentos/xml.htm.

**Larman, Craig. 2003.** Webmaster. [En línea] 2003. [Citado el: 7 de 4 de 2016.] http://is.ls.fi.upm.es/docencia/is2/documentacion/ModeloDominio.pdf..

**Linux, Red Hat Enterprise. 2003.** [En línea] 2003. [Citado el: 17 de 12 de 2015.] http://web.mit.edu/rhel-doc/3/rhel-sag-es-3/index.html.

**Martínez, Rafael Guerrero. 2010.** postgressql.org. [En línea] 2 de 10 de 2010. [Citado el: 2 de 4 de 2016.] http://www.postgresql.org.es/sobre_postgresql..

**Menéndez, Evelyn Alonso. 2009.** Monografías.com. [En línea] 2009. [Citado el: 18 de 1 de 2016.] http://www.monografias.com/trabajos73/herramientas-case-proceso-desarrollo-software/herramientas-case-proceso-desarrollo-software2.shtml..

**Montesde Oca, Dorgis Gómez. 2012.** biblioteca.uci.cu. [En línea] 6 de 2012. http://bibliodoc.uci.cu/RDigitales/2013/febrero/12/TD_05788_12.pdf.

**Murcko, Thomas. 2015.** bussinessdictionary.com. [En línea] 2015. [Citado el: 23 de 10 de 2015.] http://www.businessdictionary.com/definition/error.html..

**Navarro, Diosveldy Lores y Vicente, Francisco Pémberton. 2012.** eumed.net. [En línea] 3 de 2012. http://www.eumed.net/rev/cccss/19/nlpb2.html.

**Padilla, Luis. 2010.** Ingeniería de software.UNIMET. [En línea] 10 de 4 de 2010. [Citado el: 15 de 12 de 2015.] http://ccaneloningsoftware.blogspot.com/2010/04/segunda-actividad-virtual-metodologias.html.

**Pérez, Damian Valdés. 2007.** maestrosdelweb.com. [En línea] 26 de 10 de 2007. [Citado el: 29 de 11 de 2015.] http://www.maestrosdelweb.com/que-son-las-bases-de-datos/...

**Pressman, Roger .S. 2005.** *Ingenieria del Software. Un Enfoque Práctico.* 2005.

**—. 2005.** *Ingeniería del Software. Un enfoque práctico.* 2005.

# Referencias bibliográficas

**—. 2005.** *Ingeniería del Software. Un Enfoque Práctico. 5ta Edicion.* 2005.

**Pressman, Roger S. 2008.** *Ingeniería de Software: un enfoque práctico.* s.l. : SBN: 970-10-5473-3, 2008.

**Quesada, Juan Antonio López. 2007.** Universidad de Murcia. Departamento de informática y sistemas. Lenguajes de marcas y sistemas de gestión de información. [Citado el: 22 de febrero de 2015.]. [En línea] 2007. [Citado el: 9 de 4 de 2016.] http://dis.um.es/~lopezquesada/documentos/IES_1415/L.

**Quintana, Yoandri Rondón, Camejo, Lianet Dominguez y Díaz, Abel Berenguer. 2011.** *El diseño de la Base de Datos para Sistemas de Digitalización y Gestión de Medias.* Ciudad de la Habana : s.n., 2011. ISSN 1667.

**Romero, Felix Ivan Rodriguez, y otros. 2012.** Academia. [En línea] 10 de 2012. [Citado el: 13 de 12 de 2015.] http://www.academia.edu/11032498/PLATAFORMA_DE_TELEVISI%C3%93N_INFORMATIVA_PRIMICIA_V1.7.

**STROUSTRUP, Bjarne. Wilmington, Delaware. 2013.** *El Lenguaje de Programación C++, Segunda Edición.* E.U.A : Addison-Wesley Iberoamericana, 2013. ISBN 201-60104-4.

**Torjo, John. 2007.** torjo.com. [En línea] 2007. [Citado el: 23 de 11 de 2015.] http://torjo.com/log2/doc/html/index.html john Torjo 2007..

**Villa, Madelys Cuesta. 2007.** Repositorio Institucional de la Universidad de las Ciencias Informáticas. Modelo para la ayuda a la toma de en la selección de patrones de desarrollo. [En línea] 2007. [Citado el: 22 de 4 de 2016.] http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_0261_07/1/TD_0261_07.pdf..

**Wanta, Dave. 2004.** Sistem.net.mail. [En línea] 2004. [Citado el: 17 de 5 de 2016.] http://www.systemnetmail.com/faq/2.4.aspx.