UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
Facultad 6

HERRAMIENTA PARA LA TRANSMISIÓN DE AUDIOVISUALES BASADA EN LISTAS DE DECISIÓN DE EDICIÓN PARA EL DEPARTAMENTO DE SEÑALES DIGITALES

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autores:
Yoralis Del Toro Fornaris
Daril Rosales Rosales

Tutor:
Ing. Solangel Rodríguez Vázquez

Co-tutor:
Ing. Vladimir Martell Fernández

Consultor:
Ing. Jean Michael Suárez Pérez

Ciudad de La Habana, junio de 2012
"Año 54 de la Revolución"

Cuando por los años no puedas correr, trota.
Cuando no puedas trotar, camina.
Cuando no puedas caminar, usa el bastón.
¡Pero nunca te detengas!

Teresa de Calcuta

DECLARACIÓN JURADA DE AUTORÍA

Declaramos por este medio que nosotros, Yoralis Del Toro Fornaris y Daril Rosales Rosales, con carné de identidad 88122549776 y 88111532986 respectivamente somos los autores de este trabajo y que autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste, firmamos la presente declaración jurada de autoría en La Habana a los18 días del mes de junio del año 2012.

Yoralis Del Toro Fornaris
Autor

Daril Rosales Rosales
Autor

Ing. Solangel Rodríguez Vázquez
Tutor

DEDICATORIA

De **Yory**:

A **Dios**, por ayudarme tanto en estos 5 años de sacrificio.
A mi abuelito **Perucho**, impulsor principal de este sueño.
A mi madre **Noralis**, por guiarme, aconsejarme y apoyarme en toda mi vida.
A mi abuela **Oneida,** por contribuir, junto a mi abuelo, en mi idea de ingeniera.
A **Pocholo**, **Belexis**, **Dary** y **Eduardito**, por las tantas alegrías para mi corazón.

De **Daril**:

El resultado de 5 años de sacrificio, de estudios, lejos de la familia, de estar ausente tanto en momentos difíciles como alegres, quiero dedicarlo especialmente a mi **madre** y a mi **padre**, a quienes debo el carácter y la actitud que me forjaron como un hombre de bien y perseverante, por constituir la fuerza impulsora que solidificó la base de mis ideales de convertirme en ingeniero en ciencias informáticas.

AGRADECIMIENTOS

De **Yory**:

A mi **mamá,** por ser la persona más importante de mi vida, por estar conmigo en los momentos difíciles, por darme ánimo cuando las cosas no iban bien y parecía que se derrumbaban; por su amor, su cariño y el apoyo en estos años, por la alegría de sentirme su hija.

A mi **papá**, por su amor, su cariño y por estar orgulloso de mí.

A mi novio **Vladimir**, por ser el centro de mi vida, por demostrarme día a día su amor y su cariño, por la idea de vivir toda una vida junto a mí; por hacerme sentir única.

A mi **abuela**, a mi tía **Tata** y a mi tío **Vladimir**, por siempre confiar en mí y formar parte de mi vida.

A mis **niños**, por alegrarme con sus travesuras y hacerme olvidar los momentos difíciles, por todo el amor que brota de sus corazoncitos.

A mi tía **Graciela**, a mi prima **Gracielita**, a mi prima **Rosy** y a mi prima **Lucía** por brindarme su cariño, su apoyo y su ayuda cuando lo necesité.

A mis amigas **Dunia**, **Maidy**, **Leyanis**, **Viana**, **Janet**, **Analay** y**Yelenis,** por su preocupación, por todos los momentos que hemos vivido juntas y hacer de cada uno una historia que contar.

A mis amigos **Yandys**, **Raidel**, **Yadian**, **Yasmani**, por los tantos momentos inolvidables de estos cinco años.

Gracias a todos...

De **Daríl**:

Especialmente a mi familia, a mi **mamá** y mi **papá**, que aunque no estuvieran presentes físicamente en momentos de tensión y preocupación siempre me dieron el apoyo necesario en mis decisiones, y me transmitieron esa energía positiva que sólo un padre es capaz de dar.

A mi **hermano**, quien me aconsejó cada instante y me hizo ver con su experiencia que una meta por muy difícil que sea de lograr si se persevera se alcanza.

A mi **novia** por estar siempre a mi lado, por ser fuerte y saber sobrellevar y aguantar situaciones tan difíciles a causa de mi estrés; a sus **padres**, a su **hermana**, a mi amiga **Brenda**, una bella persona que supo asustarme en momentos en que vacilaba y constituyó sin duda una gran ayuda para mi avance en este trabajo.

A todos **mis amigos**, los que no me dejaron solo y mostraron su interés sincero por verme culminar satisfactoriamente mi ocaso estudiantil.

RESUMEN

Entre los resultados de la aplicación de la Ciencia y las Tecnologías de la Información y las Comunicaciones en la sociedad se encuentra el nacimiento del audiovisual como forma superior de la comunicación humana al reflejar una experiencia casi directa de la realidad. El proceso de producción de estos audiovisuales está dividido en tres partes fundamentales siendo la última, la postproducción, la más importante al desarrollarse allí las actividades asociadas a la edición ensamblaje e inclusión de efectos decisivos en la calidad del material. En el proceso de edición y transmisión de un video se incurren en varias acciones las cuales van en detrimento de la eficacia del proceso: retrasos en la producción del material editado, utilización de espacios duplicados e imposibilidad de tramitar modificaciones de última hora. La investigación que se presenta pretende eliminar estas limitaciones a partir de la implementación de una herramienta que se utilice para la transmisión del audiovisual a través de ficheros EDL cargando hacia el proceso de producción parte de las actividades asociadas a la edición. Se presentan, como parte de la solución, un análisis del estado del arte del proceso de transmisión y de las soluciones existentes, el resultado del proceso de desarrollo de la herramienta expresados en los diagramas de dominio, casos de uso del sistema, del diseño y de implementación así como todas las descripciones textuales de los casos de uso. Finalmente se detalla la propuesta de pruebas al sistema a partir del diseño de los casos de prueba.

Palabras Clave: audiovisual, ficheros EDL, herramienta informática, transmisión.

INDICE DE TABLAS Y FIGURAS

FIGURAS

TABLAS

ÍNDICE DE CONTENIDOS

INTRODUCCIÓN

La ciencia y la tecnología han contribuido al mejoramiento notable de la forma de vida en la tierra, siendo las técnicas, procesos y máquinas que el hombre ha desarrollado a lo largo de la historia para apoyar y potenciar su capacidad de memoria, de pensamiento y de comunicación la base de lo que hoy se conoce como *Informática.*

Desde tiempos remotos el hombre necesitó expresar sus sentimientos, por lo que comenzó a dibujar en las paredes de sus cuevas, esto dio lugar a las primeras imágenes que hoy se conservan como elementos museables. Luego de un tiempo considerable, con estudios realizados, se logra la fotografía: el arte de escribir y pintar con luz.

La comunicación humana, desde sus inicios, fue una gran necesidad igualmente pues permitió a los individuos expresar y comprender sus ideas. Un factor importante que interviene en la comunicación es el oído humano, encargado de recibir ondas mecánicas (sonoras) denominadas *sonidos*. La conversión de estas ondas sonoras en señales analógicas es conocida como *señal de audio*.

Con el estudio ya adquirido, el hombre fue capaz de percibir que una sucesión de imágenes y sonidos a cierta velocidad y frecuencia creaba la ilusión de una escena en movimiento, esta percepción un tanto más rigurosa propició el nacimiento del audiovisual como forma superior de la comunicación (DEFINICIÓN.DE 2011a) al reflejar una experiencia casi directa de la realidad.

La mayoría de los autores e investigadores del área coinciden en que el proceso de producción de los materiales audiovisuales está dirigido y regido por tres etapas fundamentales: preproducción, producción y postproducción siendo esta última la más importante por desarrollarse las actividades asociadas a la edición, ensamblaje, musicalización e inclusión de efectos, que son decisivas en la calidad y la satisfacción del que recibe el audiovisual (SALESIANA 2008).

En el proceso de edición de un video (como actividad fundamental de la etapa de postproducción) (SALESIANA 2008) se incurren en varias acciones las cuales van en detrimento de la calidad y eficiencia del proceso:

1. Retrasos en la producción del material una vez editado. *Luego de lograda la edición del audiovisual, la obtención del producto está supeditado a un tiempo de fabricación que en múltiples ocasiones puede ser tedioso y costoso.*
2. Utilización de espacios duplicados. *Una vez concluido el tiempo de fabricación el material final ocupa el mismo espacio -o similar- que el material original, utilizando recursos del dispositivo de almacenamiento, lo anterior se agrava si se consideran producciones múltiples de un material original con cambios mínimos para cada una de sus versiones.*
3. Imposibilidad de tramitar modificaciones de última hora. *Debido a la necesidad obligatoria de emplear un tiempo considerable (en la mayoría de los casos semejante al tiempo de duración real del audiovisual) para la fabricación del producto final una vez editado, los cambios que se generen de última hora no se pueden gestionar pues una modificación por mínima que sea supone la fabricación completa del producto audiovisual.*

Con el objetivo de resolver algunas de las situaciones anteriores, en el Departamento de Señales Digitales del Centro de “Geoinformática y Señales Digitales” de la facultad 6 se trabaja en la actualidad en una solución de software para utilizar en el proceso de postproducción de audiovisuales y que sustituya a las clásicas herramientas de edición de videos.

La solución de referencia basa su funcionamiento en la llamada edición no lineal, la cual, al contrario de lo conocido tradicionalmente, no accede a los segmentos de audio y video a editar necesariamente de manera secuencial en el mismo orden en que fueron grabados; ni almacena la producción final en un espacio físico necesariamente.

Una variante muy actual del proceso de edición no lineal basa su funcionamiento en la construcción de ficheros con las órdenes a realizar sobre el material audiovisual original, moviendo parte del proceso de edición y ensamblaje para el propio proceso de transmisión, lo cual elimina costos por conceptos de tiempo, capacidad de almacenamiento y respuesta ante cambios. Los ficheros mencionados deberían ser procesados por una herramienta encargada de ejecutar las acciones descritas que permita transmitir el audiovisual.

Lo anterior elimina en alguna medida la compleja situación al disponer de una herramienta para lograr la edición, no obstante, el proceso de transmisión del audiovisual, según esta edición no lineal, utilizando los ficheros que produce la solución antes mencionada no está resuelto por dos razones fundamentales:

1. La solución desarrollada no prevé la transmisión en sí misma, solo la exportación del fichero.
2. La transmisión no se logra pues el formato definido para estos ficheros y las posibles acciones definidas en él deben ser entendidas por una herramienta que las ejecute.

Como consecuencia de la situación problemática anterior, se define el siguiente **problema a resolver** que da origen a la presente investigación:

¿Cómo contribuir al proceso de transmisión de audiovisuales garantizando reducción de esfuerzo por conceptos de tiempo, almacenamiento y gestión de cambios en el Departamento Señales Digitales?

Como resultado del análisis del propio problema se define el **objeto de estudio** como *el proceso de transmisión de audiovisuales*, enmarcándose la investigación propiamente en *la automatización de la transmisión de audiovisuales basado en listas de decisión de edición en el Departamento de Señales Digitales*, siendo este su **campo de acción**.

Finalmente, la investigación persigue el siguiente **objetivo general**:

Desarrollar una herramienta para la transmisión de audiovisuales basada en listas de decisión de edición para el Departamento de Señales Digitales.

Para lograr el objetivo propuesto se desarrollarán una serie de tareas a lo largo de la investigación las cuales se enumeran a continuación:

1. Caracterizar el proceso de transmisión de audiovisuales.
2. Describir estándares de reconocimiento y transmisión audiovisuales.
3. Caracterizar las listas de decisión de edición.
4. Valorar posibles soluciones que de alguna manera ofrezcan respuesta al problema planteado.
5. Determinar las tecnologías y herramientas a utilizar para el desarrollo de la herramienta para la transmisión de audiovisuales basada en listas de decisión de edición.
6. Realizar el análisis y diseño de la herramienta para la transmisión de audiovisuales basada en listas de decisión de edición.
7. Implementar la herramienta para la transmisión de audiovisuales basada en listas de decisión de edición tomando como base el análisis y diseño desarrollado.
8. Aplicar pruebas al sistema implementado.

Una vez concluida la investigación se esperan los siguientes resultados:

- La documentación técnica del proceso ingenieril del desarrollo de la herramienta para la transmisión de audiovisuales basada en listas de decisión de edición.
- La herramienta para la transmisión de audiovisuales basada en listas de decisión de edición.

Los autores de la investigación defienden la siguiente idea:

El desarrollo de una herramienta para la transmisión de audiovisuales basada en listas de decisión de edición garantizará la reducción de esfuerzo por conceptos de tiempo, almacenamiento y gestión de cambios en el proceso de transmisión de audiovisuales.

A lo largo de cualquier proceso investigativo se utilizan una serie de métodos científicos los cuales contribuyen en gran medida a su desarrollo debido a que constituyen la forma de abordar la realidad, de estudiar la naturaleza, la sociedad y el pensamiento, con el propósito de descubrir su esencia y sus relaciones (ALFREDO 2011).

Para llevar a cabo la presente investigación se utilizan los siguientes métodos teóricos:

- **Histórico-Lógico:** El método histórico estudia la trayectoria real de los fenómenos y acontecimientos en el transcurso de su historia. El método lógico investiga las leyes generales del funcionamiento y desarrollo de los fenómenos (ÁLVAREZ 1997).

  Este método se utiliza para estudiar la evolución de los conceptos asociados a la producción audiovisual y permite la definición de términos propios. Se utiliza también para estudiar la evolución y las tendencias de las tecnologías y demás herramientas a utilizar.

- **Analítico-Sintético:** Permite la descomposición de un todo complejo en sus partes y cualidades. La síntesis, por su parte, establece la unión entre las partes, previamente analizadas y posibilita descubrir relaciones y características generales entre los elementos de la realidad (ÁLVAREZ 1997).

  Este método se utiliza para la evaluación de soluciones que respondan al problema y permite realizar una valoración crítica y detallada de cada una de ellas. Se utiliza, además, para seleccionar las herramientas y tecnologías a utilizar durante el desarrollo de la herramienta.

- **Modelación:** Permite desarrollar una reproducción simplificada de la realidad, que cumple una función heurística, ya que permite descubrir y estudiar nuevas relaciones y cualidades del objeto de estudio (ÁLVAREZ 1997).

  Este método se utiliza para realizar la modelación y la creación de los artefactos ingenieriles y permite la reproducción simplificada del proceso de desarrollo de software.

La investigación se divide en 4 capítulos:

Capítulo 1: En este capítulo se especifican y se explican los principales conceptos asociados a la transmisión de audiovisuales y las listas de decisión de edición. Se describe el proceso de transmisión de audiovisuales como objeto de estudio y se valoran y critican las soluciones actuales que de alguna manera ofrecen respuesta al problema en cuestión.

Capítulo 2: En este capítulo se definen, argumentan y valoran las principales herramientas, tecnologías, metodologías y/o lenguajes que se utilizan para la construcción de la solución.

Capítulo 3: En este capítulo se comienza la construcción de la solución según la metodología de desarrollo seleccionada, se definen los requisitos funcionales y no funcionales y se describen los casos de uso del sistema.

Capítulo 4: En este capítulo se concluye la construcción de la solución y se especifican los artefactos referidos a las etapas de diseño, implementación y pruebas. Se define el diagrama de despliegue e implementación. Finalmente, se diseñan y se aplican las pruebas del sistema.

CAPÍTULO 1
FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En el presente capítulo se describen los aspectos relacionados con la transmisión de datos específicamente la transmisión de audiovisuales, partiendo desde el mismo proceso de producción de los mismos, hasta la concepción de las listas de decisión de edición. Se documenta el proceso de transmisión streaming y los diferentes métodos o formas de realizarlo. Se realiza un análisis de sistemas similares que ofrecen respuesta a las problemáticas planteadas.

1.2. Conceptos asociados a la transmisión de audiovisuales

Producción de audiovisuales

Luego de una revisión del estado del arte, no se detecta un concepto unánime asociado a la producción de audiovisuales, no obstante, sí se conocen los conceptos de producción y audiovisual de forma individual. A continuación se proporcionarán algunas definiciones de cada uno de estos conceptos hasta llegar a una definición en conjunto.

Definición de Producción:

Según el sitio web www.definicion.de: producción proviene del latín productĭo, en términos generales y la palabra se refiere a la acción de producir (que según la Real Academia de la Lengua, en su quinta acepción, no es más que: fabricar, elaborar cosas útiles (RAE 2010), a la cosa ya producida (DEFINICIÓN.DE 2011b). Por otra parte el sitio www.wordreference.com define el vocablo producir como la acción de hacer, fabricar, trabajar, crear, elaborar, realizar (ANTÓNIMOS 2011).

Audiovisual:

Según el sitio web *www.definicion.de* audiovisual se refiere al formato de difusión de contenidos que se vale de imágenes ópticas acompañadas por grabaciones acústicas (DEFINICIÓN.DE 2011a).

Por su parte, el sitio web *www.wordreference.com*, en su segunda acepción, lo define como la proyección de imágenes de una película o diapositivas combinadas con sonidos, con fines didácticos

(DLE 2011). De lo anterior, los autores de la investigación concluyen que producción audiovisual se refiere a la acción de producir, fabricar o elaborar contenidos con imágenes y sonidos digitales capaces de visualizarse en una pantalla con un fin específico.

Etapas de la Producción de Audiovisuales

Algunos autores concuerdan en la idea de que la producción audiovisual está dividida en tres etapas principales: la preproducción, la producción y la postproducción (MENCHACA 2010); (ROMERA 2011); (MEDIACLUB 2010).

Etapa de Preproducción: Etapa previa a la realización, es la parte de la producción más importante. El material audiovisual se hace sobre el escritorio; se preparan presupuestos, cotizaciones, se realizan contactos tanto técnicos como de talento, se estudian tiempos y movimientos, se visitan las locaciones, se consideran comidas, viáticos y transporte (MENCHACA 2010).

Etapa de Producción: Es la realización del programa en sí, la grabación, ya sea en estudio o en exteriores, lo cual implica haber llevado a cabo toda la preproducción (MENCHACA 2010).

Etapa de Post-Producción: Es decisiva pues implica cómo va a quedar conformado finalmente el programa. Es la edición, la inclusión de efectos especiales, musicalización, titulaje, etc. Depende de esta etapa que el programa sea bueno o malo. Es posible tener una excelente preproducción y una buena producción, pero si la postproducción no es bien manejada, no tendrá ningún caso haber tenido éxito en las etapas anteriores (MENCHACA 2010).

Un concepto importante que se maneja en la producción de audiovisuales y que ha sido referido en los párrafos anteriores lo constituye el concepto de edición de audiovisuales. A continuación se realiza un análisis de las principales ideas que giran en torno a esta definición.

Edición de audiovisuales

La edición de audiovisuales es el proceso por el cual se organiza, se monta, se ordena y yuxtapone el video digital original (VEGA 2010). Por otra parte se refiere al proceso mediante el cual se elabora un trabajo a partir de las imágenes obtenidas de un soporte (archivo, cinta o disco óptico) de video, grabadas previamente. Para ello se necesita reproducir la fuente y realizar un troceado de la misma.

Una vez hecha la revisión de la fuente se seleccionan los fragmentos de video y audio que formarán parte del montaje (ECURED 2011a).

Existen dos tipos fundamentales de edición, la lineal y la no lineal (ECURED 2011a):

Edición lineal:

Se refiere al hecho de transferir segmento a segmento una o más cintas con el material original. En el proceso, los segmentos originales pueden ser acortados o reacomodados en otro orden, se excluye lo no deseado y se pueden agregar efectos de audio y de video (COMUNICA 2009).

La edición lineal está comprendida dentro del proceso de la postproducción audiovisual. Consiste básicamente en la realización de una "cinta" máster a partir de una serie de imágenes o pistas de sonido obtenidas de diversas fuentes (cámaras, magnetoscopios, CD, DAT, MIniDisk, etc.). Se producen transiciones, cortes y efectos especiales entre imágenes y sonidos que pueden estar grabados con anterioridad o se están obteniendo en el momento de la edición (COMPUTACIÓN 2001).

Otro concepto de edición lineal la define como la primera aproximación a la edición que todavía se encuentra en uso, es la manera más rápida de ensamblar una secuencia sencilla pero no permite la variedad de opciones que un sistema no-lineal sofisticado ofrece al editor (MADRID 2011).

Edición no lineal:

La edición no-lineal (también llamada de acceso aleatorio) es algo así como trabajar con un procesador de palabras muy sofisticado; permite insertar segmentos, eliminarlos, y cambiarlos de posición en cualquier momento durante la edición (COMUNICA 2009).

La forma no lineal es la utilizada por la tecnología digital. Esta forma de edición permite ordenar los frames[1] en el orden que se desee. Es posible tratar cualquier fotograma o cuadro de imagen de forma directa sin necesidad de seguir toda la secuencia, independiente de la forma y orden de cómo se ha obtenido el audiovisual (DIGITALFOTORED 2005).

[1] Fotograma o cuadro. Imagen particular dentro de una sucesión de imágenes que componen una animación.

Por otra parte, el Instituto Cubano de Arte e Industria Cinematográficos (ICRT) define la edición no-lineal como el montaje de imágenes y su manipulación como archivos, a través de programas informáticos. Así también, permite insertar segmentos, eliminarlos, y cambiarlos de posición (GARCÍA 2009).

Continúa precisando que durante la edición no lineal pueden agregarse una cantidad considerable de efectos especiales, entre ellos: *fades*[2], disolvencias, títulos, corrección de color por escena, recortado. También puede mejorarse el sonido utilizando filtros y efectos sonoros diversos.

Diferencias entre la edición lineal y la edición no lineal

La edición lineal permite trabajar directamente con el video original, la edición no lineal trabaja con una copia. Los soportes utilizados en la edición no lineal son de tipo informático (entiéndase DVD, DVB, archivos MPEG1-2) y se guardan en un disco duro de gran capacidad, en cambio, en la edición lineal se trabaja con cintas de video (DVCAM, DVCPRO, Digital-S). Aunque digitales, estas últimas son cintas; por lo tanto, el acceso a la información es de tipo secuencial y no aleatorio como en los formatos utilizados en edición no lineal.

Con la edición no lineal es posible producir cualquier audiovisual imaginable, producciones de gran calidad; es decir, realizar miles de fundidos, cortes y efectos especiales impensables en la edición tradicional (personajes virtuales animados, cambiar colores a un objeto en movimiento grabado anteriormente, captar movimiento real del exterior y aplicarlo a un ser virtual, entre otros) (GARCÍA 2009).

En los últimos años, como una de las alternativas de la edición no lineal ha emergido en el mundo del video y sonido digital la edición basada en ficheros de acciones comúnmente llamadas Listas de Decisión de Edición las cuales son las clasificadoras que ordenan las ediciones de cada montaje realizado.

A continuación un acercamiento a los conceptos fundamentales asociados a las Listas de Decisión de Edición.

[2] Efectos de surgimiento o desvanecimiento de la imagen o del sonido.

Listas de Decisión de Edición (EDL)

Las listas de decisión de edición no son más que el clasificador que ordena las ediciones de cada montaje realizado. Con ella el editor puede tener una referencia de todas las decisiones tomadas y de cómo han sido introducidas. Puede ser manuscrita en papel o estar hecha por ordenador y luego puede ser almacenada; incluso en los primeros tiempos de la edición y antes del desarrollo informático actual, la EDL se registraba en cinta perforada. Cuando se trabaja con equipos computarizados de off-line, el ordenador genera la EDL para grabar en cualquier dispositivo (CINE 2006).

Otra definición de EDL la ofrece la agencia de música Sweetwater inSync: una Lista de Decisión de Edición es simplemente una lista que contiene las escenas deseadas y las ediciones que se utilizarán a partir de la grabación original, unidas a los cortes que se llevarán a cabo (SWEETWATER 2011).

Por otra parte GILL considera a las listas de decisión de edición como un archivo que contiene una lista de eventos, que describen un paso en el montaje o la manipulación de contenidos en la media con el fin de crear un producto final (GILL 2007).

Los autores de la investigación asumen que una Lista de Decisión de Edición no es más que un fichero de texto o binario (o documento en copia dura), en el cual se tienen diferentes acciones a ejecutar sobre una o varias medias o audiovisuales en el proceso de postproducción.

Importancia y ventajas de las EDL

Las EDL utilizadas en el proceso de postproducción de audiovisuales permiten realizar acciones sobre un video original sin afectar su calidad ni su contenido, silenciar secciones del video durante la reproducción, permite convertir efectos en cortes, eliminar o aumentar velocidades.

Su principal utilización viene dada por la posibilidad de llevar al cliente final del audiovisual justamente lo que este desea sin afectar el material original. Una película con varias escenas de violencia puede ser rediseñada sin ellas sin necesidad de afectar el archivo original, así, entonces, sobre un mismo material se podrán tener tantas ediciones como se desee, de acuerdo al interés final. Otra ventaja viene asociada a la posibilidad de las ediciones no lineales de realizar miles de cortes, añadiduras y efectos con la misma calidad y en un mismo instante de tiempo, moviendo las decisiones de edición para el propio momento de la transmisión, aumenta la agilidad del proceso de edición en la producción del video.

El uso del material original para evaluar distintas alternativas de edición es peligroso. Si las tomas sufren algún daño durante el proceso (lo cual es muy probable que ocurra cuando se adelanta y retrocede con frecuencia), se perdería la única copia original (ROMERO 2010). Lo anterior no sucedería si se utilizaran EDL.

Partes de una EDL

Aun cuando no hay acuerdo común ni estandarización en cuanto a las partes de un archivo EDL (o su homólogo en documento duro) todos cuentan con un instante de tiempo inicial, uno final y la acción a ejecutar en el intervalo definido por ellos.

MPlayer Team considera a las EDL conformadas por el segundo de inicio, el segundo final y la acción a realizar. Los segundos son números de punto flotante y la acción es o bien 0 para saltar esa parte o 1 para silenciarla. A continuación un ejemplo:

[Segundo de inicio]	[Segundo final]	[Acción]
5.3	7.1	0
15	16.7	1
420	422	0

Otros de los ejemplos de EDL son las que están compuestas por eventos, cada evento está compuesto por un máximo de dos líneas, en que cada línea tiene su código de tiempo, (Time Code), que índica desde donde y hasta cuando dura la línea. Además, este formato de EDL considera los comentarios, en éstos se puede conocer el tipo de la EDL, las modificaciones que se han realizado sobre ella, la velocidad, el título, etc. A su vez cada línea puede tener también sus propios comentarios (ROMERO 2010).

Otro de los tipos de EDL contiene la información de todas y cada una de las ediciones que, por sencillas que sean, implican como mínimo los siguientes datos:

1. Número de la cinta reproducida.
2. Señalización de las pistas editadas: audio 1, audio 2 y/o video.
3. Nombre de la función programada: corte, encadenado.
4. Tiempos de entrada y salida de la cinta grabadora (ROMERO 2010).

Formatos de las EDL

En la actualidad, existen diferentes formatos que se utilizan para la creación de archivos EDL, dentro de los más utilizados se encuentran los formatos CMX, CMX-340, CMX-360 y CMX-3600.

La mayor diferencia entre ellos se encuentra en la descripción de sus campos, regularmente a la hora de identificar el Tape Name[3], dependiendo de un formato u otro se identifica de una manera u otra. Por ejemplo en el formato CMX-340 que es muy utilizado, se tiene que el Tape Name debe de ser un número entre el 1 y el 251, en CMX-360 debe de ser un número entre el 1 y el 998, y el CMX-3600 pueden ser números y letras, pero no pueden contener espacios en blanco ni caracteres especiales (ROMERO 2010).

1.3. Descripción general de la transmisión de audiovisuales

Según la definición brindada por la Real Academia de la Lengua Española, la **transmisión** se refiere a algunos de los significados siguientes: Trasladar, transferir / Dicho de una emisora de radio o de televisión: Difundir noticias, programas de música, espectáculos, etc / Hacer llegar a alguien mensajes o noticias / Comunicar a otras personas enfermedades o estados de ánimo / Conducir o ser el medio a través del cual se pasan las vibraciones o radiaciones / En una máquina, comunicar el movimiento de una pieza a otra / Enajenar, ceder o dejar a alguien un derecho u otra cosa (ESPAÑOLA 2010).

Por otra parte, el popular diccionario *Definición ABC* define la **transmisión** como el traspaso de energía, ondas o información desde un punto de inicio hacia un punto de llegada diferente, pudiendo alterarse o no aquello que es transmitido en el recorrido (ABC 2011).

La transmisión de audiovisuales es uno de los posibles escenarios de transmisión y ha sido descrito en los dos conceptos anteriores como transmisión de información (en este caso información audiovisual).

El sector de Normalización de las Comunicaciones de la Unión Internacional de las Telecomunicaciones define la transmisión de datos como la acción de cursar datos, a través de un medio de telecomunicaciones, desde un lugar en que son originados hasta otro en el que son recibidos (UNION 2009).

[3] Nombre de la cinta.

A partir de la unión de las definiciones anteriores, en el contexto de esta investigación, los autores definen la **transmisión audiovisual** como el proceso por el cual se trasladan o traspasan los contenidos audiovisuales desde un punto de inicio hasta un punto de llegada diferente a través de un medio de telecomunicación determinado y con un fin específico.

La transmisión de datos tiene como objetivos fundamentales:

- Reducir tiempo y esfuerzo.
- Aumentar la velocidad de entrega de la información.
- Reducir costos de operación.
- Aumentar la capacidad de las organizaciones a un costo incremental inteligente.
- Aumentar la calidad y cantidad de la información (SUPERIORES 2010).

Transmisión analógica y digital

En sentido general, la transmisión de datos, según la naturaleza del dato, de acuerdo con las principales definiciones del área, se realiza de manera digital o de manera analógica, en correspondencia con la intención, finalidad y características propias del entorno.

La *transmisión analógica* es una forma de transmitir las señales analógicas independientemente de su contenido; estas señales pueden representar datos analógicos (por ejemplo la voz) o datos digitales (por ejemplo, los datos binarios modulados en un modem). En cualquier caso, la señal analógica se irá debilitando (atenuándose) con la distancia (STALLING 2000).

La información contenida en una señal de transmisión analógica es representada por algunas características de la misma señal tales como amplitud, frecuencia o fase del voltaje de la señal. Las transmisiones de señales analógicas son aquellas que se escuchan diariamente en la radio de AM o FM citando un ejemplo (ESPARTA 1997).

La *transmisión digital*, por su parte, es dependiente del contenido de la señal. Una señal digital solo se puede transmitir a una distancia limitada, ya que la atenuación y otros aspectos negativos pueden afectar a la integridad de los datos transmitidos. Para conseguir distancias mayores se usan repetidores. Un repetidor recibe la señal digital, regenera el patrón de ceros y unos y los retransmite. De esta manera se evita la atenuación (STALLING 2000).

El contenido de información de una señal digital está relacionado con estados discretos de la señal y puede estar en forma digital o en forma analógica (esta última debe convertirse a pulsos digitales antes de su transmisión y devolverla nuevamente a la forma analógica una vez que llegue a su destino) (ESPARTA 1997).

A pesar de que los sistemas de transmisión analógica han absorbido grandes inversiones, la industria de las telecomunicaciones y los usuarios han optado por la transmisión digital. Tanto las comunicaciones a larga distancia como los servicios de comunicación a distancias muy cortas (un ejemplo podría ser la distancia entre edificios) están implementando gradualmente la transmisión digital, más aún, se está introduciendo la señalización digital en todos los sistemas donde se considere factible. En los párrafos siguientes se especifican las razones más importantes que justifican esta elección:

Tecnología digital: las mejoras en las tecnologías de integración a grande y muy grande escala se traducen en una disminución continua tanto en coste como en el tamaño de la circuitería digital. El instrumental analógico no ha experimentado una reducción similar.

Integridad de los datos: al usar repetidores en lugar de amplificadores, el ruido y otros efectos negativos no son acumulativos. Es por esto que, usando tecnología digital es posible transmitir datos conservando su integridad a distancias mayores utilizando incluso líneas de calidad inferior.

Seguridad y privacidad: las técnicas de encriptación se pueden aplicar fácilmente a los datos digitales, o a los analógicos que se hayan previamente digitalizado.

Integración: en el tratamiento digital de datos analógicos y digitales, todas las señales tienen igual forma y pueden ser procesadas de una forma similar. Este hecho posibilita la integración de voz, video y datos usando la misma infraestructura (STALLING 2000).

Como parte de la transmisión digital, nace en el año 1995 la tecnología *streaming* revolucionando el mundo del audio y video sobre internet. Antes de esta fecha, la reproducción de contenidos audiovisuales sobre la red requería la descarga completa del material a visualizar. Con el surgimiento de la tecnología *Streaming* esta limitación quedó resuelta al no tener que necesariamente descargar todo el video a visualizar.

Streaming

El streaming es una importante tecnología que permite la transmisión y recepción de contenido audiovisual a través de radio y televisión por Internet. Una tecnología económica de fácil implementación y de muchas utilidades (BERROTERÁN 2011).

Significa que el contenido está disponible mientras se sigue descargando en su totalidad. Este concepto es aplicable tanto a audio, video o, incluso, a una aplicación flash (QUALITYNET 2006).

La tecnología streaming se utiliza para aligerar la descarga y ejecución de audio y video en la web, ya que permite escuchar y visualizar los archivos mientras se están descargando (WEB 2001).

Un servidor de "media streaming", entendido como tal, es un elemento muy valioso para actividades de tele formación. Permite ofrecer como recurso educativo verdaderas presentaciones virtuales multimedia, en directo o como video a demanda (GÓMEZ 2006).

El proceso de streaming se divide en dos categorías fundamentales para la transmisión, en función de cómo se obtiene la información a difundir: en vivo o bajo demanda (VIRTUAL 2010).

Transmisión de audiovisuales en tiempo real

En este documento, luego de un análisis pormenorizado, se utiliza la siguiente definición de transmisión en tiempo real, introducida por Ferrari: “Se denomina servicio de Transmisión en Tiempo Real a aquel cuyos clientes pueden especificar los requisitos sobre prestaciones y obtener garantía del cumplimiento de estos requisitos” (VIRTUAL 2010).

Las imágenes y el sonido son digitalizados y retransmitidos en tiempo real a Internet. En este caso, los usuarios pueden seguir el desarrollo de un evento en el mismo momento que éste se está produciendo. Es lo que ocurre con la videoconferencia y las difusiones en directo. En este caso, es el servidor el que se encarga de controlar la transmisión de los datos (ALEMAN 2007).

Los autores consideran que existen tres métodos o formas fundamentales de transmisión de audiovisuales que se agrupan dentro de la Transmisión en tiempo real, estos son: *Broadcast, Unicast y Multicast*.

BroadCast (Difusión)

Mediante el método broadcast se realiza una transmisión simultánea de datos a una audiencia de gran volumen. Los clientes que reciben una difusión no pueden controlar el inicio del contenido ni la velocidad de reproducción, ni detener ni avanzar de forma rápida, ni tampoco rebobinar la secuencia. Es el servidor el que tiene el control de la secuencia. Los contenidos pueden ser creados en ese momento en vivo (*live broadcast*), o almacenados previamente en el servidor. El sistema de difusión tiene analogías con los canales de TV (GÓMEZ 2006).

UniCast (Unidifusión)

En la transmisión *unicast* cada cliente inicia su propio streaming, independientemente de que todos los clientes estén interesados en el mismo streaming de datos (esto es, aunque sea una difusión "en tiempo real"), de forma que se inician muchas conexiones uno-a-uno (una entre el servidor y el cliente por cada uno de los clientes) (VALENCIA 2007).Transmite flujos desde el emisor para cada petición realizada por el receptor al servidor.

MultiCast (Multidifusión)

En el *multicast* se establece una relación de uno a muchos entre un servidor de streaming y los clientes que reciben el flujo. Con una secuencia de multidifusión el servidor transmite a una dirección *multicast* IP[4] en la red y los clientes reciben el flujo al subscribirse a la dirección IP. Todos los clientes reciben el mismo flujo y no tienen el control de la reproducción del contenido. Sólo existe una corriente desde el servidor, independientemente del número de clientes que reciben el flujo. El uso de un flujo de multidifusión conserva el ancho de banda y puede ser útil para las redes de área local con bajo ancho de banda (MICROSOFT 2007).

Transmisión de audiovisuales bajo demanda

En la transmisión bajo demanda las imágenes y el sonido proceden de un fichero digitalizado y almacenado en un servidor de media streaming, de modo que los usuarios pueden solicitar su

[4] Internet Protocol.

visualización a través de Internet en cualquier momento. En este caso es el cliente el que controla la transmisión y recepción del contenido multimedia (ALEMAN 2007).

La transmisión de video bajo demanda no es más que un conjunto de peticiones realizadas por clientes individuales a ficheros almacenados en el servidor, cada cliente que solicita una secuencia suele tener el control total de la misma y puede aplicar las características de avance rápido, rebobinado, pausa y reinicio del contenido. Esto se debe a que se proporciona una sola ruta de acceso a los datos para cada cliente que solicita el contenido (GÓMEZ 2006).

1.4. Soluciones existentes que responden al problema de la investigación

En el Departamento de Señales Digitales, donde se origina esta investigación, se desarrolla un producto denominado *Sistema de gestión y transmisión de contenidos audiovisuales* (SIAV) que contiene la transmisión de audiovisuales.

El producto de referencia constituye un sistema capaz de automatizar procesos asociados a la gestión, procesamiento y transmisión de contenidos audiovisuales de cualquier entidad dedicada a esta rama. Se compone de tres productos que pueden desplegarse independientemente o integrados: SIAV Primicia, SIAV WebTV y SIAV Transmisión. Además, posee facilidades para la personalización.

SIAV Primicia, plataforma de televisión informativa

Formado por dos subsistemas (Subsistema de Administración y Subsistema de Transmisión) que en su conjunto constituyen el canal informativo o la plataforma de televisión informativa como se le conoce.

De los dos subsistemas antes mencionados el Subsistema de Transmisión es el que se relaciona con el objeto de estudio de la investigación debido a que se encarga de la visualización de las noticias y materiales publicados. Durante las transmisiones permite la reproducción de un fondo musical que puede ser personalizado según la noticia que se muestra, además, es posible la utilización de cintillos informativos o info-cintas que permiten el adelanto o emisión de breves informaciones de carácter relevante o promocional (DIGITALES 2012).

Primicia reproduce un flujo de streaming únicamente cuando se transmite un audiovisual durante la noticia, en otro caso se limita, a través de la tarjeta exportadora de video, a visualizar en varios destinos (televisores, datashow u otros medios para mostrar información) el contenido de la noticia que se muestra en la aplicación original, es decir, la corrida de la aplicación se difunde a través de la tarjeta de red.

Lo anterior, lógicamente, limita las posibilidades de Primicia debido a que un error en tiempo de ejecución es visualizado por los receptores al mismo tiempo que ocurre sin posibilidades de resolverlo o no mostrarlo.

SIAV Transmisión, plataforma de transmisión abierta de televisión

Constituye la plataforma de transmisión abierta de la solución, está compuesto por tres subsistemas fundamentales: programación, transmisión y monitorización.

El subsistema de transmisión provee la transmisión a través de un flujo de streaming de los recursos multimedia según la planificación previamente establecida en el subsistema de programación. Las transmisiones se realizan de forma automática atendiendo a las especificaciones realizadas durante la planificación y programación de la transmisión. Además provee facilidades para tomar el control de dichas transmisiones y administrar el flujo de streaming (DIGITALES 2012).

SIAV WebTV, plataforma de publicación de contenidos audiovisuales en la web

Constituye la plataforma web para la publicación y transmisión de contenidos audiovisuales bajo demanda o en vivo. Está compuesto por dos subsistemas fundamentales, administración y presentación.

En el subsistema de presentación se provee el acceso a los materiales publicados haciendo uso de la filosofía de video y audio bajo demanda. También es posible la visualización en vivo de los canales trasmitidos o de una señal externa (DIGITALES 2012).

1.4.1. Propuesta tecnológica a partir del análisis de las soluciones existentes

Este producto, de manera general, presenta varias variantes para la transmisión de audiovisuales, pero ninguna de ellas se realiza teniendo en cuenta archivos EDL, significando esto un punto de mejora identificado para implementaciones futuras.

En el ámbito internacional, existen varias soluciones de software que contemplan el trabajo, creación y edición de ficheros EDL, pero en ninguno de los casos consultados estas soluciones realizan la transmisión del audiovisual utilizando estos archivos.

Un ejemplo de lo anterior es la ***Avid EDL Manager*** (TECHNOLOGY 2001) y **Final Cut Pro** (APPLE 2012). Estas dos aplicaciones se suman a la amplia gama de soluciones que utilizan, administran, exportan o importan ficheros EDL, no obstante, luego de un exhaustivo análisis bibliográfico los autores concluyen que no existen soluciones que respondan al problema de la investigación, lo que constituye un aporte de facto.

La principal limitación radica en la imposibilidad de crear el flujo de streaming una vez realizadas las modificaciones al video original de acuerdo al EDL para luego ser trasmitido.

Luego del análisis de las soluciones anteriores, y teniendo como base la imposibilidad de contar con la transmisión del audiovisual de alguna herramienta conocida, disponible o reutilizable, los autores de la investigación proponen desarrollar el proceso de transmisión de acuerdo a la variante propuesta por SIAV Primicia.

Esta variante realiza la transmisión a partir de una tarjeta exportadora de video, limitando el desarrollo de la solución a la reproducción del material audiovisual de acuerdo a las acciones propuestas por el fichero EDL, moviendo la responsabilidad de la transmisión a la tarjeta de video que se convertirá en requisito indispensable para la explotación de la solución.

1.5. Conclusiones parciales

Durante este capítulo se desarrolló un análisis de los elementos teóricos que sirven como base al problema científico y a los objetivos planteados en el presente trabajo de diploma. Además de lo anterior,

1. El estudio minucioso de la bibliografía y luego de un análisis pormenorizado de lo publicado en Internet, arrojó que las soluciones que existen no se corresponden con los objetivos de esta investigación.
2. La no identificación de un componente, biblioteca o subsistema para la creación y transmisión del flujo streaming producto de la edición del video de acuerdo al EDL propicia la utilización de la variante de solución propuesta por el SIAV Primicia que transmite el audiovisual a través de la tarjeta de video, circunscribiendo la responsabilidad de la herramienta solamente a la reproducción del audiovisual.

CAPÍTULO 2
HERRAMIENTAS Y TECNOLOGÍAS

2.1. Introducción

Para obtener un sistema con calidad, es necesario utilizar herramientas y tecnologías que agilicen el proceso de desarrollo de software y que estén enfocadas en la obtención de un producto. En el presente capítulo se describen las herramientas y tecnologías que se seleccionaron para el desarrollo e implementación del sistema. Se hace un análisis de las metodologías de desarrollo de software que existen, lenguajes de programación, herramientas CASE[5], lenguaje de modelado, frameworks y el entorno de desarrollo integrado, conformando así la propuesta de solución de esta investigación.

2.2. Metodología de Desarrollo y Lenguaje empleado

Una metodología de desarrollo de software es un conjunto de procedimientos, técnicas y soporte documental bien organizados, como marco de trabajo para la gestión y control del desarrollo de aplicaciones o sistemas informáticos. Las metodologías persiguen como característica fundamental, uno o varios modelos del ciclo de vida del software, indicando en cada fase los artefactos a obtener a lo largo del desarrollo del proyecto, enfocados en elevar la calidad del producto final.

2.2.1. Rational Unified Process

Proceso Unificado de Software (RUP por sus siglas en inglés, *Rational Unified Process*) es un marco de procesos altamente configurables para satisfacer necesidades específicas, implementando las mejores prácticas para el desarrollo de software. Es una metodología pesada que proporciona un enfoque disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo e implementa además el paradigma de Programación Orientada a Objetos o POO (OOP según sus siglas en inglés, *Object Oriented Programming*) y utiliza el Lenguaje Unificado de Modelado (UML según sus siglas en inglés, *Unified Modeling Language*) como lenguaje para la representación visual para construir, especificar y documentar el sistema. Su principal meta es asegurar la producción de software de alta calidad que resuelva las necesidades de los usuarios a partir de presupuestos y

[5] Computer Aided Software Engineering o Ingeniería de Software Asistida por Computadora.

tiempos establecidos, además que puede especializarse para una gran variedad de sistemas y diferentes áreas de aplicación, tipos de organizaciones y/o tamaños de proyecto (JACOBSON 2000a).

Principales características de RUP:

- Dirigido por casos de uso**:** Los casos de uso guían los procesos de diseño, implementación y prueba. Estos constituyen un elemento integrador y una guía del trabajo. Además proporcionan un hilo conductor, permitiendo establecer trazabilidad entre los artefactos que son generados en las diferentes actividades del proceso de desarrollo.
- Centrado en la arquitectura**:** La arquitectura de un sistema es la organización o estructura de sus partes más relevantes, lo que permite tener una visión común entre todos los involucrados (desarrolladores y usuarios) y una perspectiva clara del sistema completo, necesaria para controlar el desarrollo.
- Iterativo e incremental**:** RUP divide el trabajo en partes más pequeñas o mini proyectos. Permitiendo que el equilibrio entre casos de uso y arquitectura se vaya logrando durante cada mini proyecto, así durante todo el proceso de desarrollo.

Las cuatro fases de RUP son:

1. Inicio: Se encarga de describir el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.
2. Elaboración: Se encarga de establecer una base para la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.
3. Construcción: Completar el desarrollo del sistema basado en la línea base de la arquitectura. Se obtiene 1 o varios reléase o versiones del producto que han pasado las pruebas.
4. Transición: Garantizar que el software está listo para entregarlo a los usuario. El producto ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

Además de las fases mencionadas anteriormente, RUP también cuenta con 9 flujos, los primeros seis son conocidos como flujos de trabajo y los tres últimos como flujos de apoyo. La figura que se muestra a continuación contempla los flujos y las fases inmersos en esta metodología (JACOBSON 2000a).

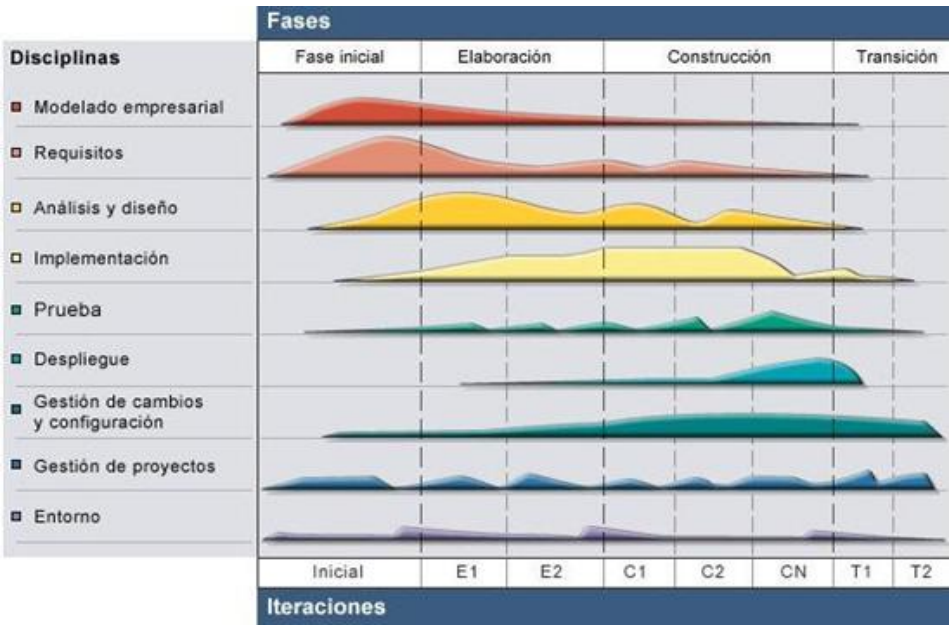

Figura 1: Fases y Flujos de trabajo de RUP

2.2.2. Extreme Programming

La metodología Programación Extrema (XP por sus siglas en inglés, *Extreme Programming*) es una metodología ágil centrada en fortalecer las relaciones entre el cliente y el equipo de desarrollo, como el eslabón más débil para el éxito en el desarrollo de software, encaminando el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, propiciando un buen clima de trabajo y sencillez en las soluciones implementadas.

Se basa también, en la idea de que existen cuatro variables que guían el desarrollo de sistemas: costo, tiempo, calidad y alcance, haciéndolas visibles durante todo el proceso de desarrollo.

XP intenta implementar una forma de trabajo donde se adapte fácilmente a las circunstancias, es decir, el desarrollador decide cómo se implementan los procesos, crea el sistema con la mejor calidad posible y pide al cliente en cualquier momento aclaraciones de los requisitos.

Fundamentalmente trabaja estrechamente con el cliente, haciendo pequeñas iteraciones, cada dos semanas, donde no existe más documentación que el código en sí; cada versión contiene las modificaciones necesarias según el cliente vaya retroalimentando el sistema (por eso es necesaria la disponibilidad del cliente durante todo el desarrollo).

Otra característica de XP es que utiliza historias de usuario, que no son más que técnicas para especificar los requisitos funcionales y no funcionales del software, es decir, describen brevemente las características y funciones que debe poseer el sistema, por lo que cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas. Esta metodología solo basa su trabajo en el desarrollo de software y no en todo el ciclo de vida del proyecto.

Un factor que debilita la utilización de XP en proyectos de software lo representa el alto costo que significa fallar en la construcción del sistema pues no se genera prácticamente documentación durante el desarrollo, es por ello que se recomienda utilizarla fundamentalmente en proyectos de corta duración (PÉREZ 2008).

2.2.3. Microsoft Solution Framework

La metodología Microsoft Solution Framework (MSF por sus siglas en inglés), es una metodología ágil enfocada en dirigir proyectos o soluciones de innovación. En ella no se detalla ni se hace énfasis en la organización o el tamaño del equipo de desarrollo, está más bien centrada en la gestión y administración del proyecto para lograr el impacto deseado.

Esta es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. MSF se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas.

Características de MSF

1. **Adaptable**: es parecido a un compás, usado en cualquier parte como un mapa, del cual su uso es limitado a un específico lugar.
2. **Escalable**: puede organizar equipos tan pequeños entre 3 o 4 personas, así como también, proyectos que requieren 50 personas o más.
3. **Flexible**: es utilizada en el ambiente de desarrollo de cualquier cliente.
4. **Tecnología Agnóstica**: porque puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología.

Esta metodología está compuesta por varios modelos que se encargan en planificar las diferentes partes en el desarrollo de un proyecto. Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y finalmente el Modelo de Aplicación (SANCHEZ 2004).

Selección de la Metodología de Desarrollo: *RUP*

Una vez estudiadas y analizadas las características de estas metodologías se seleccionó utilizar la metodología RUP. Dicha decisión está fundamentada por las siguientes características que aportan a la investigación el peso necesario para su utilización en la misma. Su uso es aplicable a variedad de aplicaciones informáticas y determina un grupo de actividades que guían los esfuerzos de las personas implicadas en el proyecto.

- Este trabajo en equipo genera una gran cantidad de documentación para hacer más entendible todos los procesos y resulta excelente su aplicación en la implementación de proyectos grandes.
- Al suponer una minuciosa planificación de las tareas que se llevarán a cabo, de qué manera y cómo se realizarán así como quién será el encargado de ejecutarlas, traerá como consecuencia que se cumplan las metas establecidas y se consuma el presupuesto y calendario de desarrollo del proyecto concebido.

Por otra parte, las metodologías anteriores, no enfatizan en el proceso de desarrollo de software sino más bien en el desarrollo del producto como tal, en la investigación actual, se necesita la documentación de todo el proceso para que pueda ser usada por el equipo de desarrollo para posteriores modificaciones y/o ampliaciones.

2.2.4. El Lenguaje Unificado de Modelado

Una vez seleccionada la metodología que guiará el desarrollo del software, se hace necesario definir qué lenguaje utilizar para modelar el sistema. Debido a que esta metodología de desarrollo utiliza como lenguaje de modelado el Lenguaje Unificado de Modelado (UML por sus siglas en inglés, *Unified Modeling Language*), para esta investigación se define utilizar dicho lenguaje de modelado para el diseño de los diagramas necesarios en la implementación de la propuesta.

UML es un lenguaje estándar para el modelado de software que permite a los desarrolladores visualizar los resultados de su trabajo en esquemas o diagramas estandarizados. Este se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software como es el caso de RUP, pero no especifica en sí mismo qué metodología o proceso usar. Es importante destacar que un modelo UML describe lo que supuestamente hará un sistema, pero no indica cómo se debe implementar.

Características de UML:

- Es independiente del proceso, para utilizarlo de forma óptima se debe usar en un proceso que sea *dirigido por casos de uso, centrado en la arquitectura e iterativo e incremental* (LARMAN 1999b).
- Permite modelar sistemas utilizando *técnicas orientadas a objetos* (POO).
- Permite especificar todas las *decisiones de análisis*, *diseño* e *implementación*, construyéndose así modelos precisos, no ambiguos y completos.
- Permite documentar todos los *artefactos* de un *proceso de desarrollo* (requisitos, arquitectura, pruebas, versiones, etc.).
- Es un lenguaje muy expresivo que cubre todas las vistas necesarias para *desarrollar* y luego *desplegar* los sistemas.

2.3. Herramienta CASE

Las herramientas CASE (*Computer Aided Software Engineering por sus siglas en Inglés*), fueron desarrolladas para automatizar procesos y facilitar las tareas de coordinación de los eventos que necesitan ser mejorados en el ciclo de desarrollo de software. Se enfoca en un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo.

2.3.1. Visual Paradigm

Es una herramienta CASE que permite recorrer el ciclo de vida completo del desarrollo de software, desde el modelado de negocio hasta el despliegue del producto, generando diferentes artefactos en su trayectoria. Es una *herramienta libre* o *software libre* de probada utilidad para el analista, que permite generar diagramas de clases, código y documentación, además que provee tutoriales de UML, así como demostraciones interactivas sobre dicho lenguaje.

Se caracteriza por su usabilidad y portabilidad, además que está respaldado por una amplia bibliografía y material audiovisual. Ofrece soporte a varios usuarios, para trabajar en la misma plataforma durante su uso; presenta licencia gratuita y comercial, es multiplataforma, desarrollado para diseñar software según el paradigma de POO, además que permite exportar e importar los diagramas en formatos XML e imágenes y que contiene los elementos necesarios para aplicar la metodología RUP.

Visual Paradigm es compatible con varios lenguajes en generación e ingeniería inversa. Sus principales características son:

- ***Ingeniería inversa de bases de datos:*** Desde Sistemas Gestores de Bases de Datos (DBMS) existentes a diagramas de Entidad-Relación.
- ***Soporte ORM:*** Generación de objetos Java desde la base de datos.
- ***Distribución automática de diagramas:*** Reorganización de las figuras y conectores de los diagramas UML.
- ***Importar/ exportar:*** de ficheros XML.
- ***Integración con Visio:*** Dibujo de diagramas UML con plantillas (stencils) de MS Visio.
- ***Generación de bases de datos:*** Transformación de diagramas de Entidad-Relación en tablas de base de datos (VIZCAÍNO 2000).

2.3.2. Rational Rose Enterprise Edition

Es una herramienta que se encarga de llevar a cabo la realización de los diferentes diagramas y generación del código posterior, es decir, se pueden generar códigos en distintos lenguajes de programación a partir de un diseño en UML. Provee además, mecanismos para realizar la denominada *Ingeniería Inversa*, a partir del código de un programa, se puede obtener información sobre su diseño. Es una herramienta propietaria que utiliza como plataforma el sistema operativo Windows, lo que significa que es necesario pagar por su licencia para utilizarla. Conjuntamente no es compatible en el Sistema Operativo GNU/Linux lo cual limita su uso.

Rational Rose brinda la ayuda para la comprensión del sistema y de sus distintos componentes, siempre que estén orientados a objetos. Estos componentes van a contener una serie de archivos dentro de los cuales se encuentran las distintas clases pertenecientes ha dicho componente. Mediante la especificación de la sintaxis que presentan dichos ficheros, se realiza de forma automática la

ingeniería inversa. Se plantea que presenta una pequeña desventaja, y es que necesita de mucha memoria para poder de alguna forma ser manejado de forma rápida y eficiente (EGAS 2007).

Selección de la Herramienta CASE: *Visual Paradigm*

Después de haber realizado el estudio de estas dos herramientas, se determinó que la herramienta aplicable al modelado de la ingeniería de software del sistema que se propone, es Visual Paradigm. Para su selección se tuvieron en cuenta las cualidades siguientes: facilita la codificación desde diagramas así como el desarrollo de documentación de una aplicación, posee una amplia bibliografía tanto en material audiovisual como documentación digital que la respalda.

Es una herramienta libre lo que supone que no hay que pagar licencia por su uso y distribución; el hecho de ser multiplataforma garantiza su ejecución tanto en Windows como en diferentes distribuciones de Linux; es fácil de instalar así como de actualizar hacia las nuevas versiones liberadas por la comunidad de desarrollo que le da soporte técnico y de desarrollo.

2.4. Lenguaje de Programación

Un lenguaje de programación es un conjunto de sintaxis y reglas semánticas que definen los programas del computador. Es una técnica estándar de comunicación para entregarle instrucciones al computador. Un lenguaje le brinda la capacidad al programador de especificarle al computador, qué tipo de datos actúan y qué acciones tomar bajo una variada gama de circunstancias, utilizando un lenguaje relativamente próximo al lenguaje humano (NACIONAL 2012).

2.4.1. C++

Es un lenguaje de Programación creado a mediados de los años 80 en la ciudad de New Jersey, por Bjarne Stroustrup con el fin de extender el lenguaje de programación C, una de las propuestas más completas permitiendo la manipulación de objetos como un lenguaje totalmente híbrido. Se indica que C++ es un lenguaje *multiparadigma*, gracias a la unión de paradigmas, dos de ellos la *programación estructurada* y la *programación orientada a objetos*.

Facilita la programación y el mantenimiento de sistemas de software de grandes proporciones. Se puede programar desde sistemas operativos, compiladores, aplicaciones que sean de base de datos hasta procesadores de texto. Es un lenguaje versátil, potente y general.

En la actualidad consta de nuevos tipos de datos, clases, plantillas, mecanismo de excepciones, sistema de espacios de nombre, funciones online, referencias, operadores para manejo de memoria persistente, entre otras utilidades adicionales de librería (MÁLAGA 2010).

2.4.2. Java

Es un lenguaje joven de programación orientado a objetos y multiplataforma, lo que significa que no está enlazado a un sistema operativo en concreto y los programas desarrollados con el funcionaran correctamente tanto en Windows como en Linux. Cuenta con una arquitectura neutra y con una buena seguridad, además de ser interpretado, robusto y de alto desempeño.

Fue desarrollado por la empresa Sun Microsystems. Permite el desarrollo de aplicaciones bajo el esquema o arquitectura cliente-servidor, como de aplicaciones distribuidas, siendo capaz de conectar computadoras u ordenadores ejecutando tareas simultáneamente, permitiéndole lograr una distribución organizada del trabajo a realizar y la modularidad, por lo que se pueden hacer rutinas individuales que sean usadas por más de una aplicación.

Una de las principales características que ha favorecido la difusión y el crecimiento del lenguaje Java, es su capacidad de que el código funcione sobre cualquier plataforma de software y hardware, esto quiere decir que si se realiza un programa en Java podrá funcionar en cualquier ordenador del mercado. Java presenta algunas desventajas como son:

1. Eficiencia: Java es hasta 30 veces más lento que C++ a causa del tiempo invertido en:
- Recogida de basura.
- Sincronización de threads (hilos de ejecución).
- Otras actividades (carga de clases, comprobación de límites, entre otros).
2. No posee herencia múltiple.

2.4.3. C#

C# es un lenguaje desarrollado por Microsoft muy potente, con propósito general y de programación independiente. Implementa el paradigma de la Programación Orientada a Objetos, es simple, elegante y con seguro en el tratamiento de tipos, que permite a los programadores de aplicaciones empresariales crear una gran variedad de aplicaciones.

Es concebido como un lenguaje nativo de su famosa plataforma .Net para aplicaciones web y de escritorio, trata principalmente tanto aspectos de C++ como de Java y Visual Basic, pero de una forma más versátil y mejorada agregándole cada vez más elementos que faciliten su uso. C# presenta algunas desventajas como que requiere de *forma obligatoria del framework .Net* para poder ejecutar sus aplicaciones, además su principal IDE de programación, Visual Studio, es propietario.

Selección del Lenguaje de Programación: *C++*

Una vez estudiado lo referente a los lenguajes de programación expuestos anteriormente, se selecciona C++ para utilizar en la implementación del sistema. Este lenguaje posee grandes potencialidades, además de ser compatible en varias plataformas incluyendo Linux.

Tiene características que lo ubican por encima de los lenguajes mencionados anteriormente, además, en el caso de Java, las aplicaciones resultantes son mucho más lentas que las derivadas de C++ y tienen dependencia de la máquina virtual, por lo que su uso no es el más adecuado para los propósitos finales de la herramienta que se desarrolla.

Otro de los aspectos por lo cual se selecciona este lenguaje de programación es debido a que se encuentra entre la arquitectura que se definió en el Departamento de Señales Digitales para desarrollar sus soluciones, además, el proyecto para el cual se desarrolla esta investigación y sus componentes también están implementados sobre el mencionado lenguaje.

2.5. Framework de desarrollo

Un framework o marco de trabajo es un software que se puede personalizar e intercambiar para favorecer en tiempo el desarrollo de una aplicación. Se puede considerar como otra aplicación incompleta y configurable a la que se le añaden las últimas piezas para construir una aplicación final.

Ayudan a desarrollar aplicaciones con mayor rapidez, pues poseen una estructura definida y una organización para el desarrollo y mantenimiento del software desarrollado.

Las principales ventajas de la utilización de un framework son:

- El desarrollo rápido de aplicaciones. Los componentes incluidos en un framework constituyen una capa que libera al programador de la escritura de código de bajo nivel.
- La reutilización de componentes software al por mayor. Los frameworks son los paradigmas de la reutilización.
- El uso y la programación de componentes que siguen una política de diseño uniforme. Un framework orientado a objetos logra que los componentes sean clases que pertenezcan a una gran jerarquía de clases, lo que resulta en bibliotecas más fáciles de aprender a usar (MORÁN 2009).

2.5.1. Qt

Para el desarrollo de la propuesta de solución se trabaja con el framework de desarrollo Qt (Quasar Technologies), definido igualmente entre las tecnologías del Departamento de Señales Digitales y el proyecto. Es una biblioteca de software multiplataforma ampliamente difundida para el desarrollo de aplicaciones con interfaz gráfica de usuario y, en menor medida, pero utilizada igualmente, para el desarrollo de programas sin interfaz gráfica como herramientas de comandos y consolas para servidores.

Desarrollada por Haavard Nord y Eirik Chambe-Eng en el año 1991, la primera versión solo estaba disponible para Windows. En 1994 se crea la compañía Tolltech, bajo el nombre Quasar Technologies y disponible para sistemas tipo Unix con el servidor gráfico X Windows System (Linux, BSDs, Unix), para Apple Mac OS X, para sistemas Microsoft Windows, y para dispositivos que utilizan Windows. Utiliza el lenguaje de programación C++ lo que permite un conjunto de herramientas para su mejor uso (GUTIÉRREZ 2008).

Ventajas de QT:

- Es multiplataforma.
- Es Software Libre.
- Posee cientos de recursos y guías en Fórum Nokia.
- Maneja efectos: transparencias, sombras, etc, de forma fácil.

- Ide Drag and Drop. Menos código y resultados rápidos.
- Soporte full para la plataforma Symbian.
- Soporta librerías que permiten el acceso a las medias, ficheros, XML y BD (TUXINFO 2012).

2.6. Entorno de desarrollo integrado

Un Entorno de Desarrollo Integrado (IDE) es un programa compuesto por un conjunto de herramientas para un programador. Provee un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Java, C#, Basic, Object Pascal, pudiendo utilizarse en el mismo uno o varios lenguajes de programación.

2.6.1. Qt Creator

Es multiplataforma, permite la creación de aplicaciones complejas utilizando el lenguaje de programación orientado a objetos C++. Está disponible para las plataformas de desarrollo Linux, Mac OSX, Windows, Windows CE, Symbian y Maemo. Se encuentra basado en la librería Qt, se caracteriza por ser abierto, gratuito y muy eficiente.

Principales características de Qt Creator:

- Posee un avanzado editor de código C++.
- Posee también una GUI integrada y diseñador de formularios.
- Herramienta para proyectos y administración.
- Ayuda sensible al contexto integrado.
- Depurador visual.
- Resaltado y auto-completado de código.
- Soporte para refactorización de código.

Es distribuido bajo tres tipos de licencia GPL, para el desarrollo de software de código abierto y software libre. Los cambios realizados al código fuente de Qt deben ser compartidos con la comunidad.

LGPL[6], licencia gratuita para aplicaciones comerciales de código cerrado. Los cambios realizados al código fuente de Qt deben ser compartidos con la comunidad (ECURED 2011b).

2.6.2. NetBeans

Nació en un proyecto estudiantil en la República Checa en 1996, su nombre original era Xelfi, tiempo después Jarda Tulach miembro del equipo propone el nombre de NetBeans. Es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. NetBeans IDE es un producto libre y gratuito sin restricciones de uso, el código fuente está disponible para su reutilización de acuerdo a la licencia GNU General Public License (GPL) en su versión 2 (NETBEANS 2012).

Principales características de NetBeans:

- Mejoras en el editor de código.
- Instalación y actualización más simple.
- Características visuales para el desarrollo web.
- Posee mejoras para SOA y UML.
- Soporte para PHP (CERDA 2009).

Consiste en una plataforma para construir aplicaciones completas para el cliente, permitiendo crear ventanas, menús, barras de herramientas y acciones fácilmente. Permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Además, está disponible para los sistemas operativos Windows, Mac OS, OpenSolaris, Linux y Java.

2.6.3. Microsoft Visual Studio

Es un IDE para sistemas operativos Windows. Soporta varios lenguajes de programación tales como Visual C++, Visual C#, ASP.NET y Visual Basic .NET, permite a los desarrolladores crear sistemas, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET. Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles.

[6] GNU Lesser General Public License.

Entre sus más destacables características, se encuentran la capacidad para utilizar múltiples monitores, así como la posibilidad de desacoplar las ventanas de su sitio original y acoplarlas en otros sitios de la interfaz de trabajo. La sincronización de hilos, tecnologías de clientes inteligentes, acceso a datos y muchas otras funcionalidades novedosas. El manejo de los errores se hace muchísimo más fácil y eficiente puesto que existen opciones de depuración más enriquecidas que brindan la información sobre los errores de forma intuitiva.

Selección del entorno de desarrollo integrado: *Qt Creator*

Después de realizar un estudio de los IDEs expuestos anteriormente se selecciona el Qt Creator como el más apropiado debido a la selección anterior del framework de desarrollo Qt. Este IDE fue creado fundamentalmente para desarrollar aplicaciones utilizando Qt lo cual posibilita una integración y una comunicación muy eficiente y cómoda entre la pareja framework – IDE.

Además de lo anterior, Qt Creator:

- Ofrece una gran documentación y una avanzada ayuda con ejemplos incluidos.
- Es multiplataforma, este hecho facilita su uso en sistemas operativos no privativos como Linux, siendo esta una gran ventaja debido a que continuamente se desarrollan nuevas funcionalidades por parte del grupo de desarrolladores.

2.7. Biblioteca

Una biblioteca es una colección o conjunto de subprogramas usados para desarrollar software. En general, las bibliotecas no son ejecutables, pero sí pueden ser usadas por ejecutables que las necesitan para poder funcionar correctamente. La mayoría de los sistemas operativos proveen bibliotecas que implementan la mayoría de los servicios del sistema. Dichas bibliotecas contienen comodidades que las aplicaciones modernas esperan que un sistema operativo provea.

2.7.1. Libvlc

Es una biblioteca gráfica desarrollada por Video LAN (VLC) bajo los términos de la GNU General Public License en su versión 2. Los desarrolladores pueden utilizar la biblioteca con el objetivo de explotar las complejas funcionalidades implementadas por VLC, brinda la posibilidad de reproducir contenido audiovisual y hacer streaming del mismo.

Libvlc se distribuye de forma compartida, lo que permite al desarrollador de aplicaciones acceder a las funcionalidades del reproductor y usarlas a su conveniencia. Además presenta las siguientes características.

- Portabilidad con un gran número de sistemas operativos, tales como Microsoft Windows, GNU Linux, MacOS, BeOS y FreeBSD.
- Soporte para múltiples tipos de salida de video, tales como DirectX, X11.
- Escrita en C, proporcionando un alto rendimiento necesario para el trabajo con los archivos multimedia.
- Cuenta con soporte nativo para operaciones multiproceso (TROJAHN 2010).

El uso de la biblioteca Libvlc se justifica a partir de su utilización en el tratamiento del material audiovisual pues proporciona varias funcionalidades, ya implementadas, que son utilizadas para la reproducción y el streaming del material en cuestión. Lo anterior proporciona un ahorro considerable en el tiempo de desarrollo de la propuesta pues parte del proceso de transmisión es apoyado por esta librería.

2.8. Conclusiones parciales

En este capítulo se han descrito las herramientas, tecnologías y metodologías de desarrollo más apropiadas para la implementación de la aplicación. Por los resultados obtenidos el lenguaje de programación adecuado es el C++, utilizando como metodología de desarrollo RUP y a su vez como lenguaje de modelado UML. Se define, además, la utilización del IDE de desarrollo QT Creator y Visual Paradigm como herramienta CASE para documentar y modelar los artefactos generados en cada etapa.

Además de lo anterior, se concluye que las herramientas a utilizar se corresponden con las políticas de software libre propuestas por la universidad donde se desarrolla esta investigación propiciando así la soberanía tecnológica.

CAPÍTULO 3
PRESENTACIÓN DE LA SOLUCIÓN PROPUESTA

3.1. Introducción

En el presente capítulo se expone la propuesta de solución en términos de ingeniería de software; se realiza el proceso de modelación del sistema y se especifican los requisitos funcionales y no funcionales que se consideran necesarios. Por otra parte, se lleva a cabo un estudio de los patrones de diseño para lograr un entendimiento de cómo se deben desarrollar los diagramas de clase del diseño y se describen los principales casos de uso que muestran la solución.

3.2. Modelo de Dominio

Un modelo del dominio es un modelo conceptual de un sistema, que identifica las relaciones entre todas las entidades importantes dentro del sistema e identifica generalmente sus métodos y cualidades. Puede ser tomado como el punto de partida para el diseño del sistema. El modelo del dominio se crea para documentar los conceptos dominantes del sistema, representa los conceptos u objetos del mundo real, significativos para un problema o área de interés. Ayuda además a los usuarios, clientes, desarrolladores y otros interesados a utilizar un vocabulario común (JACOBSON 2000b).

3.2.1. Diagrama de clases del dominio

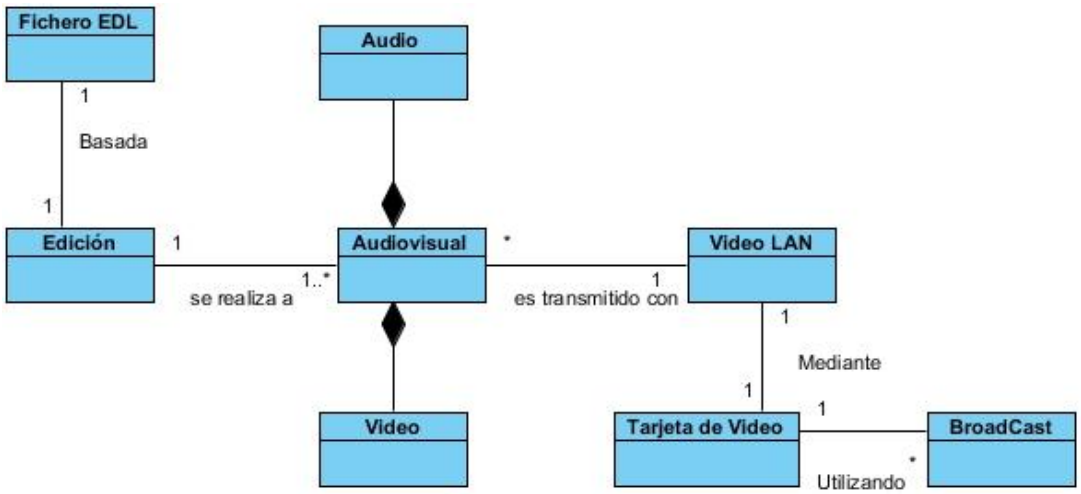

Figura 2: Diagrama de clases del dominio.

3.2.2. Conceptos principales del Modelo de Dominio

Fichero EDL: archivo que contiene las escenas deseadas y las ediciones que se aplicarán a la media o al grupo de medias correspondientes a un directorio. Se relaciona con el concepto edición al estar este último basado en un fichero EDL.

Edición: proceso mediante el cual se elabora un trabajo a partir de las imágenes obtenidas de un soporte de video, asociadas a los ficheros EDL que contienen dichas ediciones.

Audiovisual: es información que se transmite de forma visual y/o sonora, constituida por audio, imágenes, animaciones, texto y/o video.

Audio: archivo de sonido que forma parte de un audiovisual.

Video: archivo constituido por una secuencia de imágenes que conforman una escena en movimiento y forma parte de un audiovisual.

Video LAN: software que permite la reproducción/transmisión del audiovisual o el directorio de medias asociadas a un fichero EDL, previamente solicitado por un usuario.

Tarjeta de video: hardware que permite la transmisión del audiovisual o el grupo de audiovisuales.

Broadcast: tipo de transmisión en vivo donde se difunde de forma simultánea el paquete de medias o audiovisual desde un emisor para una gran cantidad de clientes que realizan una misma petición.

3.3. Requisitos

Los requisitos de software son las necesidades de los clientes, los servicios que los usuarios desean que proporcione el sistema de desarrollo y las restricciones en las que debe operar. Los requisitos se dividen en funcionales y no funcionales muestran las capacidades o condiciones que el sistema debe cumplir y las propiedades o cualidades que el producto debe tener, los cuales en la fase de construcción deben ser posibles de probar o verificar (PRESSMAN 2005).

3.3.1. Requisitos Funcionales (RF)

Los RF son capacidades o condiciones que el sistema debe cumplir y definen las funciones que será capaz de realizar. Para la aplicación propuesta se definen los siguientes requisitos funcionales:

RF1. Cargar fichero EDL

Descripción: La aplicación debe brindar al usuario la posibilidad de reconocer en un directorio un fichero EDL, lo cual hará mediante la comparación con el tipo de extensión de dicho archivo (*.edl*).
Entradas: No tiene.
Salidas: El sistema mostrará un mensaje notificando que el fichero EDL fue reconocido y cargado correctamente.

RF2. Reconocer ficheros de tipo media desde un directorio

Descripción: El sistema debe brindar la posibilidad de incorporar a una lista de reproducción el elemento o los elementos de tipo media con las extensiones reconocibles por el reproductor VLC (avi, mpg, mp4, wav, mp3, etc.) correspondientes al fichero EDL cargado previamente por el usuario.
Entradas: No tiene.
Salidas: El sistema mostrará en una lista los archivos de tipo media asociados al fichero EDL.

RF3. Interpretar fichero EDL

Descripción: El sistema debe ser capaz de leer el fichero con extensión EDL con el objetivo de reconocer el/los nombre/s de la/s media/s asociada/s a las ediciones que incluye el mismo. El software será capaz de reconocer mientras lee el fichero EDL determinados eventos de edición de medias.
Entradas: No tiene.
Salidas: No tiene.

RF4. Interpretar una transición

El sistema debe ser capaz de reconocer una transición en el fichero EDL correspondiente a las medias asociadas a él. Las transiciones a reconocer son las siguientes: aparecer, desaparecer, desaparecer-aparecer, barrido de izquierda a derecha, barrido de derecha a izquierda, barrido de arriba a abajo, barrido de abajo a arriba, barrido de esquina superior izquierda a inferior derecha, barrido de esquina superior derecha a inferior izquierda, barrido de esquina inferior izquierda a superior derecha y barrido de esquina inferior derecha a superior izquierda.

RF5. Interpretar efectos

El sistema debe ser capaz de reconocer algunos efectos en el fichero EDL correspondientes a las medias asociadas a él. se reconocerán los siguientes efectos: convertir a blanco y negro, modificar brillo, convertir a escala de grises, invertir colores, suprimir azul, suprimir rojo, suprimir verde y modificar color.

RF6. Interpretar corte de fragmentos

La aplicación debe ser capaz de reconocer en el fichero EDL cuándo se realiza una omisión de un fragmento de medias.

Entradas: No tiene.

Salidas: No tiene.

RF7. Reproducir el paquete de medias

Descripción: El sistema debe ser capaz de reproducir el audiovisual con el conjunto de acciones definidas en el fichero EDL.

Entradas: No tiene.

Salidas: El sistema permitirá la reproducción de la/s media/s presente/s en la lista de reproducción asociadas al fichero *EDL*.

RF8. Mostrar los nombres de las medias presentes en el EDL y que no están en el directorio del fichero edl

Descripción: El sistema debe ser capaz de listar las medias asociadas al EDL que no se encuentren en el directorio especificado en el fichero EDL.

Entradas: No tiene.

Salidas: No tiene.

3.3.2. Requisitos No Funcionales (RNF)

Los RNF son definidos formalmente como las propiedades o cualidades que el producto debe tener, características que lo hacen más agradable, rápido o confiable. Luego de analizado el entorno de desarrollo y futuro despliegue del sistema, para el desarrollo de la aplicación se definen los siguientes RNF:

Usabilidad

Facilidad de Operación: El sistema tendrá la característica de ser operable por usuarios sin grandes conocimientos informáticos gracias a la sencillez de su interfaz, todas sus funcionalidades deben ser accesibles intuitivamente.

Soporte

El sistema deberá permitir la modificación de módulos en el momento que así lo requiera, asegurando lograr mejores prestaciones y su extensibilidad.

Requisitos de software

La computadora donde estará alojada la aplicación deberá cumplir con las siguientes características:

- Poseer instalada la librería Libvlc 1.1.4 o superior.
- Framework de desarrollo Qt, Licencia Qt GNU LGPL v. 2.1 o superior.
- Sistema Operativo GNU/Linux versión 10.10 o superior.

Requisitos de hardware

La computadora donde estará alojada la aplicación deberá cumplir con las siguientes características:

- Procesador Dual-Core Xeon 2.33 GHz o superior.
- Memoria RAM de un 1 Gb o superior.
- Disco Duro de 80 Gb o superior.
- Tarjeta de Red Ethernet 10/100 Mbps.
- Tarjeta de Video Exportadora ATI Radeon X300.

Requisitos Legales, de Derecho de Autor y otros

La herramienta para la transmisión de audiovisuales es propiedad exclusiva de la UCI.

3.4. Determinación y justificación de los actores del sistema

Un actor es un rol que cumple un usuario, puede intercambiar información o puede ser un recipiente pasivo de información y representa a un ser humano, a un software o a una máquina que interactúa con el sistema. Durante el desarrollo de la aplicación se definió un único actor, el mismo se describe a continuación (PRESSMAN 2005).

Actor	*Descripción*
Usuario	Persona con acceso al sistema que realizará la solicitud de visualizar los audiovisuales correspondientes a un fichero EDL.

Tabla 1: Descripción del Actor del Sistema.

3.5. Descripción del Sistema. Modelo de Casos de Uso del sistema

El modelo de caso de uso ayuda al cliente, a los usuarios y a los desarrolladores a llegar a un acuerdo sobre cómo utilizar el sistema. La mayoría de los sistemas tienen muchos tipos de usuarios. Cada usuario se representa mediante un actor. Los actores utilizan el sistema al interactuar con los casos de uso (PRESSMAN 2005).

La descripción del modelo de casos de uso del sistema se realiza haciendo uso de las ventajas que brinda el lenguaje de modelado UML, se formulan las funcionalidades y representación mediante un diagrama del sistema, para ello es de vital importancia definir los actores y los casos de uso que representarán las responsabilidades del mismo.

3.5.1. Diagrama de Casos de Uso del sistema propuesto

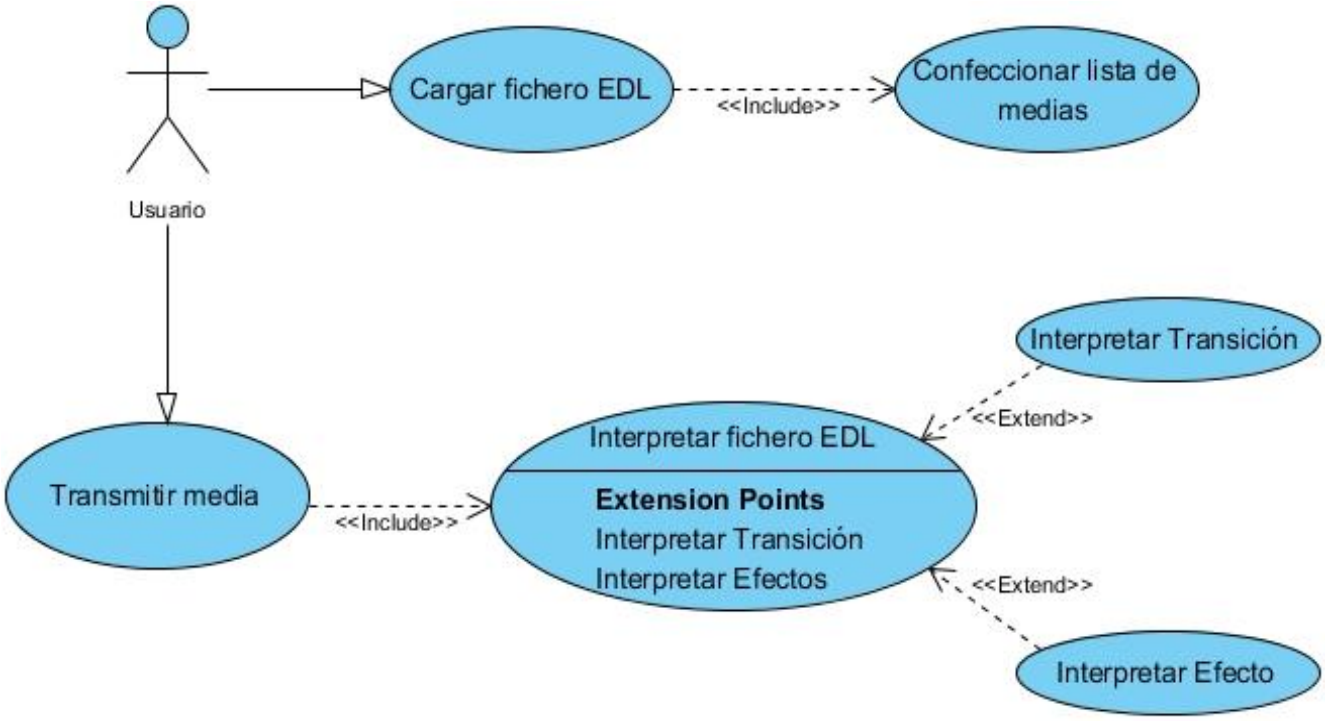

Figura 3: Diagrama de casos de uso del sistema.

3.5.2. Descripción textual de los Casos de Uso

Descripción textual del caso de uso: Cargar fichero EDL.

Caso de Uso	Cargar fichero EDL.
Actores	Usuario.
Resumen	El caso de uso se inicia cuando el usuario busca la ubicación del fichero EDL, lo selecciona y al cargarlo se listan las medias asociadas al mismo, finalizando así el caso de uso.
Prioridad	Alta.
Referencias	RF1.
Precondiciones	Ninguna.

Flujo Normal de eventos		
	Actor	**Sistema**
1.	El caso de uso se inicia cuando el usuario selecciona la opción “Cargar fichero EDL”.	1.1 El sistema muestra una ventana brindando la posibilidad de escoger el directorio donde se encuentra el fichero EDL.
2.	El usuario selecciona un directorio.	2.1 El sistema muestra el fichero EDL existente en el directorio seleccionado.
3.	El usuario selecciona el fichero EDL y accede a la opción “Aceptar”.	3.1 El sistema muestra un mensaje de operación satisfactoria. 3.2 Se ejecuta el caso de uso “Confeccionar lista de medias”. 3.3 El sistema retorna a la interfaz principal mostrando la lista de medias obtenida en el paso anterior y finaliza así el caso de uso.
Flujos alternos		
Actor		**Sistema**
		2.1. En caso de no existir ficheros EDL en el directorio seleccionado el sistema muestra el directorio vacío y finaliza así el caso de uso.
3.	El usuario accede a la opción “Cancelar”.	3.1. En caso de que el usuario seleccione la opción “Cancelar” el sistema regresa a la pantalla inicial finalizando así el caso de uso.
		3.3. (a) En caso de que la lista de medias esté vacía el sistema envía un mensaje de error notificando este hecho.
		3.3. (b) En caso de que la lista de medias no esté completa de acuerdo a lo especificado en el fichero EDL el sistema envía un mensaje notificando este hecho.
Postcondiciones	Un fichero EDL es cargado.	

Tabla 2: Descripción textual del caso de uso Cargar fichero EDL.

Debido a lo extenso que resulta la descripción textual de todos los Casos de Uso, las demás descripciones textuales no se realizarán en este apartado del documento y para ello se propone consultar el ***Anexo 1***.

3.6. Elementos de la Arquitectura del Software

Una arquitectura adecuada es la pieza clave para lograr tanto los requerimientos funcionales como no funcionales de un sistema. La arquitectura de software, ha surgido como una disciplina de gran importancia dentro de la ingeniería de software. A continuación se brindan algunas aproximaciones teóricas asociadas a este importante elemento dentro del proceso de desarrollo de software.

Según la IEEE la arquitectura del software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos, el contexto en el que se implantarán, y los principios que orientan su diseño y evolución (ASSOCIATION 2000). Otra de las definiciones más aceptadas de arquitectura de software, la cual la presente investigación considera como más apropiada, es la enunciada por Booch, Jacobson y Rumbaugh en su libro “El Proceso Unificado de Desarrollo de Software”, en el cual se señala lo siguiente:

Arquitectura es la estructura de los componentes más significativos de un sistema interactuando a través de interfaces con otros componentes conformados por componentes sucesivamente pequeños e interfaces (JACOBSON 2000a).

La arquitectura de software es muy importante en el proceso de desarrollo del software. Facilita la comunicación entre todas las partes interesadas en el desarrollo de un sistema basado en computadora; destaca decisiones tempranas de diseño con un profundo impacto en todo el trabajo de ingeniería del software que sigue, y es tan importante en el éxito final del sistema como una entidad operacional; constituye un modelo relativamente pequeño e intelectualmente comprensible de cómo está estructurado el sistema y, además, de cómo trabajan juntos sus componentes (PRESSMAN 2005).

3.6.1. Estilos y Patrones

Los estilos arquitectónicos son un conjunto de reglas de diseño que identifican las clases de componentes y conectores que se pueden utilizar para componer un sistema o subsistema, junto con las restricciones locales o globales en que la composición se lleva a cabo (REYNOSO 2004).

En esta investigación se utilizará el estilo arquitectónico denominado "*llamada y retorno*". Esta familia de estilos enfatiza la modificabilidad y la escalabilidad. Son los estilos más generalizados en sistemas en gran escala. Miembros de la familia son las arquitecturas de programa principal y subrutina, los sistemas basados en llamadas a procedimientos remotos, los sistemas orientados a objetos y los sistemas jerárquicos en capas. Para la construcción del sistema propuesto se decidió utilizar el *patrón arquitectura en capas*, definido según Garlan y Shaw como una organización jerárquica tal que cada capa proporciona servicios a la capa inmediata superior y se sirve de las prestaciones que le brinda la inmediata inferior (REYNOSO 2004).

Algunas de las principales ventajas de la arquitectura en capas radica en que soporta un diseño basado en niveles de abstracción crecientes, lo cual a su vez permite a los programadores la participación de un problema complejo en una secuencia de pasos incrementales, admite optimizaciones y refinamientos y proporciona amplia reutilización, lo que se convierte en uno de los fuertes de esta arquitectura. En la presente investigación se pondrá en práctica la *arquitectura en tres capas* que permite dividir la aplicación en tres partes fundamentales:

La *capa de presentación*: funciona como la interfaz con el usuario. Resuelve la presentación de datos al usuario. Se encarga entonces de "dibujar" las pantallas de la aplicación al usuario y tomar los eventos que el cliente genere (por ejemplo, el hacer click en un botón).

La *capa de negocio*: resuelve la lógica de la aplicación. Contiene los algoritmos, validaciones y coordinación necesaria para resolver la problemática. Los elementos fundamentales de esta capa son los objetos de dominio. Estos objetos representan los objetos principales del negocio. Son simples objetos que sólo contienen los datos que representan (por ejemplo, un Cliente, una Factura, una Dirección, un Producto).

La *capa de datos*: almacena los datos del sistema y de los usuarios. Su función es lograr la persistencia de los datos almacenando y devolviendo los datos a la capa de negocio. Para esto es necesario en algunos casos, contar con procedimientos de almacenado y otras funciones específicas del acceso a los datos, sobre todo cuando estos se tornan complejos (DOSIDEAS 2011).

Los *patrones de diseño*, constituyen soluciones estándar que brindan respuesta a un problema común durante el diseño de un software. Una vez que se ha desarrollado el modelo de análisis, el diseñador puede examinar una representación detallada del problema que debe resolver y las restricciones que impone el problema. Estos permiten establecer una gran comunicación entre los diseñadores,

facilitando además el aprendizaje del programador inexperto y logrando crear parejas problema-solución. Cada patrón sugiere además numerosas ventajas, dentro de ellas la reutilización de código y el permitir la realización de un diseño apto para el cambio (PRESSMAN 2001).

En esta investigación se utilizan fundamentalmente los patrones de diseño GRASP (*General Responsibility Assignment Software Patterns*, Patrones generales de asignación de responsabilidad de software), específicamente los siguientes:

Experto: Se utilizará este patrón para la asignación de responsabilidades indicando que la clase que cuenta con la información necesaria para cumplir una tarea debe ser la responsable de ejecutar la misma, proporcionando que los objetos exploten su propia información para cumplir con sus funcionalidades, lo cual conservará el encapsulamiento.

Bajo Acoplamiento: Se utiliza para establecer una escasa dependencia entre las clases, reduciendo el impacto de posibles cambios en el sistema.

Alta Cohesión: Se ocupa de que las clases del diseño realicen las funcionalidades necesarias para cumplir con las tareas que tienen definidas. Plantea la contribución entre clases para realizar tareas de elevada complejidad. Ante una tarea de gran magnitud los objetos se comparten el trabajo colaborando entre ellos, evitando así la existencia de clases saturadas de métodos redundantes (LARMAN 1999a).

3.7. Conclusiones parciales

Los autores concluyen:

1. Con la modelación del dominio y el sistema se logra un entendimiento común entre el equipo de desarrollo de manera que las etapas sucesivas del desarrollo de la solución pueden ser comprendidas y ejecutadas sin dificultad por los demás miembros.
2. La documentación generada producto de la realización de las actividades anteriores asegura la continuidad y la ampliación de la solución en algún momento posterior a esta investigación.
3. Los requisitos funcionales y no funcionales ubican al equipo de desarrollo en los objetivos de la solución y proporcionan una guía para su desarrollo y posterior validación.
4. La utilización de patrones de diseño permite ahorrar cantidades considerables de tiempo en el desarrollo, propiciando facilidades para la comprensión, mantención y extensión de la solución.

CAPÍTULO 4
IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN

4.1. Introducción

En el presente capítulo se describe la solución en términos de diagramas de clases del diseño, se enuncian los principales aspectos tenidos en cuenta en el diseño gráfico de la aplicación (apariencia), se presenta todo lo relacionado con las pruebas realizadas al sistema, además, se describen el diagrama de componentes y el de despliegue.

4.2. Modelo de Diseño

El modelo del diseño describe la realización física de los casos de uso y se corresponde directamente con los elementos físicos del ambiente de implementación. Consiste en colaboraciones de clases, que pueden ser agregadas en paquetes y/o subsistemas (JACOBSON 2000a).

4.2.1. Diagrama de Clases de Diseño

Los diagramas de clases de diseño son lo más utilizados en el modelado de sistemas orientado a objetos. Un diagrama de clases muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Estos son importantes no solo para visualizar, especificar y documentar modelos estructurales, sino también para construir sistemas ejecutables, aplicando ingeniería directa e inversa (PRESSMAN 2005).

A continuación se presentará la distribución de la aplicación de acuerdo al Patrón Arquitectónico seleccionado Arquitectura en Capas, específicamente arquitectura en tres capas, donde cada una de ellas se relaciona estrictamente con la del nivel inmediato inferior, logrando la comunicación y organización propias de la implementación del Patrón.

Por lo complejo de la visualización del diagrama se representan solamente las relaciones entre las clases presentes. La implementación exacta de cada clase, incluyendo atributos y responsabilidades, puede ser consultada en el ***Anexo 2***.

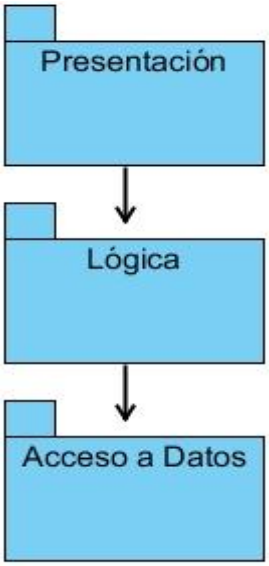

Figura 4: Implementación del Patrón Arquitectónico "En tres capas".

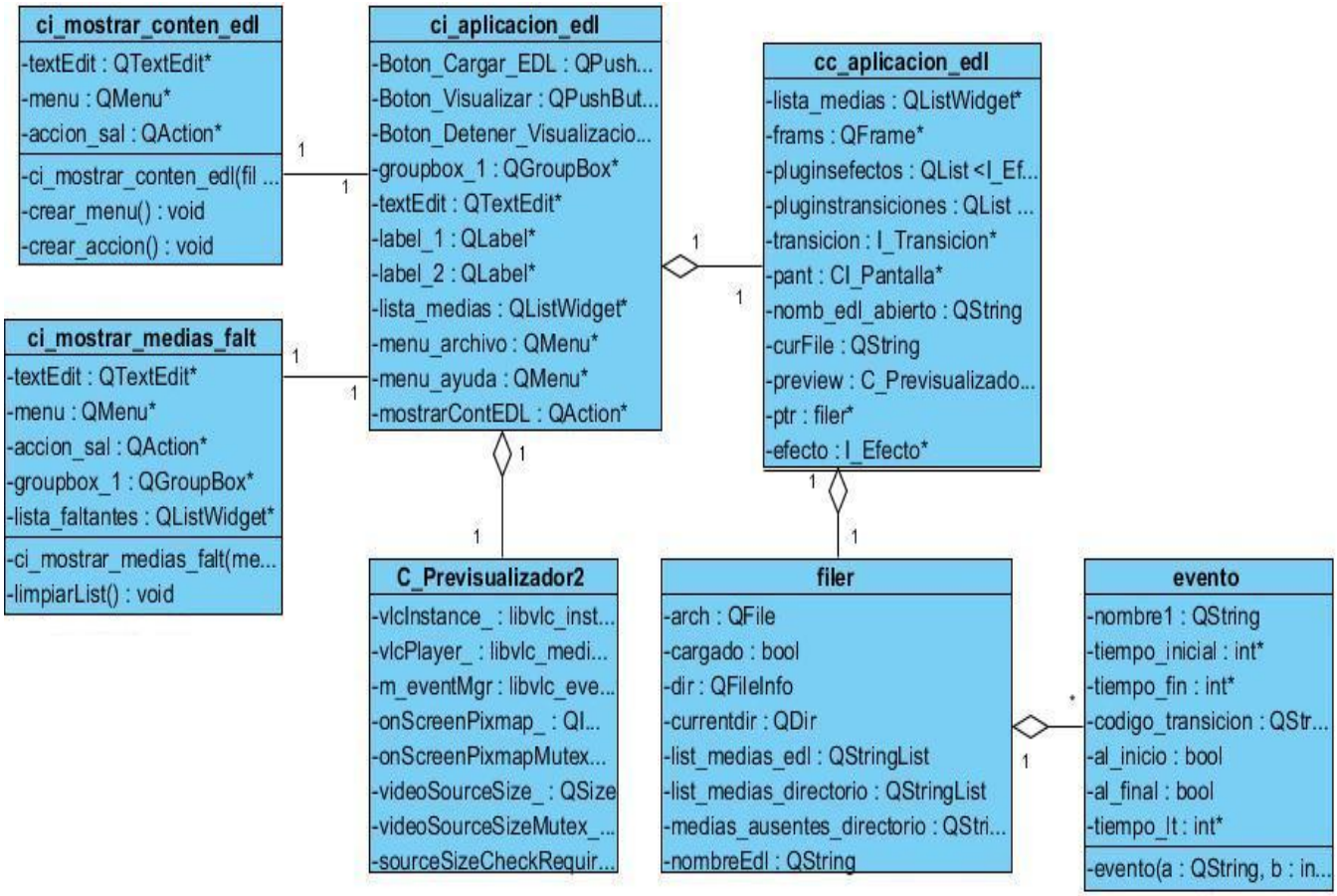

Figura 5: Diagrama de Clases del Diseño.

4.3. Principios del diseño del sistema

4.3.1. Estándares de la Interfaz

La herramienta informática en desarrollo contará con una interfaz visual amigable, intuitiva y de fácil utilización, conteniendo colores sobrios y propios de una herramienta profesional de edición de videos. Se utilizarán preferentemente la gama de colores grises.

El espacio de la aplicación se divide en tres partes fundamentales, un área de Menú, donde se ubican las principales funcionalidades de la aplicación en forma de menús desplegables, esta área está ubicada en la parte superior horizontalmente, otra área dedicada a la visualización de información, fundamentalmente asociadas a la carga del fichero EDL, esta área se encuentra en la parte izquierda verticalmente. Finalmente, el área de visualización de la edición que está ubicada en la parte centro-derecha, cubriendo la mayor porción de la ventana.

4.4. Modelo de Implementación

El modelo de implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes, como ficheros de código fuente, ejecutables, entre otros. El modelo de implementación describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros (GRADY BOOCH 2004).

4.4.1. Diagrama de Componentes

Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas. Un diagrama de componentes representa las dependencias entre componentes software, incluyendo componentes de código fuente, componentes del código binario, y componentes ejecutables (JACOBSON *et al.* 2004).

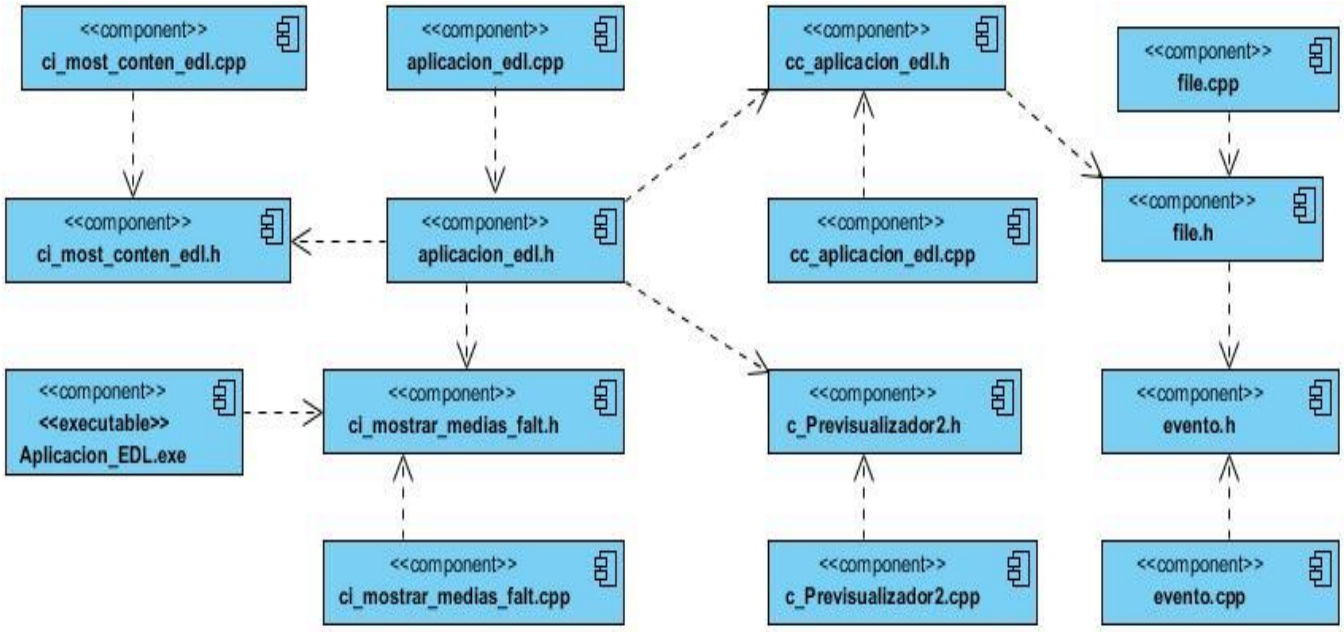

Figura 6: Diagrama de Componentes.

4.5. El proceso de pruebas

En el desarrollo de cualquier software las pruebas tienen gran importancia, debido a que son un proceso que permite verificar y revelar la calidad del mismo. Cada prueba tiene su estrategia y propósito. Pueden ser utilizadas para identificar posibles fallos relacionados con la implementación, calidad, o usabilidad del sistema. La ausencia de defectos no puede ser asegurada a través de las pruebas, sino que solo se puede demostrar que existen errores en el software.

4.5.1. Definición de pruebas

Las pruebas son un conjunto de actividades que se planean con anticipación y se realizan de manera sistemática. Por tanto, se debe definir una plantilla para las pruebas del Software (un conjunto de pasos en el que se puedan incluir técnicas y métodos específicos del diseño de caso de prueba) (PRESSMAN 2001).

Otro concepto de prueba de software la define como “una actividad en la cual un sistema o uno de sus componentes se ejecuta en circunstancias previamente especificadas, los resultados se observan y registran y se realiza una evaluación de algún aspecto” (BERCIAL 2006).

4.5.2. Pruebas de caja negra

Las *pruebas de caja negra,* también denominadas, *pruebas de comportamiento,* se concentran en los requisitos funcionales del software. Es decir, permiten al ingeniero de software derivar conjuntos de condiciones de entrada que ejercitarán por completo todos los requisitos funcionales de un programa.

La prueba de caja negra no es una opción frente a las técnicas de caja blanca. Es, en cambio, un enfoque complementario que tiene probabilidades de descubrir una clase diferente de errores de los que descubrirían los métodos de caja blanca. Este método de prueba es realizado a nivel de sistema es decir no se tiene ninguna relación con el código del producto (PRESSMAN 2005).

Varias razones propiciaron la selección del paradigma de las pruebas de tipo caja negra; entre otras, porque pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta. Además de esto, estas pruebas permiten encontrar:

- Funciones que estén 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.

4.5.3. Diseños de Casos de Prueba

El diseño de las pruebas se basa en la creación de casos de prueba cuya ejecución permitirá observar posibles síntomas de defectos. Se puede definir un caso de prueba como “el conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para un objetivo particular como, por ejemplo, ejercitar un camino concreto de un programa o verificar el cumplimiento de un determinado requisito” (FERNÁNDEZ 2001).

Para la ejecución del proceso de pruebas de la solución, se desarrollaron, por cada Caso de Uso, las secciones a probar y la matriz de datos a utilizar en las pruebas, constituyendo estos dos artefactos los Diseños de los Casos de Prueba.

Diseños del Caso de Prueba Cargar Fichero EDL.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC1: Cargar fichero EDL	EC1.1: Cargar fichero EDL satisfactoriamente	El usuario selecciona la opción “Cargar fichero EDL”, el sistema muestra una ventana con la posibilidad de escoger el directorio donde se encuentra el fichero. El usuario selecciona un directorio y el sistema muestra el fichero existente, el usuario selecciona el fichero EDL y accede a la opción “Aceptar”. El sistema muestra un mensaje de operación satisfactoria. Se ejecuta el Caso de Uso “Confeccionar lista de medias” y se realiza el EC 1.4 o EC 1.5. El sistema retorna a la interfaz principal mostrando la lista de medias obtenida en el paso anterior.	Clic en el botón “Cargar fichero EDL”/ Clic en el botón “Aceptar”.
	EC1.2: Directorio vacío	En caso de no existir ficheros EDL en el directorio seleccionado el sistema muestra el directorio vacío y finaliza así el caso de uso.	Clic en el botón “Cargar fichero EDL”.
	EC1.3: Cancelar la carga del fichero EDL	En caso de que el usuario seleccione la opción “Cancelar” el sistema regresa a la pantalla inicial finalizando así el caso de uso.	Clic en el botón “Cargar fichero EDL”/ Clic en el botón “Cancelar”.
	EC1.4: Lista vacía de medias asociadas	En caso de que la lista de medias esté vacía el sistema envía un mensaje de error notificando este hecho.	Clic en el botón “Cargar fichero EDL”.
	EC1.5: Lista incompleta de medias asociadas	En caso de que la lista de medias no esté completa de acuerdo a lo especificado en el fichero EDL el sistema envía un mensaje notificando este hecho.	Clic en el botón “Cargar fichero EDL”/ Clic en el botón “Aceptar”.

Tabla 3: Secciones a probar del Caso de Uso Cargar fichero EDL.

ID del escenario	Escenario	Respuesta del Sistema	Resultado de la Prueba
EC1.1:	Cargar fichero EDL satisfactoriamente	El sistema muestra una ventana con la posibilidad de escoger el directorio donde se encuentra el fichero. El sistema muestra el fichero existente y muestra un mensaje de operación satisfactoria.	
EC1.2:	Directorio vacío	El sistema muestra el directorio vacío.	
EC1.3:	Cancelar la carga del fichero EDL	El sistema regresa a la pantalla inicial.	
EC1.4:	Lista vacía de medias asociados	El sistema envía un mensaje de error notificando este hecho.	
EC1.5:	Lista incompleta de medias asociados	El sistema envía un mensaje notificando este hecho.	

Tabla 4: Matriz de Datos del Caso de Uso Cargar Fichero EDL.

Los restantes Diseños de Casos de Prueba de la solución se recogen en el ***Anexo 3.***

4.6. Conclusiones parciales

1. El diseño de la herramienta fue realizado siguiendo buenas prácticas de diseño a partir de la implementación de varios patrones y estilos arquitectónicos.
2. La implementación de la arquitectura en capas permitirá aumentar la escalabilidad, disponibilidad, seguridad e integración de la solución propuesta.
3. La herramienta informática tendrá un diseño sencillo, con una distribución estándar del contenido, aumentando así los niveles de usabilidad y adaptación del usuario final.
4. Las pruebas que se definen como parte del proceso de desarrollo se consideran adecuadas y muy necesarias para lograr un producto libre de defectos.

CONCLUSIONES GENERALES

Una vez completado el proceso investigativo los autores concluyen lo siguiente:

1. La revisión y caracterización del estado del arte del tema de la investigación permitió afirmar que las soluciones que existen hoy, que de alguna manera tributan a la investigación, no resuelven la problemática planteada, expresándose la necesidad del desarrollo de la propuesta.
2. Las limitaciones en la definición de un componente o biblioteca para la transmisión del flujo streaming propició la utilización de la tarjeta de video como medio para alcanzar la difusión de los contenidos audiovisuales.
3. Las herramientas, tecnologías y lenguajes que se proponen, en todos los casos, se corresponden con las políticas de soberanía tecnológica que impulsa la universidad y el país.
4. Todos los requisitos funcionales que se definieron fueron debidamente implementados, igualmente, se incluyeron en la propuesta las exigencias de todos los requisitos no funcionales detectados.
5. La solución que se presenta es multiplataforma, lo que amplía las posibilidades de utilización y la gama de usuarios de la misma.
6. Los artefactos generados durante el proceso de desarrollo del software permitirán continuar escalando la solución en el futuro.
7. La utilización de estilos y patrones promueve buenas prácticas en el desarrollo de la solución al proporcionar uniformidad en la implementación, siendo más entendible y escalable en el tiempo. Por otra parte, tanto estilos como patrones, permiten ahorrar grandes cantidades de tiempo en la implementación.
8. Los diseños de casos de prueba desarrollados, como parte de las pruebas de caja negra, permitieron validar los requisitos de la aplicación con las funcionalidades implementadas.
9. La utilización de la herramienta producto de esta investigación en el Departamento de Señales Digitales garantiza la reducción de costos por conceptos de tiempo, almacenamiento y gestión de cambios en el proceso de transmisión de audiovisuales.

RECOMENDACIONES

Los autores recomiendan:

1. Implementar una biblioteca que posibilite la creación del flujo de streaming a partir de las listas de decisión de edición para lograr una transmisión bajo demanda.
2. Continuar desarrollando la solución implementada agregándole efectos asociados a las ediciones de audio.
3. Mejorar el diseño de la interfaz de la aplicación que posibilite un aumento en la usabilidad y la experiencia del usuario.
4. Incluir como parte de la solución la posibilidad de realizar ediciones asociadas al audio en materiales audiovisuales.
5. Realizar las pruebas al sistema de acuerdo a partir de los Diseños de Casos de Prueba especificados en esta investigación.

REFERENCIAS BIBLIOGRÁFICAS

1. ABC, D. *Definición de transmisión*, 2011. [2011]. Disponible en: http://www.definicionabc.com/general/transmision.php41
2. ALEMAN, Y. R. S. Y. G. N. *Transferencia Tecnológica del Portal Inter-Nos* Señales Digitales. La Habana, Universidad de las Ciencias Informáticas, 2007. 154. p.
3. ALFREDO, R. *El proceso de investigación científica*. La Habana, Editorial Universitaria, 2011. 110 p. 978-959-16-1307-3
4. ÁLVAREZ, I. *Curso de Investigación Científica*. Santa Clara Cuba, 1997. p.
5. ANTÓNIMOS, D. D. S. Y. *Definición de producir*, 2011. [2011]. Disponible en: http://www.wordreference.com/sinonimos/producir
6. APPLE. *Final Cut Pro a fondo*, 2012. [2012]. Disponible en: http://www.apple.com/es/finalcutpro/all-features/#video-editing
7. ASSOCIATION, I. S. *RECOMMENDED PRACTICE FOR ARCHITECTURAL DESCRIPTION OF SOFTWARE-INTENSIVE SYSTEMS* 2000. [2008]. Disponible en: www.standards.ieee.org/reading/ieee/std_public/description/se/1471-2000_desc.html
8. BERCIAL, L. S. Y. P. J. L. *Universidad Europea de Madrid: Taller sobre Pruebas en Ingeniería del Software.*, 2006. 7.
9. BERROTERÁN, C. *Streaming Stored*, 2011. 10.
10. CERDA, F. *Netbeans 6.5 único IDE que necesitas*, 2009.
11. CINE, E. *La Edición*, 2006. [2011]. Disponible en: http://ecine.blogcindario.com/2006/11/00010-la-edicion.html
12. COMPUTACIÓN, S. I. E. *Edición No-Lineal de Audio y Video*, 2001. [2011]. Disponible en: http://www.teclayraton.net/nolineal/edicion.html
13. COMUNICA, E. T. C. *Edición de Video*, 2009.
14. DEFINICIÓN.DE. *Definición de audiovisual*, 2011a. [2011]. Disponible en: http://definicion.de/audiovisual/
15. ---. *Definición de producción*, 2011b. [2011]. Disponible en: http://definicion.de/produccion/
16. DIGITALES, D. D. S. *Sistema de gestión y transmisión de contenidos audiovisuales (SIAV)*. La Habana, UCI, 2012. 11.
17. DIGITALFOTORED. *Edición Lineal*, 2005. [2011]. Disponible en: http://www.digitalfotored.com/videodigital/tiposedicionvideo.htm
18. DLE. *Definición de audiovisual*, 2011. [2011]. Disponible en: http://www.wordreference.com/definicion/audiovisual
19. DOSIDEAS. *Arquitectura de 3 Capas*, 2011. [2012]. Disponible en: http://www.dosideas.com/cursos/mod/resource/view.php?id=10
20. ECURED. *Edición de Video*, 2011a. [2011]. Disponible en: http://www.ecured.cu/index.php/Edici%C3%B3n_de_v%C3%ADdeo
21. ---. *QT Creator*, 2011b. [2012]. Disponible en: http://www.ecured.cu/index.php/Qt_Creator
22. EGAS, V. E. V. *Portal web para el conservatorio superior nacional de música*. Quito, Escuela Politécnica Nacional, 2007. 137. p.
23. ESPAÑOLA, R. A. *Definición de transmitir*, 2010. [2011]. Disponible en: http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=transmitir
24. ESPARTA, U. N. *Sistemas de Transmisión*, 1997. [2011]. Disponible en: http://www.une.edu.ve/~jduran/disertaciones_unidad1.html
25. FERNÁNDEZ, P. L. Y. L. *Generación de casos de prueba a partir de especificaciones UML*, 2001. 11.
26. GARCÍA, D. C. *La edición en TV: diferencias entre los sistemas lineal y no lineal.*, 2009.

27. GILL. *Edit Decision list for media products distribution*. Otawa, 2007. 84.
28. GÓMEZ, A. V. *SERVICIO DE MEDIA STREAMING PARA LA WEB. PORTAL INTER-NOS. MÓDULOS: TELECLASES Y TV*. Señales Digitales. La Habana, Universidad de las Ciencias Informáticas, 2006. 90. p.
29. GRADY BOOCH, J. R. Y. I. *El Lenguaje Unificado de Modelado*. 2004. p.
30. GUTIÉRREZ, D. G. *Tutorial de Qt4 Designer y QDevelop*, 2008. 98.
31. JACOBSON, I. *EL PROCESO UNIFICADO DE DESARROLLO DE SOFTWARE*. 2000a. 435 p.
32. ---. Los flujos de trabajo fundamentales. en: *El Proceso Unificado de Desarrollo*. 2000b. 193.p.
33. JACOBSON, I.; J. RUMBAUGH*, et al. El Lenguaje Unificado de Modelado*. 2004. p. *Parte_3_Referencia_Enciclopedia_de_terminos_parte_1.pdf.*
34. LARMAN, C. Fase del Diseño. en: *UML y Patrones*
35. *Introducción al análisis y diseño orientado a objetos*. México, 1999a. 4: 126.p.
36. ---. *UML y Patrones*
37. *Introducción al análisis y diseño orientado a objetos*. México, 1999b. 536 p.
38. MADRID, U. C. I. D. *Edición Digital No Lineal*, 2011. 2.
39. MÁLAGA, U. D. *Microsoft Visual C++ 6.0*, 2010. [2012]. Disponible en: http://www.eumed.net/libros/2010c/759/Microsoft%20Visual%20C.htm
40. MEDIACLUB. *Etapas de Producción de Videos*, 2010. [2011]. Disponible en: http://www.mediaclub.com.mx/Etapas_Produccion_Video_Toluca_Metepec_Lerma.html
41. MENCHACA, N. *Universidad de Guadalajara: Curso de Video*, 2010.
42. MICROSOFT. *About multicast streaming*, 2007. [2011]. Disponible en: http://technet.microsoft.com/en-us/library/cc753242%28WS.10%29.aspx
43. MORÁN, J. M. F. *Framework para la capa de presentación de aplicaciones web*, 2009. 66.
44. NACIONAL, U. T. *Lenguaje de Programación*, 2012. [Disponible en: http://www.slideshare.net/reingsys/nuevas-tecnologas-reingsys-31309
45. NETBEANS. *¿Qué es NetBeans?*, 2012. [2012]. Disponible en: http://netbeans.org/index_es.html
46. PÉREZ, I. C. *METODOLOGIA DE DESARROLLO DEL SOFTWARE*, 2008. 8.
47. PRESSMAN, R. *Ingeniería del Software. Un enfoque práctico.* 5ta. 2001. p.
48. ---. *Ingeniería del Software: Un enfoque práctico*. Sexta edición. 2005. 927 p.
49. QUALITYNET. *Streaming de video*, 2006. [Disponible en: http://www.qualitynet.es/
50. RAE. *Definición de producir*, 2010. [2011]. Disponible en: http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=producir
51. REYNOSO, C. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft*, 2004. 73.
52. ROMERA, F. E. *Cultura audiovisual: Etapas de Realización*, 2011.
53. ROMERO, D. C. *Aplicaciones de las listas de decisión de edición*. Arquitectura de Computadores. Cataluña, Universidad Politécnica de Cataluña, 2010. 122. p.
54. SALESIANA, U. P. Etapas del proceso de producción. en: *Video Documental Cuencano*. Quito, 2008. 28.p.
55. SANCHEZ, M. A. M. *Metodologías de desarrollo de Software*, 2004.
56. STALLING, W. *Comunicaciones y redes de Computadoras*. 6ta edición. 2000. 751 p.
57. SUPERIORES, I. S. D. E. *Transmisión de datos*, 2010. [2011]. Disponible en: http://hcdsc.gov.ar/biblioteca/ISES/educacion/ciencias%20de%20la%20comunicacion/Transmision%20de%20datos.pdf
58. SWEETWATER. *Edit Decision List*, 2011. [2011]. Disponible en: http://www.sweetwater.com/insync/word.php?find=EDL
59. TECHNOLOGY, A. *Avid EDL Manager User's Guide*, 2001. 94.
60. TROJAHN, T. H. *UMA IMPLEMENTAÇÃO DO COMPONENTE MEDIA PROCESSING USANDO A*
61. *BIBLIOTECA LIBVLC PARA O MIDDLEWARE GINGA*, 2010. 4.
62. TUXINFO. *Qt Development Framework*, 2012.

63. UNION, I. T. *Definition of transmission*, 2009.
64. VALENCIA, U. P. D. *Transmisión de datos multimedia*, 2007.
65. VEGA, D. *Edición de video digital*, 2010.
66. VIRTUAL, E. E. *Transmisión en Tiempo Real*, 2010. [2011]. Disponible en: http://www.eumed.net/libros/2008a/348/Transmision%20en%20Tiempo%20Real.htm
67. VIZCAÍNO, A. *Trabajando con Visual Paradigm for UML*, 2000.
68. WEB, D. *¿Qué es streaming?*, 2001. [2011]. Disponible en: http://www.desarrolloweb.com/articulos/482.php

ANEXO 1: DESCRIPCIÓN TEXTUAL DE LOS CASOS DE USO DEL SISTEMA

Descripción textual del caso de uso: Confeccionar lista de medias.

Caso de Uso	Confeccionar lista de medias.
Actores	
Resumen	El caso de uso se inicia cuando se ejecuta el caso de uso “Cargar fichero EDL” y finaliza cuando se listan las medias asociadas a dicho archivo EDL.
Prioridad	Media.
Referencias	RF2, RF8.
Precondiciones	El EDL debe estar cargado.

Flujo Normal de eventos		
	Actor	**Sistema**
1.		1.1 El sistema lee el fichero EDL cargado buscando los archivos de tipo media con las extensiones de video (*.avi,*.mpg). 1.2 El sistema crea una lista con los nombres de las medias.

Flujos alternos	
Actor	**Sistema**
	1.1a En caso de que el sistema no encuentre algunas de las medias asociadas al EDL se muestra una interfaz con las medias implicadas notificando al usuario a través de un mensaje.

Postcondiciones	La lista de medias está confeccionada.

Tabla 5: Descripción textual del caso de uso Confeccionar lista de medias.

Descripción textual del caso de uso: Transmitir media.

Caso de Uso	Transmitir media.
Actores	Usuario.
Resumen	El caso de uso se inicia cuando el usuario accede a la opción “Transmitir”, posterior a esto se ejecuta el caso de uso “Interpretar fichero EDL” para

	reconocer las acciones de edición asociadas a las medias y transmitirlas mediante una tarjeta de video utilizando Broadcast.	
Prioridad	Alta.	
Referencias	RF7.	
Precondiciones	Debe existir un fichero EDL cargado y una tarjeta exportadora de video configurada en la computadora.	
Flujo Normal de eventos		
	Actor	**Sistema**
1.	El caso de uso se inicia cuando el usuario selecciona la opción “Transmitir”.	1.1 Se ejecuta el caso de uso “Interpretar fichero EDL”. 1.2 El sistema reproduce las medias asociadas al EDL. 1.3 El sistema transmite las medias asociadas al EDL a través de la Tarjeta de Video y finaliza así el caso de uso.
Flujos alternos		
Actor		**Sistema**
		1.1 En caso de que no se ejecute satisfactoriamente el caso de uso interpretar EDL el sistema mostrará un mensaje notificando los problemas detectados.
		1.3 En caso de que no se pueda habilitar la conexión el sistema mostrará un mensaje notificando el problema detectado.
Postcondiciones	La media es transmitida a través de la Tarjeta de Video.	

Tabla 6: Descripción textual del caso de uso Transmitir media.

Descripción textual del caso de uso: Interpretar fichero EDL.

Caso de Uso	Interpretar fichero EDL.
Actores	
Resumen	El caso de uso se inicia cuando se ejecuta el caso de uso base “Transmitir media” y finaliza una vez leído el fichero EDL transmitido según los eventos de edición que lo conforman.
Prioridad	Alta.

Referencias	RF3, RF4, RF5, RF6.	
Precondiciones	Debe existir un fichero EDL cargado.	
Flujo Normal de eventos		
	Actor	**Sistema**
1.		1.1 El sistema lee el fichero EDL para ejecutar los eventos de edición que este contiene. a) En caso de encontrar un evento de tipo corte de fragmento se ejecuta la sección “Interpretar Corte de Fragmento”. b) En caso de encontrar un evento de tipo transición: ver el caso de uso extendido “Interpretar transición”. c) En caso de encontrar un evento de tipo efecto: ver el caso de uso extendido “Interpretar efecto”.
Flujos alternos		
Actor		**Sistema**
		1.1 En caso de que el fichero EDL no contenga ningún evento de los definidos en 1.1 el sistema notifica el hecho y finaliza así el caso de uso.
Sección “Interpretar corte de fragmento”		
	Actor	**Sistema**
2.		2.1 El sistema reconoce el tiempo de comienzo y el tiempo de culminación del corte de la media indicado en el fichero. 2.2 El sistema realiza el corte al audiovisual de acuerdo a los tiempos anteriores.
Postcondiciones	La reproducción de las medias con los cortes.	

Tabla 7: Descripción textual del caso de uso Interpretar fichero EDL.

Descripción textual del caso de uso: Interpretar transición.

Caso de Uso	Interpretar transición.
Actores	
Resumen	El caso de uso se inicia cuando se ejecuta el caso de uso base “Interpretar fichero EDL” se interpreta qué tipo de transición se debe realizar y luego se

	muestra el resultado, finalizando así el caso de uso.
Prioridad	Media.
Referencias	RF4.
Precondiciones	Debe existir un fichero EDL cargado.

Flujo Normal de eventos		
	Actor	**Sistema**
1.		1.1 El sistema reconoce las medias asociadas a la transición. 1.2 El sistema reconoce el tiempo de inicio y el tiempo de fin de la transición. 1.3 El sistema realiza la transición entre él o los videos. a) En caso de que el tipo de transición sea aparecer se ejecuta la sección “Aparecer”. b) En caso de que el tipo de transición sea desaparecer se ejecuta la sección “Desaparecer”. c) En caso de que el tipo de transición sea desaparecer-aparecer se ejecuta la sección “Desaparecer-aparecer”. d) En caso de que el tipo de transición sea barrido de izquierda a derecha se ejecuta la sección “Barrido de izquierda a derecha”. e) En caso de que el tipo de transición sea barrido de derecha a izquierda se ejecuta la sección “Barrido de derecha a izquierda”. f) En caso de que el tipo de transición sea barrido de arriba abajo se ejecuta la sección “Barrido de arriba a abajo”. g) En caso de que el tipo de transición sea barrido de abajo a arriba se ejecuta la sección “Barrido de abajo a arriba”. h) En caso de que el tipo de transición sea barrido de

		esquina superior izquierda a inferior derecha se ejecuta la sección “Barrido de esquina superior izquierda a inferior derecha”. i) En caso de que el tipo de transición sea barrido de esquina superior derecha a inferior izquierda se ejecuta la sección “Barrido de esquina superior derecha a inferior izquierda”. j) En caso de que el tipo de transición sea barrido de esquina inferior izquierda a superior derecha se ejecuta la sección “Barrido de esquina inferior izquierda a superior derecha”. k) En caso de que el tipo de transición sea barrido de esquina inferior derecha a superior izquierda se ejecuta la sección “Barrido de esquina inferior derecha a superior izquierda”.
Flujos alternos		
Actor		**Sistema**
		1.1 En caso de que no se reconozca ninguna transición el sistema notifica el hecho y finaliza así el caso de uso.
Sección “Aparecer”		
	Actor	**Sistema**
2.		2.1 El sistema reconoce que el tipo de transición es aparecer y muestra como resultado la visión de un nuevo conjunto de imágenes.
Sección “Desaparecer”		
	Actor	**Sistema**
3.		3.1 El sistema reconoce que el tipo de transición es desaparecer y muestra como resultado el desvanecimiento de la imagen.
Sección “Desaparecer-aparecer”		
	Actor	**Sistema**

4.		4.1 El sistema reconoce que el tipo de transición es desaparecer-aparecer, y revela cómo se va desvaneciendo la imagen y apareciendo la imagen siguiente.
Sección "Barrido de izquierda a derecha"		
	Actor	**Sistema**
5.		5.1 El sistema reconoce que el tipo de transición es barrido de izquierda a derecha y revela cómo la imagen se va excluyendo desde el lateral izquierdo al derecho.
Sección "Barrido de derecha a izquierda"		
	Actor	**Sistema**
6.		6.1 El sistema reconoce que el tipo de transición es barrido de derecha a izquierda y revela cómo la imagen se va excluyendo desde el lateral derecho al izquierdo.
Sección "Barrido de arriba a abajo"		
	Actor	**Sistema**
7.		7.1 El sistema reconoce que el tipo de transición es barrido de arriba a abajo y muestra cómo la imagen se va excluyendo desde la parte superior de la pantalla hacia la inferior.
Sección "Barrido de abajo a arriba"		
	Actor	**Sistema**
8.		8.1 El sistema reconoce que el tipo de transición es barrido de abajo a arriba y muestra cómo la imagen se va excluyendo desde la parte inferior de la pantalla hacia la superior.
Sección "Barrido de esquina superior izquierda a inferior derecha"		
	Actor	**Sistema**
9.		9.1 El sistema reconoce que el tipo de transición es barrido de esquina superior izquierda a inferior derecha y muestra cómo la imagen se va excluyendo desde la esquina superior izquierda hacia la esquina inferior derecha.
Sección "Barrido de esquina superior derecha a inferior izquierda"		
	Actor	**Sistema**

10.		10.1 El sistema reconoce que el tipo de transición es barrido de esquina superior derecha a inferior izquierda y muestra cómo la imagen se va excluyendo desde la esquina superior derecha hacia la esquina inferior izquierda.
Sección "Barrido de esquina inferior izquierda a superior derecha"		
	Actor	**Sistema**
11.		11.1 El sistema reconoce el tipo de transición barrido de esquina inferior izquierda a superior derecha y muestra cómo la imagen se va excluyendo desde la esquina inferior izquierda hacia la esquina superior derecha.
Sección "Barrido de esquina inferior derecha a superior izquierda"		
	Actor	**Sistema**
12		12.1 El sistema reconoce el tipo de transición barrido de esquina inferior derecha a superior izquierda y muestra cómo la imagen se va disolviendo desde la esquina inferior derecha hacia la esquina superior izquierda.
Postcondiciones	Una transición sobre una media es realizada.	

Tabla 8: Descripción textual del caso de uso Interpretar transición.

Descripción textual del caso de uso: Interpretar efecto.

Caso de Uso	Interpretar efecto.	
Actores		
Resumen	El caso de uso se inicia cuando se ejecuta el caso de uso base "Interpretar fichero EDL", se interpreta que tipo de efecto se va realizar y luego se muestra el resultado finalizando así el mismo.	
Prioridad	Media	
Referencias	RF5.	
Precondiciones	Debe existir un fichero EDL cargado.	
Flujo Normal de eventos		
	Actor	**Sistema**

1.		1.1 El sistema reconoce las medias asociadas al efecto. 1.2 El sistema realiza el efecto entre él o los videos. a) En caso de que el tipo de efecto sea convertir a blanco y negro se ejecuta la sección “Convertir a blanco y negro”. b) En caso de que el tipo de efecto sea modificar brillo se ejecuta la sección “Modificar brillo”. c) En caso de que el tipo de efecto sea convertir a escala de grises se ejecuta la sección “Convertir a escala de grises”. d) En caso de que el tipo de efecto sea invertir colores se ejecuta la sección “Invertir colores”. e) En caso de que el tipo de efecto sea suprimir azul se ejecuta la sección “Suprimir azul”. f) En caso de que el tipo de efecto sea suprimir rojo se ejecuta la sección “Suprimir rojo”. g) En caso de que el tipo de efecto sea suprimir verde se ejecuta la sección “Suprimir verde”. h) En caso de que el tipo de efecto sea modificar color se ejecuta la sección “Modificar color”.
Flujos alternos		
Actor		**Sistema**
		1.1 En caso de que no se reconozca ningún efecto el sistema notifica el hecho y finaliza así el caso de uso.
Sección “Convertir a blanco y negro”		
	Actor	**Sistema**
2.		2.1 El sistema reconoce que el tipo de efecto es convertir a blanco y negro y a partir de ese momento muestra el video solo con los colores blanco y negro.
Sección “Modificar brillo”		
	Actor	**Sistema**
3.		3.1 El sistema reconoce que el tipo de efecto es modificar brillo.

		3.2 El sistema reconoce el valor a modificar. 3.3 El sistema modifica el brillo de acuerdo al valor anterior.
Sección "Convertir a escala de grises"		
	Actor	**Sistema**
4.		4.1 El sistema reconoce que el tipo de efecto es convertir a escala de grises y a partir de ese momento muestra el video solo en color gris.
Sección "Invertir colores"		
	Actor	**Sistema**
5.		5.1 El sistema reconoce que el tipo de efecto es invertir colores y a partir de ese momento muestra el video con todos los colores que aparecen en él invertidos.
Sección "Suprimir azul"		
	Actor	**Sistema**
6.		6.1 El sistema reconoce que el tipo de efecto es suprimir azul y a partir de ese momento muestra el video con todos los colores contenidos en él menos el azul.
Sección "Suprimir rojo"		
	Actor	**Sistema**
7.		7.1 El sistema reconoce que el tipo de efecto es suprimir rojo y a partir de ese momento muestra el video con todos los colores contenidos en él menos el rojo.
Sección "Suprimir verde"		
	Actor	**Sistema**
8.		8.1 El sistema reconoce que el tipo de efecto es suprimir verde y a partir de ese momento muestra el video con todos los colores contenidos en él menos el verde.
Sección "Modificar color"		
	Actor	**Sistema**
9.		9.1 El sistema reconoce que el tipo de efecto es modificar

		color y a partir de ese momento muestra el video con los colores azul, rojo y verde más intensos o menos intensos.
Postcondiciones	Un efecto sobre una media es realizado.	

Tabla 9: Descripción textual del caso de uso Interpretar efecto.

GLOSARIO DE TÉRMINOS

AM: Se refiere a lo que se conoce como Amplitud Modulada. Es un tipo de modulación lineal que consiste en hacer variar la amplitud de la señal portadora de forma que esta cambie de acuerdo con las variaciones de nivel de la señal que contiene la información que se desea transmitir, llamada señal moduladora.

API: Application Programming Interface - Interfaz de Programación de Aplicaciones. Es el conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Datashow: Un proyector de vídeo o vídeo proyector. Es un aparato que recibe una señal de vídeo y proyecta la imagen correspondiente en una pantalla de proyección usando un sistema de lentes, permitiendo así visualizar imágenes fijas o en movimiento.

Departamento de Señales Digitales: Departamento que forma parte del Centro de Desarrollo "*Geoinformática y Señales Digitales"* de la UCI que ejecuta proyectos de software asociados con el procesamiento de imágenes, sonido y video digital.

Documentación técnica: Documentación asociada al proceso ingenieril del desarrollo del software que a menudo está asociada con la ejecución de una metodología de desarrollo de software determinada.

FM: La frecuencia modulada es una modulación angular que transmite información a través de una onda portadora variando su frecuencia (contrastando esta con la AM), en donde la amplitud de la onda es variada mientras que su frecuencia se mantiene constante.

GNU/GPL: Licencia Pública General de GNU. Es una licencia creada por la Free Software Foundation en 1989 y está orientada a proteger la libre distribución, modificación y uso de software.

GUI: Graphical User Interface - Interfaz Gráfica de Usuario. Es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático. Utiliza un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz.

Herramienta informática: Una herramienta informática es un tipo de software que permite la interacción entre usuario y computadora (comunicación). Son aquellos programas que permiten que el usuario pueda elegir opciones y ejecutar acciones que el programa le ofrece.

IEEE: The Institute of Electrical and Electronics Engineers - El Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas. Es la mayor asociación internacional sin fines de lucro formada por profesionales de las nuevas tecnologías, como ingenieros eléctricos, ingenieros en electrónica, científicos de la computación e ingenieros en telecomunicación.

Infocinta: Cintillo informativo que se muestra en la televisión promocionando eventos de importancia, noticias relevantes que se transmitirán o información general para los televidentes.

Plugin: Es un módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande. Son muy utilizados en los programas navegadores para ampliar sus funcionalidades.

Programación Orientada a Objetos: Como su mismo nombre indica, la programación orientada a objetos se basa en la idea de un objeto, que es una combinación de variables locales y procedimientos llamados métodos que juntos conforman una entidad de programación.

Sistema Gestor de Bases de Datos: Son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta.