Universidad de las Ciencias Informáticas
FACULTAD 6

Título: Subsistema de visualización de noticias para PRIMICIA v2.0

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Susana Yaque Rivera.

Aramis Romero Carballea.

Tutora: Ing. Yanary Hernández Sosa.

La Habana, Junio 2013

"Año 55 de la Revolución"

DECLARACIÓN DE AUTORÍA

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

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

Susana Yaque Rivera

_____

Firma del Autor

Aramis Romero Carballea

_____

Firma del Autor

Ing. Yanary Hernández Sosa

_____

Firma del Tutor

DATOS DE CONTACTO

Tutora: Ing. Yanary Hernández Sosa.

Graduada de Ingeniera en Ciencias Informáticas en el año 2012. Trabaja en el proyecto Plataforma de Televisión Informativa, PRIMICIA, del centro Geoinformática y Señales Digitales (GEYSED) de la Facultad 6 en la Universidad de las Ciencias Informáticas (UCI).

Correo electrónico: yanary@uci.cu

Teléfono: 835-8861

AGRADECIMIENTOS

Susana:

A mi mamá por ser mi amiga, confidente, cómplice, mi guía, mi ejemplo a seguir. Por ponerme siempre en primer lugar y por delante de todo. Por sus consejos oportunos. Porque siempre me ha señalado el camino que debo seguir pero sin imponerme nada y aunque en ocasiones no he tomado las mejores decisiones, siempre he podido contar con su apoyo incondicional. Porque sé que siempre va a estar ahí para mí cuando lo necesite.

A mi papá por su preocupación constante de mis estudios, por su insistencia para que entrara a esta universidad, por creer que era capaz de vencer los obstáculos que tenía delante para llegar a alcanzar estos resultados. Porque me sirvió de motivación pues siempre quise llegar a ser tan buena en mi profesión como yo considero que él lo es.

A mis abuelos Clara y Giraldo que son mis segundos padres, que aunque no están aquí en estos momentos, estoy 100% segura de que están pensando en mí y deseándome lo mejor del mundo. Porque ambos han estado siempre presente en mi vida preocupándose por mí.

A mis hermanos, por inspirarme a superarme, pues quiero que vean en mí un ejemplo a seguir.

A Magdiel porque ha sido como un padre para mí y por cuidar de mis seres más queridos cuando no he estado cerca.

A mis suegros que se han portado maravillosamente conmigo, haciéndome sentir parte de su familia, tratándome como tratan los padres a los hijos. Así es como me han hecho sentir, por eso les doy las gracias.

A mi novio y más cercano colaborador Rami, por estar siempre ahí cuando lo he necesitado. Por saber diferenciar cuando necesito al amigo y cuando al novio. Por siempre confiar en mí y darme ánimos cuando las cosas no iban bien y me parecía que todo se derrumbaba, dándome apoyo y fuerzas para seguir adelante. Por haber compartido estos cinco años conmigo. Por darme todo su amor, comprensión y apoyo en

estos momentos de tanta tensión. Por ser guía de todos mis esfuerzos durante la carrera y por ser ese hombre que tanto admiro y quiero. Por todos los momentos maravillosos que he pasado a su lado, por todo el cariño que me ha entregado. Eres especial, te quiero mucho.

A todos los profesores del proyecto que han ayudado en este trabajo.

A la tutora por la ayuda ofrecida y la guía durante el desarrollo de la investigación.

A todos los miembros del tribunal, por haber puesto todo su empeño y dedicación en perfeccionar esta investigación hasta convertirla en lo que es hoy.

Al oponente por las críticas que ayudaron mucho a mejorar el trabajo de diploma.

A todos los que hoy no menciono y me ayudaron.

Aramis:

A mis padres y a mi abuela, por todo su apoyo. Por preocuparse más que yo por mis estudios.

A Susana, por estos años de cariño y apoyo. Por aceptar mis defectos y agradecer mis virtudes.

A Clarita, Yaque y Magdiel por estar siempre ahí para ayudar.

A Pupo, Yanary, Rafa, Carlos, Félix y a todos los que ayudaron en este trabajo.

A todos mis amigos que me han tendido la mano cuando la he necesitado.

DEDICATORIA

A nuestra familia.

RESUMEN

La Plataforma de Televisión Informativa, PRIMICIA, es una solución integral que permite la transmisión automática y constante de informaciones en diferentes formatos a través de una red de televisión. Esta aplicación informática se encuentra en un constante perfeccionamiento y análisis para ir mejorando sus funcionalidades. En este trabajo se propone una nueva versión del subsistema de visualización de noticias para el proyecto Plataforma de Televisión Informativa, PRIMICIA, basada en la utilización de tecnología QML, provista por el framework QT, aplicando la concepción de elementos y efectos visuales dinámicos y configurables. Esta nueva versión permitirá la visualización de noticias de diferentes secciones temáticas de forma alterna e informaciones adicionales a la noticia durante el proceso de transmisión del canal, con el objetivo de que el proyecto pueda adquirir nuevos clientes y de brindar a los usuarios la posibilidad de mantenerse actualizados sobre múltiples informaciones a la vez.

Palabras claves: canal, información, televisión.

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS Y FIGURAS

INTRODUCCIÓN

Desde comienzos de la humanidad el hombre se ha visto en la necesidad imperante de comunicarse, lo cual constituye un elemento clave en el desarrollo y la evolución social del mismo, dicha necesidad ha contribuido a que los diferentes medios de comunicación avancen de manera acelerada en búsqueda de su perfeccionamiento. Hoy se cuenta con medios de comunicación que permiten la transmisión de la información a gran parte del mundo, con rapidez, calidad, brindándole a la sociedad el acontecer mundial en diferentes esferas: cultura, salud, educación, tecnología, política, economía. Los acontecimientos importantes, novedosos e imprevistos ocurridos en el mundo son informados a la sociedad a través de los medios de comunicación que posibilitan una correcta información y una mayor socialización de los mismos.

La televisión es un medio de comunicación de gran aceptación en la actualidad, por la capacidad que posee de llegar a millones de personas de forma simultánea. Permite la difusión de la información y constituye una herramienta para la mejora de las comunicaciones en la sociedad, convirtiéndose en una vía para canalizar el gran mercado de la información. Esta brinda entretenimiento e información a miles de televidentes con intereses y gustos diferentes vinculando la imagen y el sonido, lo que ayuda a conquistar con mayor rapidez al público. En los últimos años han surgido nuevas formas de hacer televisión, gracias al desarrollo tecnológico que ha alcanzado este medio y a la vinculación con campos como la informática y las telecomunicaciones.

Cuba no se ha mantenido exenta a los avances tecnológicos de la televisión, mejorando así sus transmisiones. No solo se ha interesado por las transmisiones sino por la informatización de los procesos que se realizan dentro de esta, para lo cual se han creado centros de desarrollo que se vinculan al trabajo directo con el procesamiento de señales digitales y materiales audiovisuales. Entre estos centros se encuentra la Universidad de las Ciencias Informáticas (UCI) y dentro de este, el centro de desarrollo de Geoinformática y Señales Digitales (GEYSED). Señales Digitales es un departamento de dicho centro, el cual cuenta con varios proyectos. Un ejemplo de ello lo constituye la Plataforma de Televisión Informativa, PRIMICIA, la cual puede ser personalizada a cualquier cliente que posea la necesidad de mantener informado constantemente a determinada cantidad de personas. Esta plataforma se mantiene en un constante desarrollo, concentrándose en la búsqueda de nuevas funcionalidades y en mejoras de las ya existentes.

Para el surgimiento de PRIMICIA se tuvo en cuenta otras soluciones informáticas que anteriormente habían sido creadas para la transmisión de noticias como son: el sistema informativo elaborado para la red de televisión interna de la UCI “Señal 3”, el sistema para la transmisión de noticias a los colaboradores cubanos en el exterior y habitantes de las zonas montañosas de difícil acceso a la televisión terrestre y la prensa escrita en Cuba “Señal ACN[1]” y el sistema desarrollado para mantener informados a los trabajadores y visitantes de la sede central del Ministerio del Poder Popular para Energía y Petróleo de Venezuela, MENPET, reconocido como “TV Energía”.

Este producto constituye un sistema multiplataforma, desarrollado sobre tecnologías libres. Cuenta con funcionalidades genéricas fácilmente escalables, que no dependen de un entorno dado y que no están ligadas a un diseño gráfico específico. Este sistema está estructurado en dos subsistemas, administración y transmisión, los cuales se relacionan trabajando como un solo sistema para lograr mostrar las noticias a través de la red de televisión de forma satisfactoria. A través del subsistema de administración se realiza la administración del canal y toda la gestión de las noticias y recursos multimedia. La visualización de las noticias y los materiales publicados es responsabilidad del subsistema de transmisión.

PRIMICIA realiza la transmisión de las noticias por secciones temáticas. Este tipo de transmisión no permite visualizar noticias de diferentes secciones temáticas de forma alterna, pues establece el orden en que serán mostradas las secciones, se comienza con la primera sección temática y las noticias asociadas a la misma, una vez que termine con esta pasa a la otra y así sucesivamente, creando un ciclo mientras exista vigencia en las noticias publicadas. Durante el proceso de transmisión del canal solo se visualizan las noticias e infocintas que han sido redactadas en el subsistema de administración, o sea, no permite que se muestren informaciones adicionales de forma simultánea como: pronósticos meteorológicos, resultados deportivos, resúmenes culturales, precios de productos u otro tipo de informaciones adquiridas; permitiendo mantener actualizados a los televidentes del canal sobre múltiples informaciones a la vez.

A partir de la situación problemática planteada anteriormente se formula como **problema a resolver:** La Plataforma de Televisión Informativa, PRIMICIA, no transmite noticias de diferentes secciones temáticas de forma alterna, ni informaciones adicionales a la noticia durante el proceso de transmisión del canal.

[1] Agencia Cubana de Noticias.

Atendiendo al problema planteado y sobre la base de la necesidad de investigación, se tiene como **objeto de estudio:** proceso de visualización de noticias en plataformas de televisión. Para ello se señala como **campo de acción:** proceso de visualización de noticias en PRIMICIA v2.0.

Para dar solución al problema se establece como **objetivo general**: Desarrollar el subsistema de visualización de noticias para la Plataforma de Televisión Informativa, PRIMICIA v2.0.

A partir del objetivo general planteado se defiende la idea de que: Con el desarrollo de una nueva versión del subsistema de visualización de noticias en PRIMICIA, se logrará transmitir noticias de diferentes secciones temáticas de forma alterna e informaciones adicionales a la noticia durante el proceso de transmisión del canal.

Para dar solución al objetivo general se deben realizar las siguientes **tareas de la investigación**:

1. Descripción del estado del arte a nivel nacional e internacional de los sistemas existentes dedicados a la visualización de noticias en plataformas de televisión.
2. Caracterización de la metodología y las herramientas a utilizar durante el proceso de desarrollo.
3. Definición de los requisitos funcionales y no funcionales del sistema.
4. Realización del diseño ingenieril de los procesos.
5. Implementación de las funcionalidades diseñadas del subsistema de visualización de noticias de la Plataforma de Televisión Informativa, PRIMICIA.
6. Diseño y aplicación de las pruebas al sistema.

Se utilizaron los siguientes **métodos científicos**:

- ✓ **Métodos teóricos:**

  Análisis Histórico-Lógico: Permitió realizar un análisis profundo sobre la evolución que han tenido los sistemas de visualización de noticias a nivel nacional e internacional.

Analítico-Sintético: Permitió el análisis de los procesos de visualización de noticias que presentan diversos sistemas informativos, además de lograr una síntesis de los elementos significativos de la investigación.

- ✓ **Métodos empíricos**:

Observación: Este método permitió recopilar información acerca del comportamiento funcional de PRIMICIA en los actuales ambientes de desarrollo.

Observación bibliográfica: Este método permitió recoger información de los aspectos tratados por otros autores y así con el estudio de sus definiciones y resultados poder obtener conocimiento sobre los procesos de visualización de noticias.

El presente documento está constituido por los siguientes capítulos:

En el Capítulo I **Fundamentos teóricos del subsistema de visualización de noticias de la Plataforma de Televisión Informativa, PRIMICIA**, se explican los diferentes conceptos asociados al dominio del problema, tales como: televisión, transmisión, infocinta y noticia. Además se describe el proceso de visualización de noticias de la Plataforma de Televisión Informativa, PRIMICIA, así como el análisis realizado a algunas de las soluciones existentes a nivel internacional y nacional que se encargan de la transmisión de noticias. Se realiza una caracterización de la metodología a utilizar, así como de las herramientas, técnicas y tecnologías.

En el Capítulo II **Características del subsistema de visualización de noticias de la Plataforma de Televisión Informativa, PRIMICIA**, se realiza un análisis del sistema que se desea construir y se conceptualiza el entorno en un modelo de dominio para lograr un mayor acercamiento al problema a resolver. Además se realiza la captura de los requisitos funcionales y no funcionales. También en este capítulo queda plasmado el diagrama de casos de uso del sistema y las descripciones de sus actores y casos de uso.

En el Capítulo III **Análisis, Diseño e Implementación del subsistema de visualización de noticias de la Plataforma de Televisión Informativa, PRIMICIA**, se especifican los artefactos generados correspondientes a los flujos de trabajo Análisis y Diseño e Implementación según la metodología de

desarrollo utilizada, que garantizan un mejor entendimiento y una correcta implementación del sistema propuesto.

En el Capítulo IV **Validación y pruebas realizadas al subsistema de visualización de noticias de la Plataforma de Televisión Informativa, PRIMICIA**, se realiza el proceso de validar el producto desarrollado para la Plataforma de Televisión Informativa, PRIMICIA, a partir de las pruebas realizadas a la aplicación y de los resultados obtenidos en las mismas. Se tiene como principal objetivo en este capítulo verificar si el sistema cumple con las especificaciones planteadas por el cliente.

Además se brindan en este documento conclusiones, recomendaciones, bibliografías, anexos y un glosario de términos.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS DEL SUBSISTEMA DE VISUALIZACIÓN DE NOTICIAS DE LA PLATAFORMA DE TELEVISIÓN INFORMATIVA, PRIMICIA

1.1 Introducción

Este capítulo tiene como objetivo fundamental abarcar los conceptos y argumentos teóricos asociados al dominio del problema para lograr una mejor comprensión del contenido de la investigación. Se realiza un estudio sobre las soluciones existentes en el ámbito internacional y nacional dedicadas a la transmisión de noticias. Además se describe el proceso de visualización de noticias de la Plataforma de Televisión Informativa, PRIMICIA. Finalmente se caracteriza la metodología y las tecnologías que se van a utilizar para la construcción del subsistema de visualización de noticias.

1.2 Conceptos asociados al dominio del problema

A continuación se presentan un conjunto de conceptos que permitirán tener un mayor dominio teórico del proceso de visualización de noticias, lo cual es primordial para el desarrollo del trabajo pues la información obtenida en la investigación servirá de base para los resultados esperados.

Televisión

Se define como televisión un sistema de telecomunicaciones para la transmisión y recepción de imágenes en movimiento y sonido a distancia. Esta transmisión puede ser efectuada mediante ondas de radio o redes especializadas de televisión por cable, donde el receptor de las señales es el televisor (1).

La televisión mecánica y la electrónica fueron los primeros modelos inventados por el hombre, ambas se desarrollaron de forma paralela y por diferentes compañías e inventores en búsqueda de un único sistema capaz de transmitir imágenes a distancia (2).

Transmisión

Según la RAE[2], transmisión es la acción y efecto de transmitir. Definiendo transmitir como hacer llegar a alguien mensajes o noticias (3).

[2] Real Academia Española.

La transmisión puede ser considerada además como la transferencia de datos a través de un canal de comunicaciones (4).

Según *The Free Dictionary* la transmisión es el acto o proceso de enviar un mensaje, imagen u otra información desde una ubicación a una o más ubicaciones por medio de ondas de radio, señales eléctricas, señales luminosas (5).

Por su parte *WordReference* conceptualiza transmisión como la difusión de una señal de radio o video (6).

Enfocándose en el mundo de las señales digitales televisivas, transmisión sería la recepción y envío de una señal digital de video o audio a través de una red de datos.

Noticia

Según *The Free Dictionary* la noticia es la comunicación o divulgación de un suceso reciente, en especial la que se hace a través de un medio de comunicación (5).

Infocinta

Elemento visual situado generalmente en el borde inferior de la pantalla, utilizado para transmitir una noticia importante sin necesidad de afectar las noticias principales que se estén transmitiendo.

1.3 Análisis de soluciones existentes

1.3.1 Ámbito internacional

En el ámbito internacional existen soluciones para la transmisión de información rápida, servicios que se valen de la televisión como soporte y que tienen como base un sistema automático. Ejemplos de estas soluciones son los sistemas de teletextos que han adoptado, principalmente, las cadenas de televisión digital de países europeos.

"El teletexto es un medio de comunicación gratuito con vocación de servicio público que se sirve de la televisión para hacer llegar sus mensajes a los usuarios. En él conviven contenidos editoriales y comerciales, ofreciéndole al receptor una información completa de toda índole y fácil de consultar. Se podría comparar con un gran periódico de más de 2.000 páginas que el usuario puede seleccionar y leer

cuantas veces desee. Su característica principal es que, a diferencia del periódico tradicional, su información está en constante renovación" (7).

*Fingertext***,** plataforma de procedencia española, es un sistema modular de teletexto y subtitulación por teletexto (8). El sistema tiene las siguientes características:

-Permite que los usuarios se autentiquen a través de la web. Verificando que sólo puedan realizar las tareas permitidas según su perfil. El cual define las carpetas y contenidos sobre los que puede actuar y cuáles son las operaciones que puede realizar. Estas carpetas tendrán las páginas y subtitulaciones que posteriormente serán mostrados a los receptores.

-Brinda la posibilidad de la administración y gestión del sistema completo desde cualquier ubicación física que tenga acceso al servidor.

-Permite la creación de páginas, utilizando para ello un editor de teletexto, el cual permite que estas sean creadas y editadas (cuando se habla de página se está haciendo alusión al teletexto que será mostrado en los televisores).

-El sistema además da la posibilidad a los usuarios con ciertos privilegios de editar las páginas de teletexto, poder eliminarlas, moverlas o cambiar su contenido, estas mismas operaciones pueden ser realizadas con las subtitulaciones (9).

Este sistema de teletexto implementado en Europa brinda enormes ventajas desde el punto de vista de la televisión digital, organización por secciones y actualización automática mediante un software que facilita este trabajo. Sin embargo esta plataforma no soporta algunas de las funcionalidades importantes a tener en cuenta en la solución a la problemática planteada. Entre estas funcionalidades se encuentran:

-Tienen como principio fundamental que el usuario es el encargado de decidir mediante su mando a distancia qué noticias desea ver dentro del sin número de informaciones que se ofrecen, lo que hace que no se ajuste, debido que se requiere que las noticias sean transmitidas sin la necesidad de la interactividad con el usuario.

-Las noticias soportan varios formatos de pantalla (texto, imagen, video). En este caso no se necesita de ningún software para realizar esta operación, todo el trabajo es realizado dentro de la misma aplicación dinámicamente.

-La emisión del propio teletexto se realiza junto con la señal televisiva haciendo uso de las llamadas líneas del intervalo de cancelación de cuadro de la señal de video, es decir, las líneas no utilizadas para la señal de imagen de televisión. Esta característica limita en gran parte la información que se puede emitir, admitiendo sólo textos y mínimamente gráficos, además no es compatible con los requerimientos del problema debido a que se necesita un producto que resulte atractivo para los televidentes y que permita la utilización de imágenes y cortos televisivos en formato de video, posibilitando de esta manera la transmisión de informaciones en distintos formatos.

Por lo antes expuesto se concluye que este sistema de teletexto no puede dar solución al problema planteado.

1.3.2 Ámbito nacional

El sistema informativo creado para la red de televisión interna de la UCI “Señal 3” está compuesto por dos módulos: administración y transmisión. Esta aplicación permite: una fácil gestión a través de la web, transmisión de texto y recursos multimedia por medio de la televisión, organización por secciones, empleo de múltiples tipos de noticias y un fondo musical. El canal no está orientado solo hacia la reproducción de noticias, sino incluye también informaciones de facilitación social en la comunidad universitaria, como pérdidas de objetos personales, felicitaciones, convocatorias, lo cual imposibilita resolver el problema planteado con el apoyo de este sistema.

El sistema para la transmisión de noticias a los colaboradores cubanos en el exterior y habitantes de las zonas montañosas de difícil acceso a la televisión terrestre y la prensa escrita en Cuba “Señal ACN”, al igual que Señal 3, está desarrollado completamente utilizando software propietario y ninguna de los dos permite realizar transmisiones en vivo.

Estos sistemas resuelven parcialmente el problema presente, tienen características que se aproximan a lo deseado como son: transmisión de textos a través de la televisión, utilización de recursos multimedia, empleo de distintos tipos de noticias, fondo musical. Sin embargo, no pueden ser usados para dar

solución al problema planteado, al no permitir visualizar noticias de diferentes secciones temáticas de forma alterna a la noticia, ni informaciones adicionales a la misma y además por no disponer de tecnologías que faciliten el uso de los elementos y efectos visuales dinámicos y configurables.

1.4 Proceso de visualización de noticias de la Plataforma de Televisión Informativa, PRIMICIA

La Plataforma de Televisión Informativa, PRIMICIA, contempla la administración y transmisión de noticias en diferentes formatos mediante un canal de televisión. La transmisión del canal se realiza por secciones temáticas, la cual no permite visualizar noticias de diferentes secciones temáticas de forma alterna, ya que establece el orden en que serán mostradas las secciones, se comienza con la primera sección temática y las noticias asociadas a la misma, una vez que termine con esta pasa a la otra y así sucesivamente, creando un ciclo mientras exista vigencia en las noticias publicadas.

Este tipo de transmisión tampoco permite que se muestren informaciones adicionales a la noticia de forma simultánea como: pronósticos meteorológicos, resultados deportivos, resúmenes culturales, precios de productos u otro tipo de informaciones adquiridas; permitiendo mantener actualizados a los televidentes del canal sobre múltiples informaciones a la vez, ya que durante el proceso de transmisión del canal solo se visualizan las noticias e infocintas que han sido redactadas en el subsistema de administración.

En PRIMICIA v2.0 las noticias serán gestionadas desde el subsistema de administración de la plataforma, cuando se redacten se asignarán a un bloque de noticias y luego serán transmitidas. Este tipo de transmisión funcionará estableciendo el horario en que saldrá al aire un bloque de noticias y durante ese tiempo se mostrarán todas las noticias asociadas al mismo, creando de esta manera un ciclo durante el horario establecido. Durante este ciclo se mostrarán además las infocintas que hayan sido publicadas. Cuando concluya el período establecido se dejarán de transmitir.

Además este sistema permitirá añadir nuevos componentes (componentes para mostrar el tiempo u otras informaciones adicionales) a los ya definidos (texto, imagen y video). De esta manera se daría solución al problema planteado pues se obtendría un producto capaz de permitir la visualización de noticias de diferentes secciones temáticas de forma alterna e informaciones adicionales a la noticia durante el proceso de transmisión del canal.

1.5 Metodología de desarrollo de software RUP[3]

Una metodología de desarrollo de software es un conjunto de procedimientos, técnicas, herramientas y soporte documental que guían el proceso de fabricación de una aplicación informática, arrojando un producto de calidad y en el tiempo requerido (10).

Las metodologías de desarrollo de software se clasifican en dos grandes grupos, robustas o ágiles, ambas tienen como principal utilidad organizar y guiar un equipo de proyecto. Las robustas se centran específicamente en el control de los procesos, demostrando ser efectivas y necesarias en proyectos de gran tamaño respecto a tiempo y recursos. Las ágiles ofrecen una buena solución para aquellos proyectos donde los requisitos no se conocen con exactitud y pueden sufrir cambios.

Para la elección de una metodología de desarrollo para el proyecto PRIMICIA se realizó un análisis de las más usadas y conocidas, con el objetivo de determinar cuál se ajusta más a sus características.

Se descartaron las metodologías ágiles, pues se cuenta con un equipo de trabajo amplio que no posee una vasta experiencia en el desarrollo de software. El equipo de desarrollo es joven, inexperto y está compuesto principalmente por estudiantes, esto implica la posibilidad de que ocurran cambios de roles y la necesidad de incorporar nuevos miembros al proyecto, por lo que se requiere una buena documentación para garantizar la realización de futuras versiones con integrantes nuevos en el proyecto.

Se determinó que es necesario usar una de las metodologías robustas, pues estas confieren gran peso a la planificación, conceptualización y descripción del sistema que se va a diseñar. *Rational Unified Process* (RUP) y *Microsoft Solution Framework* (MSF) son dos de las metodologías más conocidas y usadas.

Para el desarrollo de PRIMICIA v2.0 se utiliza como sistema operativo Ubuntu, por tanto las tecnologías que se utilicen tienen que ser libres o al menos multiplataforma. MSF es una metodología desarrollada por Microsoft, lo que provoca un alto grado de dependencia de tecnologías propietarias. Con RUP se puede utilizar *Visual Paradigm for UML*, que es una herramienta multiplataforma, la cual brinda excelentes resultados en el modelado y ha sido utilizada en la modelación de los artefactos del proyecto durante el desarrollo de las versiones anteriores de la plataforma.

---

[3] Rational Unified Process en inglés, habitualmente resumido como RUP.

RUP es la metodología escogida para la generación de artefactos de PRIMICIA v2.0 por ser la que sustenta el desarrollo general del producto. Genera una gran cantidad de artefactos, lo cual representa una garantía para la continuidad del trabajo del proyecto y la futura implementación de las funcionalidades modeladas, no depende de la presencia constante del cliente, permite el desarrollo de proyectos a largo plazo y además cuenta con una documentación detallada.

1.6 Herramientas, Técnicas y Tecnologías

Las herramientas, técnicas y tecnologías que se presentan a continuación fueron escogidas por el arquitecto de software del proyecto PRIMICIA, lo cual se tuvo en cuenta para la selección de las mismas para PRIMICIAv2.0. Esta selección se puede encontrar en el expediente del proyecto, en el documento de la arquitectura de software.

1.6.1 Lenguaje Unificado de Modelado (UML)

UML es un lenguaje de modelado visual que se usa para visualizar, especificar, construir y documentar los artefactos de un sistema de software (11). Es una notación y no un proceso/método, es decir, es una herramienta útil para representar los modelos del sistema en desarrollo. No ofrece ningún tipo de guía o criterios acerca de cómo obtener esos modelos (12).

UML es una herramienta que ayuda a capturar la idea de un sistema para comunicarla posteriormente a quien está involucrado en su proceso de desarrollo; esto se lleva a cabo mediante un conjunto de símbolos y diagramas. Cada diagrama tiene fines distintos dentro del proceso de desarrollo (13).

A través del uso de este lenguaje de modelado es posible obtener una abstracción del subsistema a desarrollar, facilitándole a los integrantes del equipo de desarrollo participar e intercomunicarse fácilmente.

1.6.2 Herramienta de modelado de software

Visual Paradigm es una herramienta profesional para el modelado con UML, ha sido diseñada para un gran número de usuarios, incluyendo ingenieros de sistemas, analistas de sistemas, analistas de negocio y arquitectos de sistemas. Permite representar todos los tipos de diagramas de clases, generar documentación y código desde diagramas (14).

Es una herramienta colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto y permite el control de versiones, los cuales son elementos a tener en cuenta en proyectos con un equipo de desarrollo bastante grande como lo es PRIMICIA. Dicha característica sustenta la elección de esta herramienta, pero la principal razón de su utilización se debe a que PRIMICIA está sustentada bajo esta herramienta y la solución que se obtenga debe ser compatible con esa estrategia. Además porque es una herramienta profesional que soporta todo el ciclo de vida del software y permite el desarrollo de aplicaciones con rapidez, facilidad y calidad con un menor costo de producción.

1.6.3 Lenguaje de programación

Haciendo uso de los lenguajes de programación se pueden crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que se ponen a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes.

Características que presenta el lenguaje C++ (15):

-Difusión: existe una gran cantidad de libros, cursos y páginas web dedicados a su estudio.

-Versatilidad: es un lenguaje de propósito general, por lo que se puede emplear para resolver cualquier tipo de problema.

-Portabilidad: está estandarizado y un mismo código fuente se puede compilar en diversas plataformas.

-Eficiencia: es un lenguaje rápido en cuanto a tiempo de ejecución.

-Herramientas: existe una gran cantidad de compiladores, depuradores y bibliotecas para el trabajo con C++.

Para la implementación del subsistema de visualización de PRIMICIA v2.0 se necesita un lenguaje de programación con el cual el equipo de desarrollo se encuentre familiarizado. Por esta razón y por las diversas potencialidades ligadas a la solución del problema a resolver, se elige C++ como lenguaje de programación.

1.6.4 Framework de desarrollo

Un framework, en el desarrollo de software es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto (16).

Qt es un framework de desarrollo de aplicaciones multiplataforma, está escrito en C++ y viene acompañado de un conjunto de herramientas para facilitar su uso. Además es posible utilizar Qt con otros lenguajes de programación a través de *bindings* (adaptación de una biblioteca para ser usada en otro lenguaje de programación).

A continuación se muestran características que presenta el framework Qt, las cuales se tuvieron en cuenta para su elección:

-Compatibilidad multiplataforma con un solo código fuente.

-Disponibilidad del código fuente.

-Excelente documentación.

El framework Qt está compuesto por:

-Las bibliotecas Qt (clases escritas en C++).

-Qt Designer: para diseñar formularios visualmente.

-Qt Assistant: acceso rápido a la documentación.

-Qt Linguist: traducción rápida de programas.

-qmake: simplifica el proceso de construcción de proyectos en las diferentes plataformas soportadas.

Para la implementación del subsistema de visualización de PRIMICIA v2.0 se elige Qt 4.8.2 como framework de desarrollo.

1.6.5 Entorno de Desarrollo Integrado (IDE)

Un entorno de desarrollo integrado (en inglés *Integrated Development Environment*) es un programa compuesto por una serie de herramientas que utilizan los programadores para desarrollar código. Esta herramienta puede estar pensada para su utilización con un único lenguaje de programación o bien puede dar cabida a varios de estos. Las herramientas que normalmente componen un entorno de desarrollo integrado son las siguientes: un editor de texto, un compilador, un intérprete, unas herramientas para la automatización, un depurador, un sistema de ayuda para la construcción de interfaces gráficas de usuario y, opcionalmente, un sistema de control de versiones (17).

Para la selección del entorno de desarrollo integrado a utilizar en la construcción de la aplicación se definieron los siguientes parámetros a cumplir:

-Facilidad de uso y abundante documentación.

-Integración con el framework de desarrollo escogido para la implementación de la aplicación.

-Soporte para el lenguaje C++.

-Integración con bibliotecas de Qt.

Qt Creator ha sido diseñado para ser un entorno de desarrollo integrado, que incluye herramientas de diseño de interfaz de usuario (18). Proporciona:

- un editor de código de C++ y JavaScript.

- un diseñador de interfaz de usuario integrado.

Estas son algunas de las características claves de "Qt Creator":

-El editor de código avanzado de Qt Creator ofrece compatibilidad con la edición de C++ y QML (JavaScript).

-Ofrece dos editores visuales integrados: Qt Designer para la creación de interfaces de usuario de *widgets* Qt y Qt Quick Designer para el desarrollo de interfaces de usuario animadas con el lenguaje QML.

-Independientemente de si se importa un proyecto existente o se crea uno desde cero, Qt Creator genera todos los archivos necesarios.

Para la implementación del subsistema de visualización de PRIMICIA se elige Qt creator 2.5.2 como IDE.

1.6.6 QML

QML es un lenguaje declarativo basado en JavaScript que permite diseñar interfaces de usuario. Se basa en declarar elementos QML en un árbol de objetos que permiten visualizar bloques, textos, gráficos, imágenes y permiten interactuar con ellos por medio de estados, animaciones, transiciones. Los elementos QML, además, pueden ser utilizados mediante el estándar JavaScript en la misma declaración QML o mediante la inclusión de ficheros JavaScript .js y pueden también ser integrados en software desarrollados con el framework Qt en C++ (19).

Con la incorporación de QML se hace más cómodo el manejo de todas las medias (texto, imagen, audio y video) necesarias en la aplicación, se aprovechan mejor los recursos de la máquina y además permite estructurar el código permitiendo darle un mejor mantenimiento.

1.6.7 Sistema Gestor de Base de Datos: PostgreSQL

Su desarrollo comenzó hace más de 16 años y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta por sus desarrolladores (20). Es un gestor de base de datos que aporta estabilidad al sistema y posibilita la consistencia de los datos almacenados.

En la construcción inicial del sistema fue utilizado PostgreSQL 9.1. Su código fuente está disponible sin costo y es multiplataforma. Por estas razones se emplea como Sistema Gestor de Base de Datos para PRIMICIA v2.0.

1.7 Conclusiones parciales

Se puede definir a PRIMICIA v2.0 como una aplicación informática capaz de visualizar informaciones a través de la televisión, tomando como base fundamental las características básicas de los sistemas que existen a nivel nacional. Sin embargo, se establecen variaciones importantes como es la visualización de noticias de diferentes secciones temáticas de forma alterna y la visualización de noticias adicionales de forma simultánea durante la transmisión del canal. La necesidad de estas nuevas funcionalidades que serán incorporadas hace que las soluciones existentes a nivel nacional e internacional no puedan ser empleadas para resolver la problemática planteada.

La utilización de RUP como metodología de desarrollo de software, UML como lenguaje de modelado y Visual Paradigm como herramienta de modelado visual trajo consigo la posibilidad de obtener un producto bien documentado en un lenguaje común, lo cual representa una garantía para la continuidad en el trabajo del proyecto.

CAPÍTULO 2: CARACTERÍSTICAS DEL SUBSISTEMA DE VISUALIZACIÓN DE NOTICIAS DE LA PLATAFORMA DE TELEVISIÓN INFORMATIVA, PRIMICIA

2.1 Introducción

Este capítulo tiene como objetivo fundamental conceptualizar el entorno en un modelo de dominio para lograr un mayor acercamiento al problema a resolver, así como realizar la captura de los requisitos para obtener la especificación de los requisitos funcionales y no funcionales que el sistema debe tener. También se detalla el diagrama de casos de uso del sistema, las descripciones de sus actores y casos de uso.

2.2 Modelo de dominio

El modelo de dominio es la representación visual de los conceptos u objetos del mundo real en un dominio de interés. En él se representan los conceptos del dominio que resultan importantes, sus características y las relaciones entre dichos conceptos. Es el punto de partida para lograr el diseño adecuado del sistema que se desea desarrollar, expresa el razonamiento obtenido del problema, además de ser considerado un diccionario visual del dominio del problema (21).

Utilizando la notación UML, un modelo de dominio se representa como un conjunto de diagramas de clases en los que no se define ninguna operación. Pueden mostrar objetos del dominio o clases conceptuales, así como asociaciones y atributos. Su objetivo es lograr la representación de las clases conceptuales del mundo real significativas en un dominio del problema, no de componentes de software, aunque algunos objetos del modelo pueden terminar siéndolo (22).

2.2.1 Principales eventos del entorno

PRIMICIA es una plataforma de televisión que tiene como objetivo mantener informada y actualizada a través de un canal de televisión a cualquier comunidad, empresa o entidad que requiera de información clara, precisa y de forma inmediata. Este sistema está estructurado en dos subsistemas, administración y transmisión, los cuales se relacionan para lograr mostrar las noticias a través de la red de televisión de forma satisfactoria, haciéndoles llegar las informaciones a los usuarios de forma clara y sencilla utilizando textos, imágenes y videos. A través del subsistema de administración se realiza la administración del

canal y toda la gestión de las noticias y recursos multimedia. La visualización de las noticias y los materiales publicados es responsabilidad del subsistema de transmisión.

2.2.2 Glosario de términos del dominio

El siguiente glosario de términos ayudará a una mejor comprensión de los conceptos presentes en el entorno del problema.

Noticia: Conjunto de información que llega a las personas de forma inmediata y actualizada. Está compuesta por pantallas y tiene asociada un fondo musical y una fecha.

Pantalla: Área que contiene información visual de la noticia a la que pertenece. Puede contener texto, imagen, video o la integración de estos.

Infocinta: Elemento visual situado generalmente en el borde inferior de la pantalla, utilizado para transmitir una noticia importante sin necesidad de afectar la noticia principal que se esté transmitiendo.

Infocinta normal: Las infocintas normales son mostradas según el orden que ocupan en la lista de infocintas.

Infocinta inmediata: Las infocintas inmediatas son mandadas a transmitir inmediatamente, luego son colocadas en la lista de infocintas en la posición siguiente a la infocinta que está siendo transmitida en ese momento.

Canales externos: Cualquier canal de televisión que no sea el generado por las noticias de PRIMICIA, los cuales sirven para enlazar la transmisión de la plataforma con estos canales en el momento deseado, ya sea para la transmisión directa de una información inmediata o simplemente para completar la programación del canal cuando no existan noticias.

Fondo musical: Instrumental que será transmitido como música de fondo mientras se muestra la noticia, excepto cuando se muestra un video.

Fecha: Cada noticia e infocinta tienen asociada una fecha en la cual serán transmitidas.

Video: Un video es un sistema de reproducción de imágenes que pueden estar acompañadas de sonidos.

Imagen: Se refiere a la figura, representación, semejanza o apariencia de algo. Una imagen también es la representación visual de un objeto a través de técnicas de la fotografía, la pintura, el diseño, el video u otras disciplinas.

Texto: Conjunto coherente de enunciados, ya sean escritos u orales. Es un componente que hace referencia a un texto determinado.

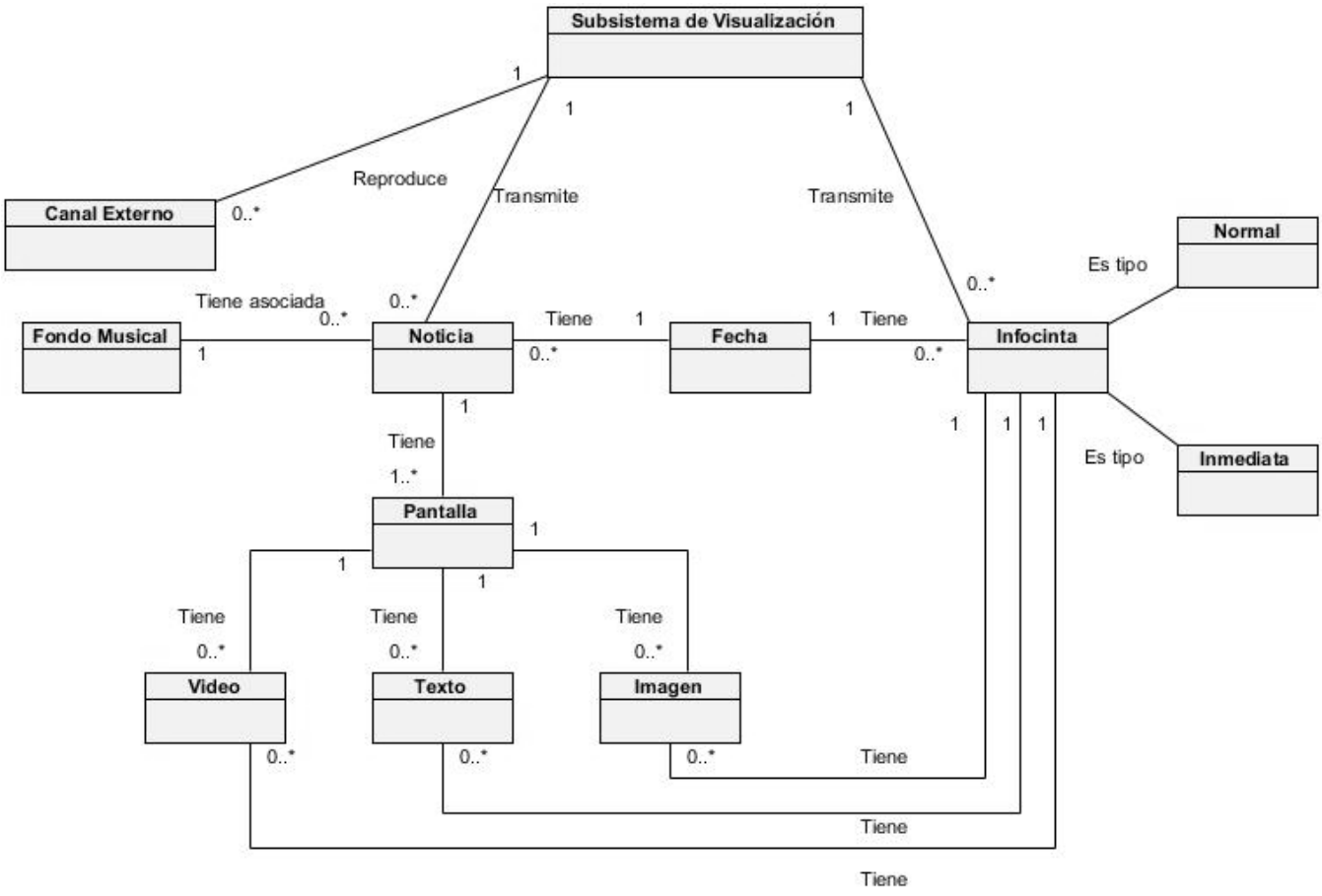

Figura 1: Modelo de dominio para el subsistema de visualización de noticias.

El subsistema de visualización se encarga de transmitir las noticias y las infocintas. Para ello crea cada una de las pantallas correspondientes a las noticias, las cuales pueden contener texto, imagen, video o la integración de estos. Las infocintas también pueden estar compuestas por estos tres componentes y pueden ser mostradas según el orden que ocupen en la lista de infocintas, en el caso de que sean

inmediatas, se muestran en el mismo momento en que son creadas y mandadas a transmitir. El subsistema también permite gestionar canales externos, los cuales sirven para enlazar la transmisión de la plataforma con estos canales en el momento deseado.

2.3 Especificación de los requisitos de software

Durante el proceso de captura de requisitos el propósito fundamental del equipo de desarrollo es guiar el desarrollo hacia el sistema que desea y necesita el cliente. Este proceso puede resultar complejo en productos adaptables como PRIMICIA, donde la captura de requisitos se realiza con los miembros del proyecto y el entorno de trabajo está en dependencia del cliente que se interese.

Como parte de esta investigación se decidió realizar las siguientes tareas para obtener y refinar los requisitos funcionales y no funcionales:

-Obtener información sobre el dominio del problema y el sistema actual.

-Identificar los requisitos funcionales y no funcionales, los actores y casos de uso del sistema a desarrollar.

-Priorizar los requisitos y casos de uso.

La plataforma constituye un producto adaptable que puede ser personalizado a cualquier interesado, por esta razón la captura de requisitos fue realizada a partir de las necesidades generales planteadas por los miembros del proyecto y con las cuales deberá cumplir el subsistema. Las tareas antes mencionadas se realizaron a través de talleres de proyecto. Como principal resultado se obtuvo una versión del listado de requisitos que luego fue refinada en otros talleres del proyecto. De esta forma se verificó que no hubiera omisión de ninguna funcionalidad y que los requisitos fueran consistentes.

2.3.1 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir (23). Se mantienen invariables sin importar con que propiedades o cualidades se relacionen.

RF 1 Transmitir señal interna**:** Consiste en la función principal del sistema. Esta transmisión interna está compuesta principalmente por noticias.

RF 1.1 Mostrar video de presentación: Al iniciar la transmisión se debe mostrar el video de presentación del canal.

RF 1.2 Mostrar cartelera: El sistema debe visualizar una cartelera como inicio del ciclo de transmisión, mostrando para cada noticia la sección y el titular en el orden en que serán mostradas posteriormente.

RF 1.3 Mostrar noticia: Se visualizarán noticias obteniendo su configuración de un fichero XML[4] que haya sido generado en el subsistema de administración. Cada noticia tiene asociada un fondo musical.

RF 1.4 Mostrar infocinta: El sistema será capaz de mostrar cintillos informativos inmediatos o programados con anterioridad obteniendo su configuración de un fichero XML que haya sido generado en el subsistema de administración.

RF 1.5 Cargar componentes adicionales: El sistema debe permitir cargar componentes visuales que muestren informaciones adicionales tales como: pronósticos meteorológicos, resultados deportivos, resúmenes culturales, precios de productos u otro tipo de informaciones adquiridas.

RF 2 Transmitir señal externa**:** Se detendrá la transmisión de la señal del canal y se comenzará a transmitir la señal proveniente de la fuente externa, ya sea para la transmisión directa de una información inmediata o simplemente para completar la programación del canal cuando no existan noticias.

RF 3 Mostrar patrón del canal: Se mostrará el patrón del canal cuando no se esté transmitiendo ningún material o cuando se encuentre fuera de servicio.

RF 4 Mostrar videos de cambio de señal: Se transmitirán videos de poca duración en los cambios de señal.

RF 5 Cargar configuración**:** El sistema debe permitir cargar las configuraciones visuales del canal almacenadas en un fichero XML y aplicar las mismas.

[4] Lenguaje de Marcas Extensible del inglés Extensible Markup Language

RF 6 Ajustar elementos visuales**:** Los elementos serán posicionados y redimensionados teniendo en cuenta la resolución con la que se va a transmitir la señal del canal, por lo que serán ubicados de forma relativa a dicha resolución.

RF 7 Comunicar subsistemas**:** El sistema debe permitir establecer comunicación con el subsistema de administración.

2.3.2 Requisitos no funcionales

Los requisitos no funcionales especifican propiedades o cualidades del sistema (23). Son características que hacen al producto, atractivo, usable, rápido y confiable.

RNF 1 Requisitos de software: El sistema operativo que soportará la solución será Ubuntu/Linux 11.04 o superior. Se necesitará un servidor NFS[5] para proveer los archivos multimedia.

RNF 2 Requisitos de hardware: Requerimientos de hardware del Servidor de Transmisión:

-Intel Pentium Dual Core G630 2.7 GHz.

-1 GB de RAM[6].

-Tarjeta de red.

-Tarjeta exportadora de video.

RNF 3 Restricciones en el diseño y la implementación:

-Se debe utilizar el lenguaje UML y Visual Paradigm como herramienta de modelado.

-El subsistema estará implementado en lenguaje C++.

-La arquitectura debe soportar migrar la interfaz de forma rápida, para lograr visualizar cualquiera de los cambios que se produzcan.

[5] Sistema de archivos de red del inglés Network File System
[6] Memoria de acceso aleatorio

RNF 4 Requisitos de Usabilidad: Se debe mostrar la información de forma lógica y correctamente estructurada.

RNF 5 Interfaces de Comunicación: Se utilizará la red interna de televisión para la transmisión de la señal del canal.

RNF 6 Interfaces de Software: Se reutilizan las funcionalidades brindadas por el Qt Multimedia Kit para la transmisión de los videos que se visualizarán en el canal y la reproducción de música.

2.4 Descripción del sistema propuesto

2.4.1 Actores propuestos para el sistema

A continuación se presenta una breve descripción que incluye las responsabilidades que poseen los actores del sistema.

Tabla 1: Actores del sistema.

Usuario	Encargado de establecer las propiedades de los elementos visuales y de iniciar la aplicación para lograr la visualización de los contenidos en el canal.
Subsistema Administración	Encargado de mandar un cambio de señal que provocará que se cambie el canal que se está transmitiendo.

2.4.2 Diagrama de Casos de Uso del Sistema

En la figura 2 se muestra la relación de los actores con los casos de uso del sistema (CUS) identificados durante la realización del modelo del sistema.

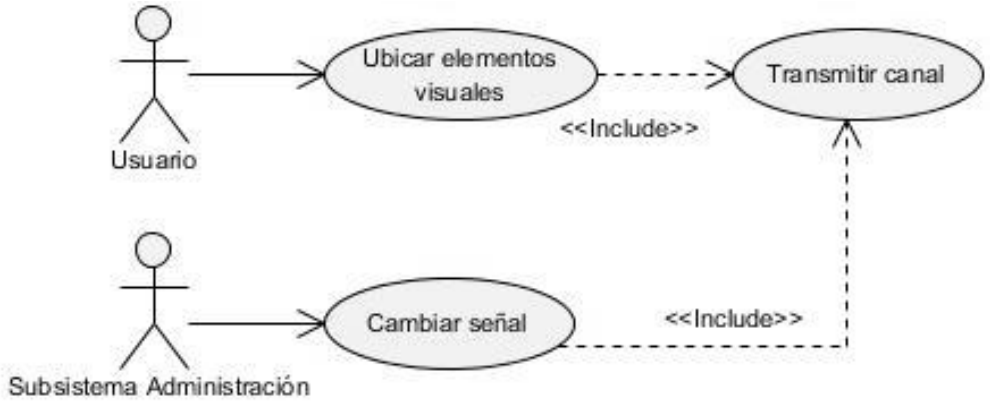

Figura 2: Diagrama de CU del subsistema de visualización de noticias para PRIMICIA v2.0

2.4.3 Especificación de los Casos de Uso

Tabla 2: CUS-Ubicar elementos visuales.

Objetivo	Ubicar los elementos visuales, tales como: título de la noticia y tamaño de la pantalla.
Actores	Usuario: (Inicia) Ubicar elementos visuales.
Resumen	El caso de uso se inicializa cuando el usuario ejecuta la aplicación. Se configuran los elementos visuales y se ajustan las propiedades de la ventana de la aplicación.
Complejidad	Baja.
Prioridad	Secundario.
Precondiciones	-
Postcondiciones	Elementos visuales ajustados.
Flujo de eventos	
Flujo básico Ajustar configuración	

	Actor	Sistema
1.	Ejecuta la aplicación.	
2.		Muestra una ventana para configurar los elementos visuales de la aplicación con los valores que se utilizaron la última vez que se ejecutó.
3.	Establece las propiedades de los elementos visuales tales como: título de la noticia y tamaño de la pantalla. Selecciona la opción aceptar.	
4.		Guarda la configuración en un fichero XML.
5.		En dependencia de los valores establecidos se ajustan las propiedades de la ventana de la aplicación.
6.		Comienza la transmisión del canal, ir a **CUS Transmitir canal.**
7.		Termina el CU.

Relaciones	**CU Incluidos**	Transmitir canal. Ver CU Transmitir canal.
	CU Extendidos	No existen.

Tabla 3: CUS-Transmitir canal.

Objetivo	Mostrar el canal que corresponde transmitir en todo momento.
Actores	Usuario: (Inicia) Transmitir canal.

Resumen	El caso de uso se inicializa cuando el usuario ajusta los elementos visuales. Se muestra el video de presentación del canal. Si la señal activa es la interna, se muestra la cartelera, las noticias e infocintas creadas en el subsistema de administración y las informaciones adicionales. Si la señal es externa se muestra un video proveniente de una fuente externa.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	Los elementos visuales han sido ajustados.
Postcondiciones	Canal transmitido.

Flujo de eventos

Flujo básico Transmitir canal

	Actor	**Sistema**
1.		Muestra el video de presentación del canal.
2.		Se lee de la base de datos la escaleta de transmisión que contiene la programación de lo que va a mostrar el canal.
3.		Se muestran las señales interna o externa según la escaleta. Si la señal que toca transmitir es: a) La señal interna del canal, ver sección 1: Señal interna. b) Una señal externa, ver sección 2: Señal externa.

4.		Termina el CU.
Sección 1:" Señal interna"		
Flujo básico Transmitir señal interna		
	Actor	**Sistema**
1.		Lee en la base de datos los bloques de noticias activos (verifica que existe algún bloque de noticias con fecha de transmisión cuyo período de transmisión incluya la fecha actual).
2.		Muestra la cartelera con los títulos de las noticias correspondientes a los bloques activos. En la cartelera se muestra el orden de aparición de la noticia, la sección y el titular. Se reproduce un fondo musical que acompaña a la cartelera.
3.		Visualiza cada una de las pantallas de las noticias del ciclo noticioso y las infocintas normales. Muestra la fecha en formato abreviado, hora, temática a la que pertenece y el título de la noticia. Durante la visualización de las noticias se reproduce una música de fondo, excepto en las pantalla de tipo video. Muestra informaciones adicionales a la noticia, tales como: pronósticos meteorológicos, resultados deportivos,

		resúmenes culturales, precios de productos u otro tipo de informaciones adquiridas.
Flujos alternos		
1a. No existe ningún bloque de noticias con fecha de transmisión cuyo período de transmisión incluya la fecha actual.		
	Actor	**Sistema**
1.		Muestra el patrón del canal.
3a. **Evento:** Detecta infocinta inmediata con intervalo de publicación que coincide con la fecha y hora actual.		
	Actor	**Sistema**
3.		Visualiza la infocinta inmediata. Regresa al evento 3 del flujo básico Transmitir señal interna.
Sección 2:" Señal externa"		
Flujo básico Transmitir señal externa		
	Actor	**Sistema**
1.		Muestra un componente de tipo video que ocupa toda la ventana de la aplicación reproduciendo la fuente de la señal.

Relaciones	**CU Incluidos**	No existen.
	CU Extendidos	No existen.

Tabla 4: CUS-Cambiar señal.

Objetivo	Cambiar la señal que se está transmitiendo.

Actores	Subsistema Administración: (Inicia) Cambiar señal.	
Resumen	El caso de uso se inicializa cuando el subsistema de administración manda un cambio de señal.	
Complejidad	Media.	
Prioridad	Crítico.	
Precondiciones	Canal en transmisión.	
Postcondiciones	Cambiada la señal que estaba en transmisión.	
Flujo de eventos		
Flujo básico Cambio de señal		
	Actor	**Sistema**
1.	Manda un cambio de señal.	
2.		Verifica que la señal que se manda a cambiar no sea la misma que se está transmitiendo.
3.		Muestra video de cambio de señal.
4.		a) Si la señal que manda a cambiar es la del canal interno: -Cambio de señal, ir a la **sección 1 "Señal interna" del CUS Transmitir canal.** b) Sino: -Cambio de señal, ir a la **sección 2 "Señal externa" del CUS Transmitir canal.**

5.		Termina el CU.
Flujo alterno		
2a. La señal que se mandó a cambiar es la misma que se está transmitiendo.		
	Actor	**Sistema**
3a.		Termina el CU.

Relaciones	**CU Incluidos**	Transmitir canal. Ver CU Transmitir canal.
	CU Extendidos	No existen.

2.5 Conclusiones parciales

PRIMICIA v2.0 constituye un producto personalizable, por lo que no existe en el mismo un proceso de negocio visible y bien definido. Sin embargo, con la realización del modelo de dominio, se logró comprender mejor el funcionamiento del sistema, pues se identificaron y detallaron los principales eventos, términos y conceptos presentes en el entorno donde éste trabajará.

Quedaron plasmadas las prestaciones y características del subsistema de visualización de noticias a través del proceso de captura de requisitos. En este proceso se determinó que la transmisión de la señal interna del canal es la función principal del sistema, además dio la posibilidad de identificar los casos de uso del sistema como elemento clave en el desarrollo de software apoyado en la metodología RUP.

Como resultado principal de este capítulo quedó plasmada la propuesta de solución para el problema actual del sistema, detallada a través de los actores, casos de usos y sus diagramas, así como las especificaciones de los casos de uso.

CAPÍTULO 3: ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DEL SUBSISTEMA DE VISUALIZACIÓN DE NOTICIAS DE LA PLATAFORMA DE TELEVISIÓN INFORMATIVA, PRIMICIA

3.1 Introducción

En este capítulo se presenta la solución propuesta mediante el diagrama de clases del diseño el cual detalla la estructura que tendrá el subsistema de visualización de noticias. Se describe la implementación del sistema en términos de componentes, describiendo cómo los elementos del modelo del diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue.

3.2 Modelo de análisis

El modelo de análisis constituye una representación abstracta del sistema que se está modelando, suficiente para capturar la lógica esencial del sistema sin entrar en muchos detalles. En él se analizan los requisitos, refinándolos y estructurándolos, utilizando el lenguaje de los desarrolladores para describir cómo debería diseñarse e implementarse el sistema (23). Sin embargo este paso no es obligatorio, la metodología es altamente configurable y hay artefactos que pueden obviarse durante el desarrollo.

El modelo del análisis es opcional, en él, se describen un conjunto de clases del análisis que se utilizan para realizar una descripción abstracta de la realización de los casos de uso del modelo de casos de uso. Las clases del análisis luego evolucionan hacia otras clases más detalladas en el modelo del diseño (24).

Por lo anterior planteado se decide hacer una transición del modelo de casos de uso al modelo de diseño sin necesidad de realizar el modelo de análisis. Teniendo en cuenta que los requisitos son conocidos, que se tiene conocimiento sobre los procesos que se desarrollan y que el lenguaje de programación, el framework y en general las tecnologías sobre las que se estará desarrollando el sistema son elementos dominados.

3.3 Arquitectura de software

La arquitectura de software es la estructura del sistema, la cual abarca componentes de software, propiedades externas visibles de estos componentes y sus relaciones (25). La definición de la arquitectura de software constituye una paso fundamental a tener en cuenta y a desarrollar de la mejor forma posible

durante el proceso de desarrollo del producto. Es la encargada de que se produzca una correcta comunicación entre todas las partes involucradas y de proporcionar una visión global del sistema a construir.

3.3.1 Patrones de arquitectura: Arquitectura en capas

La arquitectura en capas soporta un diseño basado en niveles de abstracción crecientes, lo cual a su vez permite la partición de un problema complejo en una secuencia de pasos incrementales. Brinda una organización jerárquica tal que cada capa proporciona servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. Además admite optimizaciones, refinamientos y proporciona una amplia reutilización (26).

Para el desarrollo del subsistema se definió la arquitectura en tres capas, denominadas como: presentación, lógica de negocio y acceso a datos. La capa presentación es la encargada de contener la interfaz del sistema y tiene el objetivo de mostrar a los usuarios las noticias, infocintas y canales externos. La capa lógica de negocio es la que se encarga de extraer los datos de la capa inferior necesarios para realizar el procesamiento que permitirá visualizar la información en la pantalla. La capa acceso a datos es la encargada de adquirir de la base de datos la información guardada por el subsistema de administración y que posteriormente será utilizada por la capa lógica de negocio.

Se escogió este tipo de arquitectura pues con la misma se reducen las dependencias existentes entre las clases de capas distintas, de forma tal que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. Brindando la posibilidad de mejorar el soporte del sistema y de poseer una mejor organización durante el desarrollo del software, además cada nivel tiene funcionalidades diferentes lo que permite el diseño de una arquitectura escalable que puede ser ampliada en caso de ser necesario.

3.4 Modelo de diseño

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en como los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Es utilizado como entrada fundamental a las actividades de implementación.

3.4.1 Patrones de diseño

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares (27). Durante el desarrollo del presente trabajo de diploma fueron utilizados los siguientes patrones de diseño:

Patrones generales de software para asignar responsabilidades (GRASP[7]) (22):

Experto: Este patrón asigna la responsabilidad al experto en la información, o sea a la clase que tiene los atributos involucrados. Está presente en todas las clases del sistema, cada responsabilidad está asignada a la clase que cuenta con la información necesaria para realizarla.

Creador: Guía la asignación de responsabilidades relacionadas con la creación de objetos. La intención básica del patrón es encontrar un creador que necesite conectarse al objeto creado en alguna situación. La clase Lector_XML es la encargada de crear los componentes multimedia que componen una noticia o una infocinta según el fichero XML correspondiente.

Controlador: Está diseñado para asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Este patrón se pone de manifiesto en la clase Controladora, encargada de llevar la lógica general de la aplicación.

Alta cohesión: Propone asignar la responsabilidad de manera que la complejidad se mantenga dentro de límites manejables asumiendo solamente las responsabilidades que deben manejar, evadiendo un trabajo excesivo. Este permitió una mayor capacidad de reutilización y claridad en el entendimiento del sistema. Esto se evidencia en la clase Controladora, la cual realiza las tareas referentes a la lógica de la aplicación, apoyándose en clases que complementan su trabajo.

Bajo Acoplamiento: El bajo acoplamiento soporta el diseño de clases más independientes, que reducen el impacto de los cambios, que son más reutilizables y que acrecientan la oportunidad de una mayor productividad. Este patrón se evidencia en la solución propuesta al existir poca dependencia entre las clases.

[7] General Responsibility Assignment Software Patterns.

Patrones GOF[8] (22):

Observador: Permite definir una dependencia uno a muchos entre objetos, de forma tal que cuando el objeto cambie de estado, todos sus objetos dependientes sean notificados y actualizados automáticamente. Este patrón se evidencia en la solución propuesta cuando se conecta la clase Controladora con varios temporizadores y con la clase Principal.

3.4.2 Diagrama de clases del diseño

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases y de las interfaces en una aplicación. En él se representan las clases y las relaciones existentes entre estas, así como los atributos y métodos. A continuación se muestra el diagrama de clases del diseño del subsistema de visualización de noticias.

[8] Gang of Four.

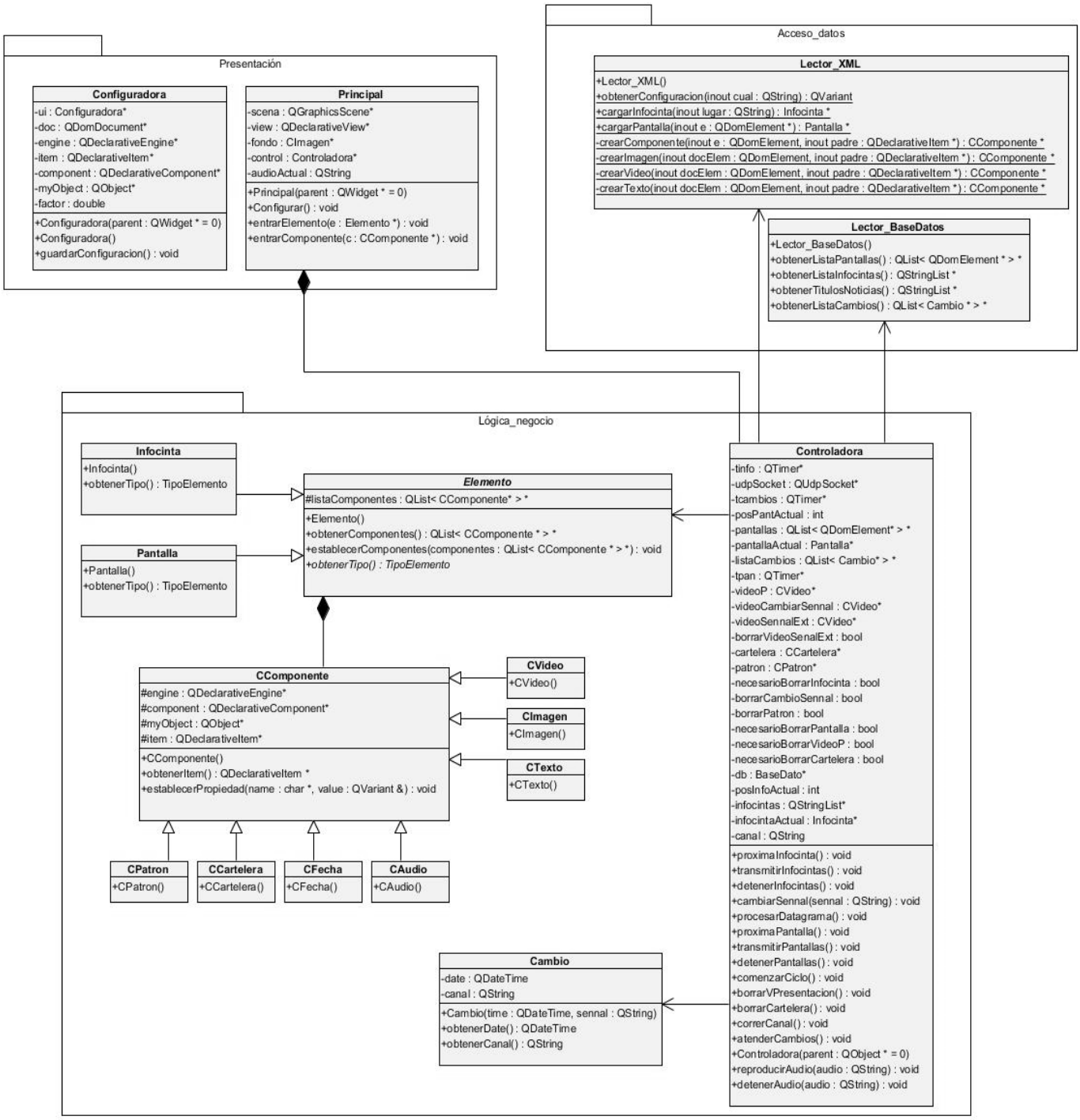

Figura 3: Diagrama de clases del diseño del subsistema de visualización de noticias.

La clase **Configuradora** le brinda al usuario la posibilidad de ajustar los elementos visuales tales como: el título de la noticia y las dimensiones de la ventana de la aplicación. La clase **Principal** es la encargada de mostrar las noticias, infocintas, informaciones adicionales y canales externos, según le ordene la clase **Controladora.** Esta contiene la lógica de la aplicación, creando los elementos visuales, obtenidos a través de la clase **Lector_BaseDatos** e interpretados con la clase **Lector_XML**. La clase **Elemento** contiene una lista de objetos de la clase **CComponente** que pueden ser de tipo imagen, texto y video. Existen otros componentes visuales que complementan la información mostrada por PRIMICIA como son: audio, cartelera, patrón y fecha. La clase **Cambio** contiene la información de cuándo es necesario un cambio de señal.

3.5 Modelo de despliegue

El modelo de despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Cada nodo representa un recurso de cómputo, normalmente un procesador o un dispositivo hardware similar y poseen relaciones que representan medios de comunicación entre ellos, la funcionalidad de un nodo se define por los componentes que se distribuyen en ese nodo (23).

Los nodos son los elementos donde se ejecutan los componentes, representan el despliegue físico de los componentes. Los componentes son los elementos que participan en la ejecución del sistema, representan el empaquetamiento físico de los elementos lógicos (29). A continuación se presenta el diagrama de despliegue que muestra el entorno donde va a estar la la aplicación que se desarrolla.

Figura 4: Diagrama de despliegue.

3.6 Modelo de implementación

El modelo de implementación describe cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y en el lenguaje de programación utilizado. También describe cómo dependen los componentes unos de otros. Identifica los componentes físicos de la implementación para que puedan comprenderse y gestionarse mejor (23).

3.6.1 Diagrama de componentes

Un componente es una parte del sistema sustituible, casi independiente e importante que desempeña una función clara en el contexto de una arquitectura bien definida (30). Algunos estereotipos de componentes son los siguientes:

-<<executable>> programa que puede ser ejecutado en un nodo.

-<<file>> fichero que contiene código fuente o datos.

-<<library>> librería estática o dinámica.

-<<table>> tabla de base de datos.

-<<document>> documento.

El diagrama de componentes muestra la relación entre los componentes, sus dependencias, comunicaciones y localización, los cuales son usados para estructurar los componentes en los sistemas del software. A continuación se muestra dicho diagrama.

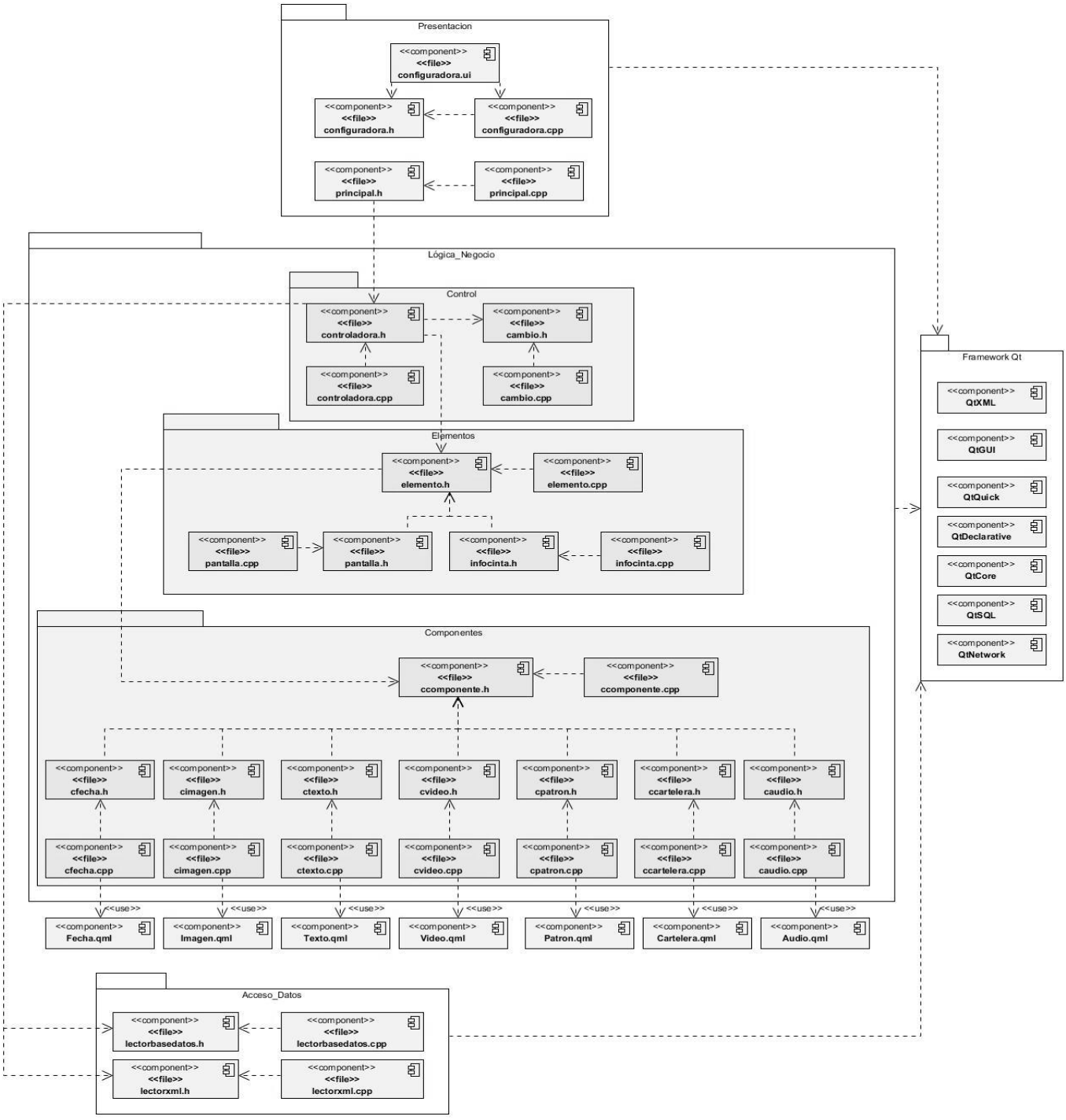

Figura 5: Diagrama de componentes del subsistema de visualización de noticias.

3.7 Estándar de codificación

Un estándar de codificación es un conjunto de reglas de notación y nomenclatura, específicas de cada lenguaje de programación, que se usan y se siguen durante la fase de implementación (codificación) de una aplicación y reducen perceptiblemente el riesgo de que los desarrolladores introduzcan errores que no son detectados por los compiladores, reduciendo el tiempo y coste de las actividades de depuración y pruebas necesarias para la detección y corrección de los mismos (31).

La utilización de un estándar de codificación constituye una buena práctica de programación que reduce considerablemente la posibilidad de cometer errores, se obtiene como producto final una aplicación con código legible y comprensible por otras personas del equipo de desarrollo. Para la elaboración del estándar de codificación utilizado se tuvo en cuenta las características del framework de desarrollo Qt, el entorno integrado de desarrollo Qt Creator y el lenguaje de programación C++. A continuación se muestran algunas de las características con las cuales cumple la implementación del subsistema de visualización de noticias:

1. Los nombres de cada uno de los elementos del programa son significativos.
2. No se maneja en el programa más de una instrucción por línea.
3. Las variables se declaran en líneas separadas.
4. Los atributos comienzan con letra minúsculas.
5. Los nombres de los métodos comienzan con letra minúscula, pero si el nombre del método está compuesto por más de una palabra, cada una, excepto la primera empieza con mayúscula. Ejemplo: transmitirInfocintas().
6. Para nombrar las clases, interfaces y archivos se utiliza un sustantivo singular, con la primera letra en mayúscula y las demás en minúsculas.
7. Para separar palabras se usa la diferenciación entre mayúsculas y minúsculas.
8. Para los atributos que representan listas, el nombre de este está constituido en su primera parte por la palabra lista seguida por un nombre sugerente. Ejemplo: listaComponentes.

9. Las funciones implementadas llevan comentarios describiendo el objetivo de la función, si se considera necesario, pero no la descripción del funcionamiento.

3.8 Conclusiones parciales

En el diagrama de clases del diseño se representaron las clases y las relaciones existentes entre ellas de manera detallada. Estas clases fueron agrupadas teniendo en cuenta la arquitectura en tres capas, lo que permitió tener una correcta estructura de los elementos de software, simplificar la complejidad y tener una organización en capas. Esta representación sirvió de base para la implementación de las funcionalidades del subsistema de visualización de noticias.

Se redujo considerablemente la posibilidad de cometer errores durante la implementación del subsistema. Se obtuvo como producto final una aplicación con código legible y comprensible por otras personas del equipo de desarrollo, pues se hizo uso de un estándar de codificación. Se describieron los componentes a construir y su organización, así como la dependencia entre los nodos físicos en los que funcionará la aplicación.

CAPÍTULO 4: VALIDACIÓN Y PRUEBAS REALIZADAS AL SUBSISTEMA DE VISUALIZACIÓN DE NOTICIAS DE LA PLATAFORMA DE TELEVISIÓN INFORMATIVA, PRIMICIA

4.1 Introducción

Las pruebas realizadas a una aplicación informática constituyen un elemento fundamental para avalar la eficacia del software y representa una revisión de las especificaciones de la aplicación. Existen varias tipologías de pruebas, cada una de ellas con metas y estrategias bien definidas. En este capítulo se describen las pruebas que se le realizaron al subsistema con el objetivo de asegurarse de que el mismo cumple con los requisitos del cliente y se recogen los resultados obtenidos luego de realizar este proceso a la solución propuesta.

4.2 Prueba de software

Las pruebas de software son una actividad en la cual un sistema o componente, es ejecutado bajo unas condiciones o requerimientos especificados, los resultados son observados y registrados. El objetivo de estas es asegurar que el software cumpla con las especificaciones requeridas y eliminar los posibles defectos que este pudiera tener. Las pruebas de software son un elemento crítico para la garantía de la calidad del software (32).

4.2.1 Pruebas de aceptación

Se realizó este tipo de prueba con el objetivo de validar que el sistema cumplía con el funcionamiento esperado por el cliente y de brindarle la posibilidad a los directivos del proyecto para el cual fue desarrollado el subsistema, que determinaran si estaban conformes con la solución obtenida, desde el punto de vista de su funcionalidad y rendimiento.

Tabla 5: Personas que participan en las pruebas de aceptación.

Especialista	Cargo
Ing. Félix Iván Romero Rodríguez.	Jefe de Proyecto. Administrador de configuración.
Ing. Carlos de Jesús Andrés.	Jefe de Proyecto.

Ing. Yanary Hernández Sosa.	Arquitecto de Software.
Ing. Rafael Lorente Miranda.	Programador.

4.2.2 Pruebas de rendimiento

Las pruebas de rendimiento son realizadas para comprobar que tan rápido un sistema efectúa una tarea bajo ciertas circunstancias pre-planificadas de trabajo. Estas pruebas también son utilizadas para validar y verificar diferentes aspectos de la calidad de software.

Diseño de prueba de rendimiento

Para probar el rendimiento se ejecutó la aplicación y se realizó un monitoreo del consumo de memoria y de la Unidad Central de Procesamiento (CPU). El proceso se ejecutó en dos estaciones de trabajo con características diferentes, lo que permitió establecer límites en cuanto a la cantidad de animaciones a transmitir respondiendo a un hardware específico. A continuación se muestran las características de ambas estaciones, los niveles de consumo de memoria y de la CPU del subsistema durante su funcionamiento:

Estación de trabajo 1:

-Sistema Operativo:

Ubuntu 12.10

GNOME 3.0

-Hardware:

Procesador: Intel Core 2 Duo E4400 2.0 ghz

Memoria RAM: 1 GB

Tabla 6: Prueba de rendimiento para la estación de trabajo 1.

Animaciones	Porciento de uso del CPU	Memoria RAM consumida(MB)

2	60	40
6	84	58
10	96	76

La aplicación funciona sin ninguna dificultad pero con el incremento de las animaciones y los elementos multimedia (principalmente videos) mostrados en una pantalla simultáneamente, ocurre una caída en los FPS[9]. El uso de la memoria RAM promedia los 58 MB, aumentando o disminuyendo en dependencia de la cantidad de elementos multimedia mostrados en pantalla.

Estación de trabajo 2:

-Sistema Operativo:

Ubuntu 12.10

GNOME 3.0

-Hardware:

Procesador: Intel Atom D510 1.66 ghz x 4

Memoria RAM: 512 MB

Tabla 7: Prueba de rendimiento para la estación de trabajo 2.

Animaciones	**Porciento de uso del CPU**	**Memoria RAM consumida(MB)**
2	92	42
6	100	70

[9] Fotogramas por segundo

La aplicación funciona con marcada lentitud y con una baja tasa de FPS. Además al poner la aplicación al límite, al transmitir 6 videos simultáneamente la aplicación se cierra puesto que el sistema es incapaz de soportar la carga de procesamiento gráfico.

4.2.3 Pruebas de caja negra

Las pruebas de caja negra son las que se llevan a cabo sobre la interfaz de software. Pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto. Examina algunos aspectos del modelo fundamental del sistema sin tener mucho en cuenta la estructura lógica interna del software. Estas pruebas se centran en los requisitos funcionales del software (32).

Un caso de prueba basado en un caso de uso o un escenario específico, incluye la verificación del resultado de la interacción entre el actor y el sistema, comprueba que se satisfacen las precondiciones y postcondiciones definidas y que se sigue la secuencia de acciones descritas por el caso de uso.

Estas pruebas permiten encontrar (32):

-Funciones incorrectas o ausentes.

-Errores de interfaz.

-Errores en estructuras de datos o en accesos a las bases de datos externas.

-Errores de rendimiento.

-Errores de inicialización y terminación.

Diseño de caso de prueba

En la fase de prueba se realizaron dos iteraciones al sistema, en la primera iteración se encontró una no conformidad en el mismo; el requisito funcional 4: Mostrar videos de cambio de señal no estaba funcionando correctamente. Esta no conformidad se analizó y se le dio solución de manera inmediata. Posteriormente se realizó una segunda iteración de prueba para verificar que se había eliminado y para ver si se encontraban nuevos problemas en el funcionamiento del sistema. Esta última concluyó

satisfactoriamente, no se encontró ningún problema en el subsistema de visualización. A continuación se presentan los casos de prueba resultantes en la segunda iteración.

Tabla 8: Secciones a probar en el CU Ubicar elementos visuales

Nombre de la sección	Escenario	Acción realizada	Respuesta del sistema	Resultado de la prueba
SC 1: Ajustar configuración.	EC 1.1: Establecer las propiedades de los elementos visuales.	El usuario establece las propiedades de los elementos visuales tales como: título de la noticia y tamaño de la pantalla. Selecciona la opción aceptar.	El sistema guarda la configuración en un fichero XML, ajusta las propiedades de la ventana de la aplicación y manda a transmitir el canal.	Satisfactorio.

Tabla 9: Secciones a probar en el CU Cambiar señal.

Nombre de la sección	Escenario	Acción realizada	Respuesta del sistema	Resultado de la prueba
SC 1: Cambio de señal.	EC 1.1: Cambiar señal.	El subsistema de administración manda un cambio de señal.	El sistema muestra el video de cambio de señal y manda a transmitir dicha señal.	Satisfactorio

	EC 1.2: La señal que se mandó a cambiar coincide con la que se está transmitiendo.	El subsistema de administración manda un cambio de señal.	El sistema mantiene la transmisión de la señal actual.	Satisfactorio.

4.2.4 Prueba de caja blanca

Las pruebas de caja blanca del software comprueban los caminos lógicos proponiendo casos de prueba que ejerciten conjuntos específicos de condiciones y/o bucles.

El método del camino básico permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución. Los casos de prueba obtenidos del conjunto básico garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa (32).

Los pasos a seguir para la realización de las pruebas de caja blanca son los siguientes:

1. Generar el grafo de flujo de datos.

2. Calcular la complejidad ciclomática, V (G).

   V (G) = NA (Número de Aristas) - NN (Número de Nodos) + 2.

   V (G) = P (Nodos predicados) + 1.

   V (G) = Número de regiones.

La complejidad ciclomática es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa. Cuando se usa en el contexto del método de prueba del camino básico, el valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y da un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecuta cada sentencia al menos una vez (32).

3. Determinar los caminos independientes o básicos.

4. Generar un caso de prueba para cada camino de ejecución.

Diseño de prueba de caja blanca

Las pruebas de caja blanca permiten comprobar que se cumplan todos los caminos lógicos del programa. Con la aplicación de la técnica de camino básico se realiza un grafo de flujo del código fuente seleccionado previamente.

Método comenzarCiclo de la clase Controladora (ver Anexo 1)

Paso 1: Generar el grafo.

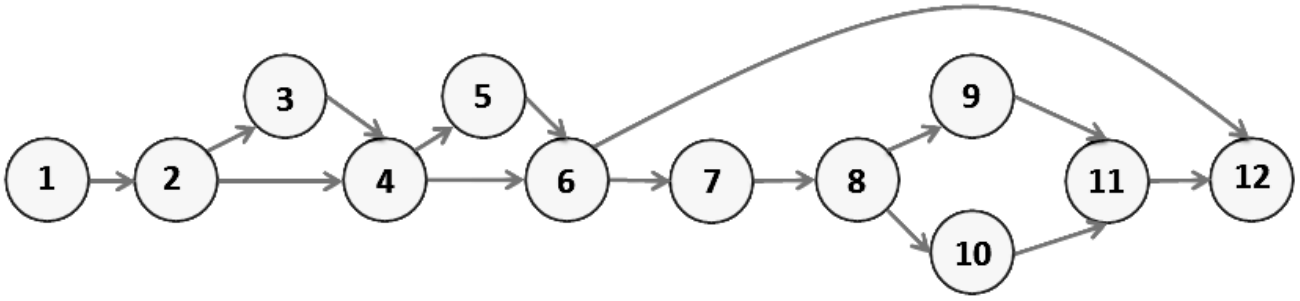

Figura 6: Grafo del flujo de datos de la función comenzar ciclo.

Paso 2: Calcular la complejidad.

V (G) = NA (Número de Aristas) - NN (Número de Nodos) + 2.

V (G) = 15-12+2=5

Paso 3: Determinar los caminos básicos.

CB 1: 1-2-3-4-6-7-8-9-11-12

CB 2: 1-2-4-6-7-8-9-11-12

CB 3: 1-2-4-5-6-7-8-9-11-12

CB 4: 1-2-4-6-12

CB 5: 1-2-4-6-7-8-10-11-12

Paso 4: Caso de prueba para el camino básico.

- *Caso de prueba para el camino básico 1*:

Entrada: borrarCambioSennal

Resultado esperado: liberar el espacio en memoria que ocupa el objeto encargado de mostrar el video de cambio de señal.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 2*:

Entrada: LectorXML::obtenerConfiguracion("fullscreen").toString()=="si"

Resultado esperado: mostrar la aplicación en pantalla completa.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 3*:

Entrada: necesarioBorrarInfocinta

Resultado esperado: liberar el espacio en memoria que ocupa el objeto que muestra la infocinta en pantalla.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 4*:

Entrada: canal != ”interno”

Resultado esperado: salir del método si la señal no es interna.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 5*:

Entrada: LectorXML::obtenerConfiguracion("fullscreen").toString()=="no"

Resultado esperado: mostrar el video de presentación ajustado a la ventana de la aplicación.

Resultado de la prueba: Satisfactorio.

Método proximaInfocinta de la clase Controladora (ver Anexo 2)

Paso 1: Generar el grafo.

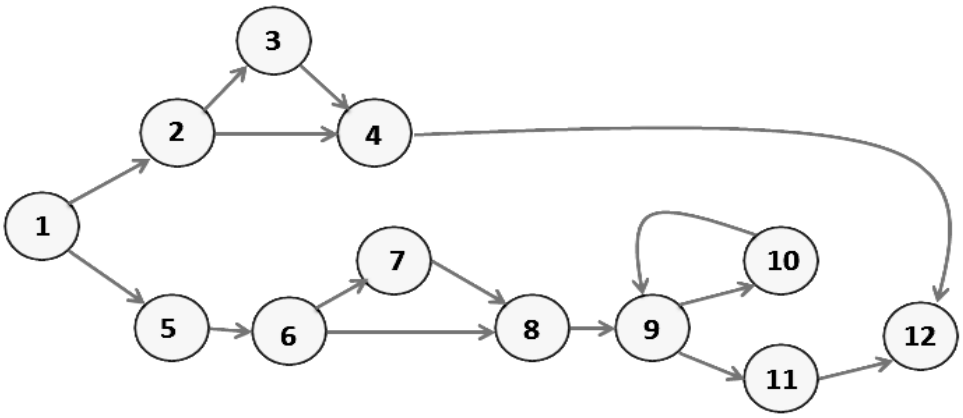

Figura 7: Grafo del flujo de datos de la función próxima infocinta.

Paso 2: Calcular la complejidad.

V (G) = NA (Número de Aristas) - NN (Número de Nodos) + 2.

V (G) = 15-12+2=5

Paso 3: Determinar los caminos básicos.

CB 1: 1-2-4-12

CB 2: 1-2-3-4-12

CB 3: 1-5-6-8-9-11-12

CB 4: 1-5-6-7-8-9-11-12

CB 5: 1-5-6-8-9-11-12

Paso 4: Caso de prueba para el camino básico.

- *Caso de prueba para el camino básico 1*:

Entrada: posInfoActual<infocintas->size()

Resultado esperado: mostrar la próxima infocinta de la lista.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 2*:

Entrada: necesarioBorrarInfocinta

Resultado esperado: si hay alguna infocinta cargada en memoria se elimina para poder poner la que corresponde.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 3*:

Entrada: posInfoActual>=infocintas->size()

Resultado esperado: borrar la lista de todas las infocintas.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 4*:

Entrada: a>0 && necesarioBorrarInfocinta

Resultado esperado: si hay alguna infocinta cargada en memoria se elimina para poder poner la que corresponde.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 5*:

Entrada: i<a

Resultado esperado: borrar todas las infocintas.

Resultado de la prueba: Satisfactorio.

Método proximaPantalla de la clase Controladora (ver Anexo 3)

Los pasos 1, 2 y 3 coinciden con el ejemplo anterior del método proximaInficinta() de la clase Controladora.

Paso 4: Caso de prueba para el camino básico.

- *Caso de prueba para el camino básico 1*:

Entrada: posPantActual<pantallas->size()

Resultado esperado: mostrar la pantalla que corresponde visualizar.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 2*:

Entrada: necesarioBorrarPantalla

Resultado esperado: si hay alguna pantalla cargada en memoria se elimina para liberar el espacio y poder poner la que corresponde.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 3*:

Entrada: posPantActual>=pantallas->size()

Resultado esperado: borrar la lista de todas las pantallas.

Resultado de la prueba: Satisfactorio.

- Caso de prueba para el camino básico 4:

Entrada: a>0 && necesarioBorrarPantalla

Resultado esperado: si hay alguna pantalla cargada en memoria se elimina para liberar el espacio y poder poner la que corresponde.

Resultado de la prueba: Satisfactorio.

- Caso de prueba para el camino básico 5:

Entrada: i<a

Resultado esperado: limpiar la lista de pantallas.

Resultado de la prueba: Satisfactorio.

Método cambiarSennal de la clase Controladora (ver Anexo 4)

Paso 1: Generar el grafo.

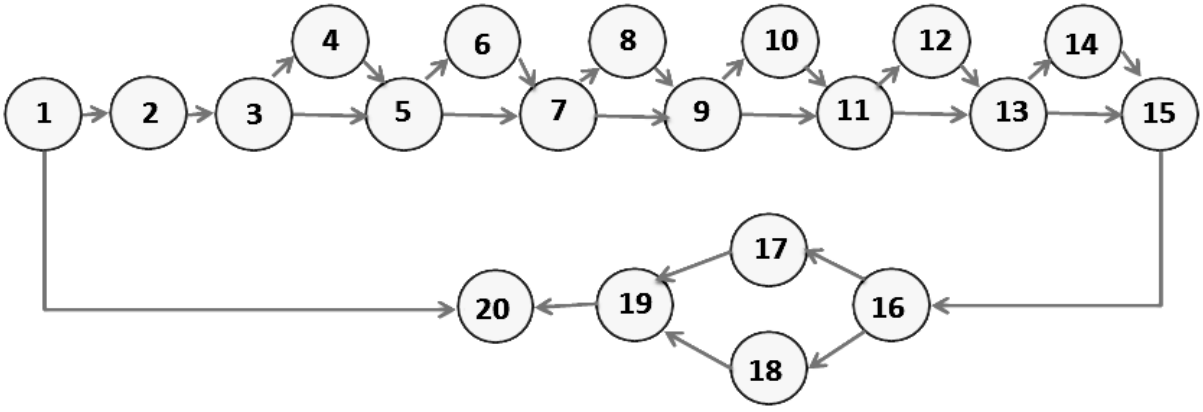

Figura 8: Grafo del flujo de datos de la función cambiar señal.

Paso 2: Calcular la complejidad.

V (G) = NA (Número de Aristas) - NN (Número de Nodos) + 2.

V (G) = 27-20+2=9

Paso 3: Determinar los caminos básicos.

CB 1: 1-2-3-5-7-9-11-13-15-16-17-19-20

CB 2: 1-20

CB 3: 1-2-3-4-5-7-9-11-13-15-16-17-19-20

CB 4: 1-2-3-5-6-7-9-11-13-15-16-17-19-20

CB 5: 1-2-3-5-7-8-9-11-13-15-16-17-19-20

CB 6: 1-2-3-5-7-8-9-10-11-13-15-16-17-19-20

CB 7: 1-2-3-5-7-9-11-12-13-15-16-17-19-20

CB 8: 1-2-3-5-7-9-11-12-13-14-15-16-17-19-20

CB 9: 1-2-3-5-7-9-11-12-13-14-15-16-18-19-20

Paso 4: Caso de prueba para el camino básico.

- *Caso de prueba para el camino básico 1*:

Entrada: LectorXML::obtenerConfiguracion("fullscreen").toString()=="si"

Resultado esperado: mostrar el video de presentación en pantalla completa.

Resultado de la prueba: Satisfactorio.

- *Caso de prueba para el camino básico 2*:

Entrada: canal==sennal

Resultado esperado: si la señal que se manda a cambiar es la misma que se está transmitiendo se sale del método.

Resultado de la prueba: Satisfactorio.

- Caso de prueba para el camino básico 3:

Entrada: necesarioBorrarCartelera

Resultado esperado: si la cartelera está cargada en memoria se elimina para liberar el espacio.

Resultado de la prueba: Satisfactorio.

- Caso de prueba para el camino básico 4:

Entrada: necesarioBorrarInfocinta

Resultado esperado: si hay alguna infocinta cargada en memoria se elimina para liberar el espacio.

Resultado de la prueba: Satisfactorio.

- Caso de prueba para el camino básico 5:

Entrada: necesarioBorrarPantalla

Resultado esperado: si hay alguna pantalla cargada en memoria se elimina para liberar el espacio.

Resultado de la prueba: Satisfactorio.

- Caso de prueba para el camino básico 6:

Entrada: necesarioBorrarVideoP

Resultado esperado: si el video de presentación está cargado en memoria se elimina para liberar el espacio.

Resultado de la prueba: Satisfactorio.

- Caso de prueba para el camino básico 7:

Entrada: borrarPatron

Resultado esperado: si el patrón del canal está cargado en memoria se elimina para liberar el espacio.

Resultado de la prueba: Satisfactorio.

- Caso de prueba para el camino básico 8:

Entrada: borrarCambioSennal

Resultado esperado: si el video de cambio de señal está cargado en memoria se elimina para liberar el espacio.

Resultado de la prueba: Satisfactorio.

- Caso de prueba para el camino básico 9:

Entrada: LectorXML:obtenerConfiguracion("fullscreen").toString()=="no"

Resultado esperado: mostrar el video de cambio de señal ajustado a la ventana de la aplicación.

Resultado de la prueba: Satisfactorio.

4.3 Conclusiones parciales

Durante la validación del subsistema mediante las pruebas de aceptación se pudieron ejecutar pruebas de software en un ambiente real y bajo la supervisión del equipo de desarrollo. En las pruebas de rendimiento se detectó que el desempeño gráfico correspondiente a la aplicación se sustenta sobre la capacidad operacional del microprocesador. Se necesita un microprocesador con buenas prestaciones puesto que PRIMICIA requiere de una capacidad elevada de procesamiento gráfico.

Con la realización de las pruebas de caja blanca se obtuvieron casos de prueba que garantizaron que se ejercitara por lo menos una vez todos los caminos independientes de las funcionalidades de la clase *Controladora* y que se ejercitaran todas las decisiones lógicas. Con la realización de las pruebas de caja negra y sus resultados satisfactorios se puede concluir que el software no presenta ningún error funcional.

CONCLUSIONES

A partir de la investigación realizada para dar cumplimiento al objetivo general planteado en el presente trabajo se generó una serie de artefactos lo cual representa una garantía para la continuidad del trabajo del proyecto. El uso de la metodología y de las herramientas seleccionadas, permitió realizar y documentar el proceso de diseño e implementación del subsistema de visualización de noticias; concluyendo que la utilización de las mismas fue la adecuada para lograr obtener un producto final fiable y de calidad avalado por las pruebas de aceptación, que cumple con todas las especificaciones planeadas por el cliente.

En PRIMICIA v2.0 el orden para mostrar las noticias no dependa de la sección temática a la que pertenece y se muestran noticias adicionales durante el proceso de transmisión del canal. Por todo lo expuesto anteriormente se puede afirmar que con el desarrollo del subsistema de visualización de noticias para la Plataforma de Televisión Informativa, PRIMICIA en su versión 2.0, se ha alcanzado satisfactoriamente el objetivo y se han cumplido todas las tareas propuestas para el presente trabajo de diploma.

RECOMENDACIONES

El objetivo general trazado en el presente trabajo de diploma fue alcanzado satisfactoriamente; no obstante, se recomienda:

–Realizar un estudio con el objetivo de buscar alternativas para mejorar el rendimiento visual de la aplicación tratando de eliminar las caídas de FPS que ocurren cuando se carga una infocinta o pantalla.

–Proponer la implantación de sistemas de este tipo en hospitales, universidades, hoteles, otras sedes empresariales y ministeriales.

BIBLIOGRAFÍA Y REFERENCIAS BIBLIOGRÁFICAS

1. **López, Eugenio García Calderón.** Televisión I. Madrid : Fundación Rogelio Segovia para el desarrollo de las telecomunicaciones, 1986, pág. 395.

2. Manuales. [En línea] [Citado el: 30 de septiembre de 2012.] http://www.manuales.com/manual-de/primeros-modelos-de-television.

3. Diccionario de la lengua española- Vigésima segunda edición. *Real Academia Española.* [En línea] 2001. [Citado el: 30 de septiembre de 2012.] http://www.rae.es/drae/.

4. Glosario.net. [En línea] [Citado el: 5 de octubre de 2012.] http://tecnologia.glosario.net/terminos-tecnicos-internet/transmisi%F3n-1631.html.

5. The Free Dictionary. [En línea] [Citado el: 5 de octubre de 2012.] http://www.thefreedictionary.com.

6. WordReference.com. [En línea] [Citado el: 5 de octubre de 2012.] http://www.wordreference.com/es/en/translation.asp?spen=transmisi%C3%B3n.

7. **Rodríguez, Raúl Lugo.** *Análisis, diseño e implementación del subsistema de Transmisión de la plataforma de televisión informativa PRIMICIA.* La Habana, Cuba : s.n., 2011.

8. FINGERTEXT (SISTEMA TELETEXTO). *Ingeniería y Desarrollo de Sistemas para los sectores Broadcast y Tecnologías de la Información.* [En línea] Anglatécnic S. L. [Citado el: 30 de octubre de 2012.] http://www.anglatecnic.com/es-6-Fingertext-%28Sistema-Teletexto%29.html.

9. **Olivares Tamayo, Jorge Daniel y Rey Almaguer, Bernardo.** *Desarrollo del Canal Informativo del Ministerio del Poder Popular para la Energía y Petróleo de Venezuela: Subsistema de Administración.* Ciudad de La Habana : s.n., 2008.

10. Scribd. [En línea] [Citado el: 15 de noviembre de 2012.] http://www.scribd.com/doc/2050925/metodologias-de-desarrollo-software.

11. EVA. *Entorno Virtual de Aprendizaje.* [En línea] 2010. http://eva.uci.cu/mod/resource/view.php?id=9303&subdir=/Metodologias/RUP.

12. **Gallón, Álvaro Rendón.** *Desarrollo de Sistemas Informáticos usando UML y RUP.* Popayán : s.n., agosto 2004.

13. **Schmuller, Joseph.** *Aprendiendo UML en 24 Horas.* México : Pearson Educación, 2009.

14. Sitio de descargas de software. [En línea] [Citado el: 1 de diciembre de 2012.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.

15. **Mora, Sergio Luján.** *C++ paso a paso.*

16. CODEBOX. [En línea] [Citado el: 4 de diciembre de 2012.] http://www.codebox.es/glosario.

17. Sitio web de la E.U. de Ingeniería Técnica de Oviedo. [En línea] [Citado el: 5 de diciembre de 2012.] http://petra.euitio.uniovi.es/~i1667065/HD/documentos/Entornos%20de%20Desarrollo%20Integrado.pdf.

18. Qt. [En línea] [Citado el: 7 de diciembre de 2012.] http://qt.digia.com/Product/).

19. PaaSOS-TipeSoft-Servicios empresariales en la nube. [En línea] [Citado el: 8 de diciembre de 2012.] http://tipesoft.com/introduccin-a-qml-i/.

20. PostgreSQL. [En línea] [Citado el: 10 de enero de 2013.] http://www.postgresql.org.es/sobre_postgresql.

21. **Gómez, Gloria Lucia Giraldo.** *Actores y sus roles.Modelo del dominio.* Escuela de Sistemas,Universidad Nacional de Colombia – Sede Medellín : s.n.

22. **Larman, Craig.** *UML Y PATRONES. Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* Segunda.

23. **Ivar Jacobson, Grady Booch y James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.* ISBN 84-7829-036-2.

24. **Cabrera, Lisandra Delgado.** *Diseño de las nuevas funcionalidades del Módulo de Redacción de la Plataforma de Televisión Informativa PRIMICIA.* Ciudad Habana : s.n., 29 de Junio de 2010.

25. **Torossi., Gustavo.** *El Proceso Unificado de Desarrollo de Software.* 2008.

26. Ingeniería de Software II(CI-4713). *Diseño de la Arquitectura. Lógica con Patrones.* [En línea] [Citado el: 1 de marzo de 2013.] http://ldc.usb.ve/~mgoncalves/IS3/Clase%201a%20IS-Arquitectura.pdf.

27. **Kiccillof, Carlos Reynoso-Nicolás.** Estilos y Patrones en la Estrategia de Arquitectura de Microfoft. [En línea] Marzo de 2004. [Citado el: 2 de marzo de 2013.] http://www.willydev.net/descargas/prev/Estiloypatron.pdf.

28. **Tedeschi, Nicolás.** msdn. [En línea] [Citado el: 4 de marzo de 2013.] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

29. **Xavier Ferré Grau, María Isabel Sánchez Segura.** Desarrollo Orientado a Objetos con UML. [En línea] [Citado el: 6 de marzo de 2013.] http://fermat.usach.cl/~msanchez/comprimido/OBJETOS.pdf.

30. **Sommerville, Ian.** *Ingeniería del Software.* Madrid : PEARSON EDUCATION S.A, 2004. 84-7829-074-5.

31. **Remedio, Yaquelin Y Morales Rodriguez y Adisneisy C Román.** *Propuesta de una guía para estandarizar la codificación en la Universidad.* Ciudad de la Habana : s.n., 2007.

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

ANEXOS

Anexo 1: Código de la función comenzar ciclo de la clase controladora.

```
  void Controladora::comenzarCiclo()
  {
1    tpan->stop();
1    tinfo->stop();
2    if(borrarCambioSennal)
     {
3       delete videoCambiarSennal;
3       videoCambiarSennal=0;
3       borrarCambioSennal=false;
     }
4    if(necesarioBorrarInfocinta)
     {
5       delete infocintaActual;
5       infocintaActual=0;
5       necesarioBorrarInfocinta=false;
5       tinfo->setInterval(100);
     }
6    if(canal!="interno")
12      return;
7    pantallas=db->obtenerListaPantallas();
7    posPantActual=0;
7    videoP =new CVideo();
8    if(LectorXML::obtenerConfiguracion("fullscreen").toString()=="si")
     {
9       videoP->establecerPropiedad("width",QApplication::desktop()->width());
9       videoP->establecerPropiedad("height",QApplication::desktop()->height());
     }
     else
     {
10      videoP->establecerPropiedad("width",LectorXML::obtenerConfiguracion("ancho").toInt());
10      videoP->establecerPropiedad("height",LectorXML::obtenerConfiguracion("alto").toInt());
     }
11   videoP->establecerPropiedad("x",0);
11   videoP->establecerPropiedad("y",0);
11   videoP->establecerPropiedad("z",600);
11   videoP->establecerPropiedad("path","../TestFiles/esc");
11   videoP->establecerPropiedad("opacity",1);
11   videoP->establecerPropiedad("stretch",true);
```

```
11  videoP->establecerPropiedad("duracion",25000);
11   videoP->establecerPropiedad("efectoEntrada","fade");
11   videoP->establecerPropiedad("velocidadEntrada",1000);
11   videoP->establecerPropiedad("efectoSalida","fade");
11   videoP->establecerPropiedad("velocidadSalida",500);
11   emit mostrarComponente(videoP);
11   necesarioBorrarVideoP=true;
11   QTimer::singleShot(26000,this,SLOT(borrarVPresentacion()));
  } 12
```

Anexo 2: Código de la función próxima infocinta de la clase controladora.

```
void Controladora::proximaInfocinta()
{
1  if(posInfoActual<infocintas->size())
   {
2     if(necesarioBorrarInfocinta)
      {
3        delete infocintaActual;
3        necesarioBorrarInfocinta =false;
      }
4     infocintaActual=LectorXML::cargarInfocinta(infocintas->at(posInfoActual));
4     qDebug()<<infocintas->at(posInfoActual);
4     necesarioBorrarInfocinta=true;
4     tinfo->setInterval(infocintaActual->obtenerComponentes()->at(0)->obtenerItem()
            ->property("duracion").toInt()+infocintaActual->obtenerComponentes()->at(0)
            ->obtenerItem()->property("velocidadSalida").toInt());
4     emit mostrarElemento(infocintaActual);
4     posInfoActual++;
   }
   else
   {
5     posInfoActual=0;
5     int a=infocintas->size();
6     if(a>0 && necesarioBorrarInfocinta)
      {
7        necesarioBorrarInfocinta=false;
7        delete infocintaActual;
7        infocintaActual=0;
      }
8 9 10 for(int i=0;i<a;i++)
      {
10       infocintas->removeFirst();
      }
11    delete infocintas;
11    infocintas=0;
11    infocintas=db->obtenerListaInfocintas();
11    tinfo->setInterval(100);
   }
} 12
```

Anexo 3: Código de la función próxima pantalla de la clase controladora.

```
void Controladora::proximaPantalla()
{
 1 if(posPantActual<pantallas->size())
   {
 2    if(necesarioBorrarPantalla)
      {
 3       delete pantallaActual;
 3       pantallaActual=0;
 3       necesarioBorrarPantalla=false;
      }
 4    pantallaActual=LectorXML::cargarPantalla(pantallas->at(posPantActual));
 4    necesarioBorrarPantalla=true;
 4    emit reproducirAudio(pantallaActual->obtenerAudio());
 4    tpan->setInterval(pantallaActual->obtenerComponentes()->at(0)->obtenerItem()
           ->property("duracion").toInt()+pantallaActual->obtenerComponentes()->at(0)
           ->obtenerItem()->property("velocidadSalida").toInt());
 4    qDebug()<<pantallas->at(posPantActual);
 4    emit mostrarElemento(pantallaActual);
 4    posPantActual++;
   }
   else
   {
 5    posPantActual=0;
 5    int a=pantallas->size();
 6    if(a>0 && necesarioBorrarPantalla)
      {
 7       necesarioBorrarPantalla=false;
 7       delete pantallaActual;
 7       pantallaActual=0;
      }
8 9 10 for(int i=0;i<a;i++)
      {
 10      pantallas->removeFirst();
      }
 11   delete pantallas;
 11   pantallas=0;
 11   tpan->setInterval(100);
 11   emit detenerAudio();

 11  comenzarCiclo();
   }
}12
```

Anexo 4: Código de la función cambiar señal de la clase controladora.

```
void Controladora::cambiarSennal(QString sennal)
{
1   if(canal==sennal)
20      return;
2   canal=sennal;
2   detenerInfocintas();
2   detenerPantallas();
2   detenerAudio();
3   if(necesarioBorrarCartelera)
    {
4       delete cartelera;
4       cartelera=0;
4       necesarioBorrarCartelera=false;
    }
5   if(necesarioBorrarInfocinta)
    {
6       delete infocintaActual;
6       infocintaActual=0;
6       necesarioBorrarInfocinta=false;
    }
7   if(necesarioBorrarPantalla)
    {
8       delete pantallaActual;
8       pantallaActual=0;
8       necesarioBorrarPantalla=false;
    }
9   if(necesarioBorrarVideoP)
    {
10      delete videoP;
10      necesarioBorrarVideoP=false;
10      videoP=0;
    }
11  if(borrarPatron)
    {
12      delete patron;
12      patron=0;
12      borrarPatron=false;
    }
```

```
13 if(borrarCambioSennal)
   {
14    delete videoCambiarSennal;
14    videoCambiarSennal=0;
14    borrarCambioSennal=false;
   }
15 videoCambiarSennal =new CVideo();
16 if(LectorXML::obtenerConfiguracion("fullscreen").toString()=="si")
   {
17    videoCambiarSennal->establecerPropiedad("width",QApplication::desktop()->width());
17    videoCambiarSennal->establecerPropiedad("height",QApplication::desktop()->height());
   }
   else
   {
18    videoCambiarSennal->establecerPropiedad("width",LectorXML::obtenerConfiguracion("ancho").toInt());
18    videoCambiarSennal->establecerPropiedad("height",LectorXML::obtenerConfiguracion("alto").toInt());
   }
19 videoCambiarSennal->establecerPropiedad("x",0);
19 videoCambiarSennal->establecerPropiedad("y",0);
19 videoCambiarSennal->establecerPropiedad("z",600);
19 videoCambiarSennal->establecerPropiedad("path","../TestFiles/cambio");
19 videoCambiarSennal->establecerPropiedad("opacity",1);
19 videoCambiarSennal->establecerPropiedad("stretch",true);
19 videoCambiarSennal->establecerPropiedad("efectoEntrada","fade");
19 videoCambiarSennal->establecerPropiedad("velocidadEntrada",1000);
19 videoCambiarSennal->establecerPropiedad("efectoSalida","fade");
19 videoCambiarSennal->establecerPropiedad("velocidadSalida",500);
19 emit mostrarComponente(videoCambiarSennal);
19 borrarCambioSennal=true;
19 QTimer::singleShot(5000,this,SLOT(correrCanal()));
 } 20
```

GLOSARIO

Sección Temática: agrupación de noticias según su contenido. Puede ser nacional, internacional, deportiva, entre otras.

Bloque de noticias: agrupación de noticias sin tener en cuenta su contenido, un bloque de noticias puede estar compuesto por noticias nacionales, internacionales, deportivas, entre otras, a la vez.

Escaleta: esqueleto o esquema de transmisión que funciona como un guión a seguir.

Red de datos: es un sistema que enlaza dos o más puntos (terminales) por un medio físico, el cual sirve para enviar o recibir un determinado flujo de información.

Fotogramas por segundo: se refiere a la cantidad de cuadros que posee la imagen por segundo.

XML: se trata de un metalenguaje (un lenguaje que se utiliza para decir algo sobre otro lenguaje) extensible de etiquetas. Es un fichero que almacena la configuración de todos los elementos que conforman la noticia y las infocintas, así como las configuraciones visuales del canal.