Universidad de las Ciencias Informáticas
Facultad 6

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

Título: Componente para la reproducción avanzada de video IP.

Autor:
Luis Alberto Terrero Ramirez

Tutor
Ing. Cesar Santos Sanabria

Junio de 2012
“Año 54 de la revolución”

DECLARACIÓN DE AUTORÍA

Declaro ser autor del presente trabajo de diploma y autorizo al Centro GEYSED de la Universidad de las Ciencias Informáticas para que haga el uso que estime pertinente con el mismo.

Para que así conste firmo la presente a los 21 días del mes de junio del año 2012.

_____
Firma del Autor
Luis Alberto Terrero Ramirez

_____
Firma del Tutor
Ing. Cesar Santos Sanabria

DEDICATORIA

A mis dos razones de ser: mi mamá y mi hermanito que me vieron salir hace cinco años de casa para comenzar la universidad y que hoy comparten este momento conmigo, como lo han hecho siempre en los buenos, y en los malos que han sido muchos, para ellos mi tesis, mi futuro título de ingeniero informático, y mi vida.

AGRADECIMIENTOS

A mi mamá, por ella estoy aquí hoy haciendo realidad mi sueño, un millón de gracias por confiar en mí siempre y estar ahí incondicionalmente, gracias por haber dado cuando no tenías, gracias por buscar la forma de que tus hijos alcancen sus metas, a ti va este logro porque eres lo más grande que tengo en la vida.

A mi hermanito a quien quiero con la vida, aunque no lo diga muy seguido y para quien trato de ser un ejemplo con cada paso que doy, eres quien con más facilidad me hace reír. Te quiero mucho

A Frank por haber sido parte de mi vida estos últimos dos años, por haberme hecho parte de lo que soy, y sobre todo por haberme soportado todo este tiempo.

A Yainery por haberse convertido en esa hermana que mis padres no pudieron darme, contigo he aprendido que existe la sencillez y la amistad verdadera en el mundo.

A Didzán por correr hacia donde yo estaba siempre que lo necesité, sin ti, sin tus consejos, no estuviese hoy frente a este auditorio.

A ti David que hiciste mucho por mí en muy poco tiempo. Eres un gran amigo y aunque estés lejos hace tres años ya, nada me ha hecho más feliz que verte cuando estabas en esta posición y yo estaba en la tuya.

A todos los que han compartido conmigo un aula, un apartamento, un laboratorio, de todos he aprendido un poquito y por ello les agradezco también.

A mi gran amigo Anibal, que hoy, aunque lejos, sé que está viviendo este momento conmigo.

A mi tutor Cesar, gracias por tu optimismo y ayuda.

A mi tribunal y mi oponente gracias por sus sugerencias y consejos.

RESUMEN

El tratamiento del video digital tiene un campo de aplicación hoy en día muy extenso dentro del mundo de la informática. Se han creado y se mantienen en constante evolución, reproductores y editores para los archivos de media. Los disímiles formatos contenedores de audio y video han facilitado la transportación de este tipo de información a través de los medios de comunicación, desde un extremo a otro del planeta e incluso fuera de este, haciendo uso siempre de los protocolos existentes para la transmisión de datos.

El Componente para la reproducción avanzada de video IP, surge a raíz de que el conjunto de funcionalidades que brinda el reproductor actual de video del sistema Suria Vision dificulta la integración de los videos sensores al mismo, por lo que se plantea como objetivo de esta investigación facilitar la integración de los videos sensores al sistema Suria Vision. Para alcanzar dicho objetivo fue utilizada la biblioteca de funciones libvlc en conjunto con el lenguaje de programación C++, el desarrollo estuvo guiado por la metodología ágil XP. Como resultado del desarrollo se obtuvo un componente capaz de reproducir flujo de video IP y de realizar el efecto de PTZ Digital sobre dicho video. A este componente le fueron aplicadas diferentes pruebas que arrojaron resultados totalmente satisfactorios.

FFMPEG, IP, LIBVLC, MPLAYER, REPRODUCTOR, RTSP, VIDEO, VISOR

ÍNDICE

INTRODUCCIÓN 1
1. FUNDAMENTACIÓN TEÓRICA PARA LA REPRODUCCIÓN DE VIDEO EN SISTEMAS DE VIDEO VIGILANCIA 5
1.1. Introducción 5
1.2. Conceptos relacionados con el dominio del problema planteado. 5
1.3. En el mundo. 8
1.3.1. Biblioteca libvlc. 12
1.4. Video IP 13
1.4.1. Protocolos TCP/IP utilizados para la transmisión de video IP 14
1.4.2. PTZ digital sobre el video IP 15
1.5. Conclusiones 18
2. HERRAMIENTAS Y TECNOLOGÍAS UTILIZADAS 19
2.1. Introducción 19
2.2. Metodología de desarrollo: XP 19
2.3. Lenguaje de Modelado: UML 21
2.4. Lenguaje de programación: C++ 22
2.5. Herramienta CASE: Visual Paradigm. 23
2.6. Framework de desarrollo: Qt 24
2.7. Entorno de desarrollo: Qt Creator 25
2.8. Conclusiones 26
3. PROPUESTA DEL SISTEMA. 26
3.1. Introducción 26
3.2. Propuesta del sistema 26
3.3. Personas relacionadas con el sistema. 26
3.4. Exploración 27
3.5. Historias de usuario 27
3.6. Fase de planificación 30
3.6.1. Estimación de esfuerzo por Historias de Usuarios. 31
3.6.2. Plan de iteraciones 31
3.6.3. Plan de duración de las iteraciones. 32
3.7. Diseño 33

3.7.1. Tarjetas CRC 33
3.8. **Patrones de diseño** 36
3.9. Conclusiones 36
4. IMPLEMENTACIÓN 38
4.1. Introducción 38
4.2. Estándares de codificación 38
4.3. Tareas de la ingeniería para dar solución a las historias de usuario. 39
4.3.1. Primera iteración. 39
4.3.2. Segunda iteración 40
4.3.3. Tercera iteración 41
4.4. Fase de Prueba 41
4.4.1. Pruebas unitarias 42
4.4.2. Pruebas de aceptación 42
4.5. Conclusiones 46
CONCLUSIONES GENERALES 47
RECOMENDACIONES 48
REFERENCIAS BIBLIOGRÁFICAS 49
BIBLIOGRAFÍA 51
GLOSARIO DE TÉRMINOS 54

ÍNDICE DE TABLAS

Tabla 1 - Protocolos TCP/IP utilizados para la transmisión de video IP. 14
Tabla 2.Personas relacionadas con el sistema. 27
Tabla 3.Reproducir flujo de video RTSP. 28
Tabla 4. Pintar sobre el video que se está reproduciendo. 28
Tabla 5. Capturar una imagen estática perteneciente al video (snapshot). 29
Tabla 6. Realizar efecto PTZ Digital. 29
Tabla 7. Activar/Desactivar Sonido. 29
Tabla 8. Controlar el volumen. 30
Tabla 9.Estimación de esfuerzo por Historias de Usuario. 31
Tabla 10. Plan de duración de las iteraciones. 32
Tabla 11. Plan de entregas del sistema propuesto. 33
Tabla 12. Tarjeta CRC NPlayer 33
Tabla 13 Tarjeta CRC QPlayer 34
Tabla 14 Tarjeta CRC IPlayerWidget 35
Tabla 15 Tarjeta CRC GLResizeEvent 35
Tabla 16 Tarjeta CRC MainWindow 35
Tabla 17. Tarea 1-Reproducir flujo de video RTSP. 39
Tabla 18. Tarea 2-Dibujar sobre el video que se está reproduciendo 40
Tabla 19 Tarea 3. Capturar una imagen estática perteneciente al video. 40
Tabla 20 Tarea 4. Realizar efecto PTZ Digital sobre el video. 40
Tabla 21 Tarea 5. Activar/Desactivar sonido. 41
Tabla 22 Tarea 6. Controlar el volumen del sonido. 41
Tabla 23. Caso de prueba de aceptación HU- Reproducir flujo de video RTSP. 43
Tabla 24. Caso de prueba de aceptación HU-Pintar sobre el video que se está reproduciendo. 43
Tabla 25. Caso de prueba de aceptación HU-Capturar una imagen estática perteneciente al video. 44
Tabla 26. Caso de prueba de aceptación HU- Realizar efecto PTZ Digital sobre el video. 44
Tabla 27. Caso de prueba de aceptación HU-Activar/Desactivar sonido. 45
Tabla 28. Caso de prueba de aceptación HU-Controlar el volumen del sonido. 45

INTRODUCCIÓN

Con el paso del tiempo la sociedad en general vio la necesidad de resguardar sus intereses particulares, bienes materiales y bienestar personal o de los suyos, debido a los casos de robos, fraudes, secuestros, asesinatos y todo tipo de acto delictivo que se estaban convirtiendo en una realidad y que aumentaban en numero día tras día. Para contrarrestar este tipo de actividades se comenzó a utilizar personas que se encargaban de asegurar la protección, a otras personas o a bienes materiales.

A lo largo de los años, el uso de personal de seguridad dejó de ser suficiente, debido a que los hechos delictivos continuaban perfeccionándose directamente proporcional al desarrollo de cada región o país. En ocasiones era imposible para un personal especializado en seguridad, proteger de manera eficiente los medios que le fueron asignados. A raíz de esto surge, el que fue considerado el primer sistema de video vigilancia, en el año 1965, que estaba en manos del personal de policía de Estados Unidos. Ya en 1969, las cámaras de la policía habían sido montadas en áreas estratégicas de la ciudad de Nueva York, y a partir de ese momento fueron creciendo y desarrollándose tecnológicamente. Hoy día, pueden cubrir toda una gran ciudad. Estos sistemas son utilizados por empresas tan pequeñas como un sencillo negocio privado, hasta por grandes empresas multinacionales, para las cuales una falla de seguridad significa la pérdida de miles de millones de dólares. Ofrecen además muchas ventajas al usuario: son fáciles de utilizar, la información que se observa es muy fiable y la video vigilancia puede hacerse vía remota, desde cualquier PC, PDA[1], e incluso un simple teléfono móvil conectado a Internet, incluso por varias personas al mismo tiempo, aunque esto no significa que no sea necesario el trabajo de las personas, pues siempre serán ellas las encargadas de ejecutar acciones sobre las infracciones de seguridad que puedan ser detectadas por dicho sistema.

Los sistemas de video vigilancia IP[2], emplean cámaras que capturan video y son capaces de transmitirlo bajo el protocolo IP utilizando para manejar el flujo de datos el protocolo RTSP[3] (*Real Time Streaming Protocol*), un protocolo a nivel de aplicación de presentación multimedia cliente/servidor, que permite controlar el flujo de video IP.

[1] - Dispositivo de mano que brinda utilidades al usuario: agenda, calculadora, orientación, conexión a redes.
[2] - Internet Protocol (Protocolo de Internet)
[3] - Real Time Streaming Protocol (Protocolo de Transmisión en Tiempo Real)

En Cuba se encuentra la Universidad de las Ciencias Informáticas, una universidad que vincula el estudio con el trabajo guiándose por un nuevo modelo de formación, que tiene entre sus fines la creación de software para la resolución de problemas reales. Esta universidad posee varios centros de desarrollo de software, entre ellos el centro GEYSED ubicado en la facultad # 6 encargado de gestionar los proyectos relacionados con el video y el sonido digital, y al que pertenece el proyecto Video Vigilancia que ha estado desarrollando y ya ha liberado una versión de un sistema que lleva por nombre “Suria Vision”, capaz de decodificar el flujo de video que envían las videocámaras de seguridad, éste además le facilita al usuario la gestión de las videocámaras que estén conectadas al sistema y cuenta con un reproductor que permite realizar solamente las operaciones “reproducir”, “pausar” y “detener”, sobre el video que se está reproduciendo. Estas operaciones no satisfacen los requerimientos actuales que se presentan para las nuevas versiones del software ya que se hace necesario realizar operaciones de pintado sobre el video, para facilitar el envío de datos a otros componentes del sistema que necesitan realizar sus operaciones de acuerdo a lo que el cliente desea, ya sea solo un área del video, una imagen, una porción de la imagen que el seleccione, o solo recibir una alerta en medio del video, enviada por el sistema, es por ello que se establece como situación problemática: La necesidad de un reproductor de video para el sistema que le permita al usuario la reproducción avanzada sobre los flujos de video que se están capturando, de esta manera el sistema Suria Vision podrá integrarse con el subsistema de video sensores que se ha estado desarrollando para el mismo.

Se define como **problema a resolver** que**:** El conjunto de funcionalidades que brinda el componente reproductor de video de Suria Vision dificulta la integración de los videos sensores con el sistema.

Por ello se toma como **objeto de estudio**: El proceso de reproducción avanzada de video.

Definiéndose como **campo de acción**: La reproducción avanzada de Video IP en el sistema Suria Vision.

Para dar solución al problema científico que se plantea, se establece como **objetivo general**: Facilitar la integración de los videos sensores al sistema Suria Vision.

Se establece como **idea a defender** que: Si se desarrolla un reproductor de video capaz de dibujar alertas y notificaciones, permitir la selección de áreas sobre la imagen y realizar PTZ Digital se facilitará la integración de los video-sensores con el sistema Suria Vision.

Para dar cumplimiento al objetivo anteriormente planteado se han definido las siguientes **tareas de la investigación científica:**

- Caracterizar las tendencias actuales respecto a la captura y reproducción de video IP.
- Analizar los principales protocolos para la transmisión de video IP utilizados por cámaras de video vigilancia.
- Identificar la tecnología a utilizar para la recepción de flujos de video IP.
- Realizar el levantamiento de los requisitos de software.
- Identificar una arquitectura que robusta y flexible.
- Realizar el diseño del sistema.
- Implementar el componente.
- Validar el componente implementado.

Al finalizar la investigación se esperan los siguientes resultados:

- Un componente de visualización para video que permita la reproducción avanzada de video IP.
- La documentación asociada al componente desarrollado.

Para lograr un mejor entendimiento la investigación se organizó en 4 capítulos, de cada uno de los cuales se muestra una breve descripción a continuación:

Capítulo 1: Se especificarán los principales conceptos asociados a la investigación. Se caracterizará el efecto de video “PTZ digital” y se definen cuales elementos de los analizados serán utilizados para el desarrollo del sistema. Se estudiarán las bibliotecas de software libre que brinden funcionalidades como las que son necesitadas para el sistema Suria Vision y se definirá cual de ellas será empleada para la implementación del componente.

Capítulo 2: Se explican las metodologías y herramientas seleccionadas para el desarrollo de la aplicación.

Capítulo 3: Se describe la solución propuesta y se da una explicación del proceso de modelación de la misma y sus características generales. Se muestran los diagramas necesarios para un mayor entendimiento.

Capítulo 4: Se explica brevemente cómo se realiza la implementación del software y se describen los casos de prueba aplicados al mismo.

1. FUNDAMENTACIÓN TEÓRICA PARA LA REPRODUCCIÓN DE VIDEO EN SISTEMAS DE VIDEO VIGILANCIA

1.1. Introducción

En este capítulo se especificará qué es un sistema para la video vigilancia, qué son los video sensores, las cámaras IP y el concepto de componente que será utilizado en la investigación. Se caracterizará el efecto de video “PTZ digital” y se definen cuales elementos de los analizados serán utilizados para el desarrollo del sistema. Se estudiarán las bibliotecas de software libre que brinden funcionalidades como las que son necesitadas para el sistema Suria Vision y se definirá cual de ellas será empleada para la implementación del componente.

1.2. Conceptos relacionados con el dominio del problema planteado.

Sistemas de Video Vigilancia: Los sistemas de vigilancia han evolucionado atravesando diferentes etapas en las últimas décadas. La primera generación de sistemas de vigilancia basados en video emplea señales y transmisión analógicas. En la toma de decisiones, siempre es el operador humano el encargado de realizar todas las tareas de análisis de secuencias de video presentadas en varios monitores situados en una sala de control remota, donde las escenas monitorizadas por las distintas cámaras se *multiplexan* y presentan en un orden periódico y predefinido. Adicionalmente, la vigilancia tradicional precisa gran cantidad de espacio de almacenamiento. Todo lo que captura una cámara de seguridad se graba en cintas que, o bien se sobrescriben periódicamente, o bien se guardan en un archivo de video.

> Los sistemas analógicos emplean amplificadores de baja potencia y cables coaxiales para ver de forma remota las imágenes de las cámaras de seguridad. Las cámaras y los monitores están interconectados mediante una red de conmutación y distribución de video complejo y caro, que direcciona las imágenes de cada cámara hacia un monitor. Esta tecnología tiene demasiadas limitaciones y defectos. El video analógico solo se puede distribuir en una red local de cables coaxiales. A continuación se representa a través de una imagen dicho proceso. (1)

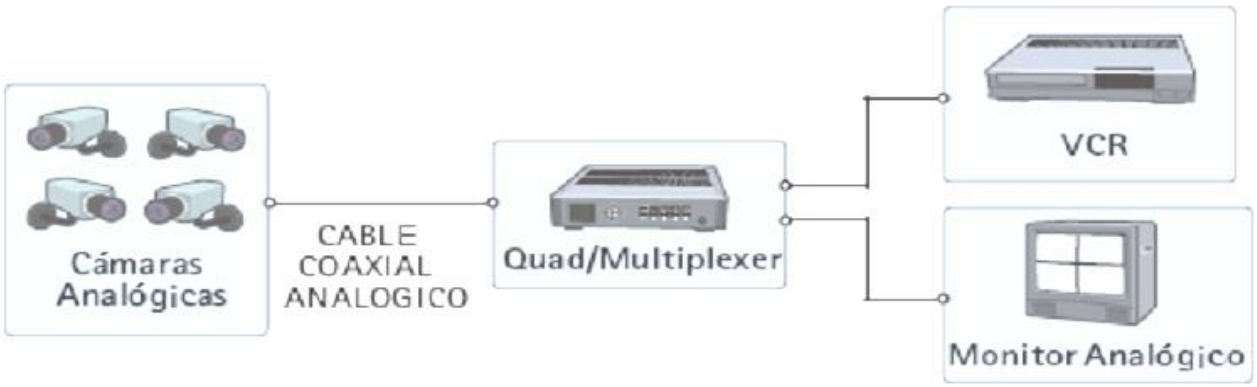

Figura 1- Primera generación de sistemas de video vigilancia (1)

Con el paso del tiempo se fueron perfeccionamiento estas tecnologías y surgieron los sistemas de vigilancia basados en métodos de procesamiento y comunicación híbridos analógico-digitales. Estos utilizaban los primeros algoritmos de procesado de video que centraban la atención en métodos de compresión digital para aprovechar el ancho de banda de transmisión.

Ya la tercera generación de los sistemas de video vigilancia aprovechan el progreso de las redes de ordenadores y las comunicaciones multimedia fijas, y móviles. La investigación en este campo trabaja en las siguientes líneas:

- Técnicas distribuidas de procesamiento de video.
- Procesado de secuencias de video en tiempo real.
- Conseguir sistemas robustos de transmisión de imagen.
- Procesamiento de imagen en color.
- Generación de alarmas basada en eventos.
- Reconstrucción de secuencias a partir de modelos.
- Segmentación y análisis en tiempo real de secuencias de imágenes 2D.
- Identificación y seguimiento de múltiples objetos en escenas complejas.
- Reconocimiento de comportamientos humanos. (1)

A continuación se muestra un esquema de dicha generación de sistemas de video vigilancia.

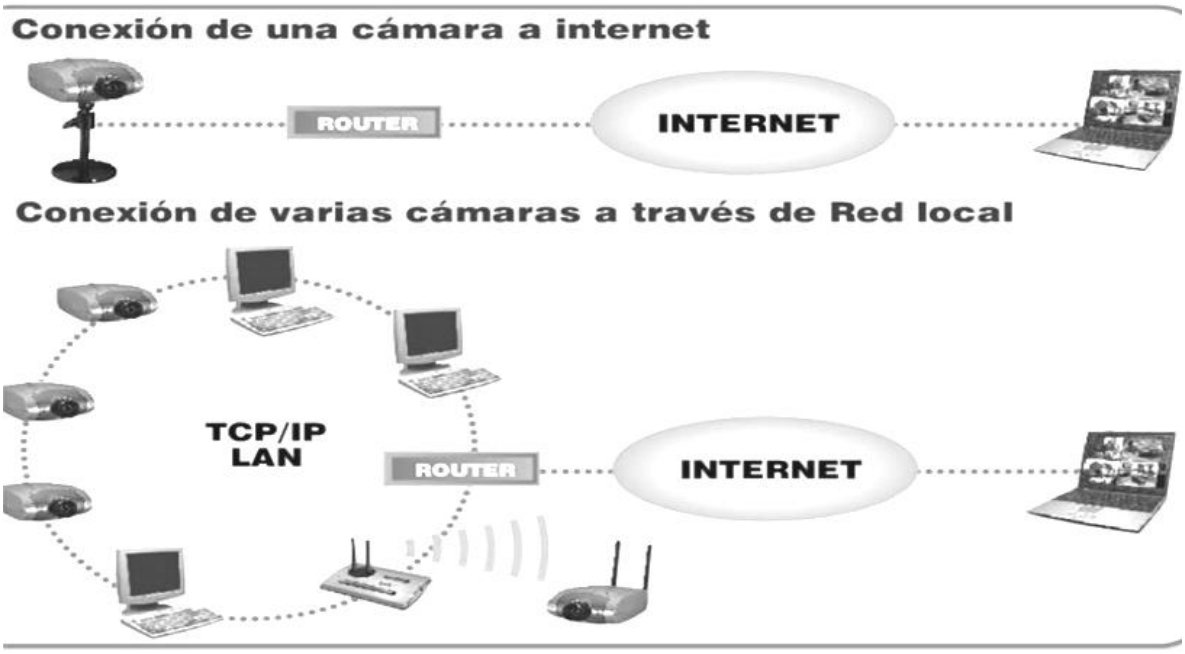

Figura 2- Tercera generación de sistemas de video vigilancia. (1)

En la actualidad, debido al desarrollo tecnológico alcanzado, este tipo de sistema de video vigilancia es el que más se utiliza, aprovechando así las facilidades que le brindan al personal encargado de controlar y monitorizar su correcto funcionamiento.

Video sensor: es una herramienta de análisis de video digital que ofrece información significativa proveniente de una secuencia de video. El desarrollo de video sensores tiene su base en el procesamiento digital de la imagen. (2)

Cámara IP: Puede describirse como una cámara y un ordenador combinados para formar una única unidad inteligente. Captura y envía video en vivo, a través de una red IP, como una LAN, Intranet o Internet, permite a los usuarios ver o gestionar la cámara con un navegador Web estándar o con software de gestión de video en cualquier equipo local o remoto conectado a una red. Permite a usuarios autorizados de distintas ubicaciones acceder simultáneamente a las imágenes captadas por la misma cámara IP de red. (3)

Componente: (Según el SEI[4]) es una implementación opaca de una funcionalidad, sujeta a composición por terceros y que cumple con un modelo de componentes. Con respecto al primer aspecto, un componente se considera una implementación opaca debido a que su distribución predominantemente es en formato binario y sus consumidores lo utilizan como una "caja negra" a través de su interfaz. Dicho aspecto está alineado con el principio de encapsulamiento de la programación orientada a objetos. Por otra parte, la composición por terceros implica que los componentes son intrínsecamente reutilizables debido a que un sistema basado en componentes puede ser ensamblado con relativa facilidad por un integrador empleando componentes suministrados por múltiples proveedores independientes. Finalmente, la coordinación e interacción entre componentes exige un marco regulatorio estandarizado (modelo de componentes) que establece la infraestructura de software requerida (*framework*) y las convenciones y restricciones de diseño de los mismos. (4)

1.3. En el mundo.

A continuación serán nombrados los diferentes proyectos que han creado bibliotecas para el trabajo con archivos de media que son completamente de software libre y se realizará un estudio de las cada una de ellas. Se hará especial énfasis en las que brinden funcionalidades para la captura y reproducción de flujo de video IP a través de las redes de comunicación, así como la captura de los fotogramas del video que se está reproduciendo.

Mplayer Project: Comenzó siendo solo una persona que encontró algunas fallas en los reproductores existentes para Linux, actualmente los miembros de su equipo están dispersos por todas partes del mundo, desarrollando el reproductor que fue nombrado igual que el proyecto "MPlayer".

FFmpeg Project: Este proyecto fue creado por Gerard Lantau, un seudónimo de Fabrice Bellard, y ahora es mantenido por Michael Niedermayer. Todo el desarrollo que se realiza en las bibliotecas FFmpeg es distribuido bajo licencia GPL[5].

VideoLan Project: Surge como un proyecto académico desarrollado por estudiantes, utilizado en sus inicios como un cliente y un servidor para transmitir video en el campus universitario. Actualmente es

[4] Instituto de Ingeniería de Software (por sus siglas en inglés)
[5] General Public License

desarrollado por personas de todo el planeta y coordinado por la organización francesa *VideoLan*. Produce software libre para el trabajo con multimedia y es liberado bajo la licencia GPL. (5)

Luego de conocer las diferentes empresas o proyectos que se dedican a la creación de bibliotecas libres de código abierto para el trabajo con archivos de media, se realizará un estudio de las características que ofrecen cada una de ellas:

Mplayer: Es una biblioteca para el trabajo con archivos de media multiplataforma y liberada bajo la licencia GPL. Trae implícita decodificadores para los archivos MPEG, VOB, AVI, OGG/OGM,MKV, VIVO, ASF/WMA/WMV, QT/MOV/MP4, FLI, RM, NuppelVideo, YUV4MPEG, FILM, RoQ, PVA, soportados por algunos códecs[6] nativos, XAnim, y DLL's Win32.

Además puede ser utilizada para reproducir VideoCD, SVCD, DVD, 3ivx y DivX/Xvid 3/4/5.

También trae la opciones para subtítulos, soportando 14 formatos diferentes (MicroDVD, SubRip, SubViewer, Sami, VPlayer, RT, SSA, AQTitle, JACOsub, VobSub, CC, OGM, PJS y MPsub).

Posee herramientas para la decodificación de video. Es capaz de capturar y reproducir flujo de video proveniente desde la red. Fue utilizada para el desarrollo del reproductor de video *Mplayer.* (6)

Tiene como única desventaja para su utilización en el desarrollo del componente que no permite la captura de los fotogramas del video que se está reproduciendo.

FFmpeg: Es una biblioteca libre de código abierto que brinda funcionalidades para grabar, convertir y hacer emisiones de audio y vídeo. Está desarrollada en GNU/Linux, pero puede ser compilada en la mayoría de los sistemas operativos, incluyendo Windows. Es destacable que la mayoría de los desarrolladores de FFmpeg lo sean también del proyecto Mplayer. Está liberada bajo licencia General Public License (*GPL*).

La biblioteca FFmpeg está compuesta por los siguientes módulos:

- *ffmpeg*: es una herramienta de línea de comandos para convertir un video de un formato a otro. También puede capturar y codificar en tiempo real desde una tarjeta de televisión.

[6] Abreviatura de codificador-decodificador

- *ffserver*: es un servidor para la emisión multimedia en directo que soporta HTTP (la compatibilidad con RTSP aun se encuentra en desarrollo). Todavía no está en fase estable.
- *ffplay*: es un reproductor multimedia basado en SDL y las bibliotecas FFmpeg.
- *libavcodec*: es una biblioteca que contiene todos los códecs de FFmpeg. Muchos de ellos fueron desarrollados desde cero para asegurar una mayor eficiencia y un código altamente reutilizable.
- *libavformat*: es una biblioteca que contiene los multiplexadores/demultiplexadores para los archivos contenedores multimedia.
- *libavutil*: es una biblioteca de apoyo que contiene todas las rutinas comunes en las diferentes partes de FFmpeg.
- *libpostproc*: es una biblioteca de funciones de post proceso de video.
- *libswscale*: es la biblioteca de escalado de video.

Sus decodificadores están implementados en el módulo *libavcodec* dirigidos fundamentalmente a los siguientes formatos contenedores: Asus v1, Asus v2, AVS (solo decodificación), CamStudio (solo decodificación), Cinepak (solo decodificación), Creative YUV (CYUV, solo decodificación), Dirac (solo decodificación), DNxHD, Duck TrueMotion v1 (solo decodificación), Duck TrueMotion v2 (solo decodificación), Flash Screen Video, FFV1, H.261, H.263, H.264/MPEG-4 AVC (decodificador nativo, codifica usando x264), Huffyuv, id Software RoQ Video, Intel Indeo (solo decodificación), Lagarith (solo decodificación), LOCO (solo decodificación), Mimic (solo decodificación), MJPEG, MPEG-1, MPEG-2/H.262, MPEG-4 Part 2 (DivX y Xvid codecs), On2 VP3 (solo decodificación), On2 VP5 (solo decodificación), On2 VP6 (solo decodificación), On2 VP8 (decodificador nativo, codificada a través de libvpx), Apple ProRes, Apple Computer QuickDraw (solo decodificación), QuickTime Graphics SMC, RealVideo RV10 y RV20, RealVideo RV30 y RV40 (solo decodificación), VC-1 (solo decodificación), Smacker video (solo decodificación), Snow, Sorenson SVQ1, Sorenson SVQ3 (solo decodificación), Theora (decodificador nativo, codificada a través de libtheora), Sierra VMD Video (solo decodificación), VMware VMnc (solo decodificación), Westwood Studios VQA (solo decodificación), WMV — versión 7 y 8, WMV — versión 9 (solo decodificación), Wing Commander/Xan Video (solo decodificación).
(7)

A pesar de que esta biblioteca permite la decodificación de varios formatos contenedores de video, la emisión y recepción de flujo de video IP y la captura de los fotogramas del video que se está reproduciendo haciendo uso del módulo *libavcodec*, su principal objetivo es la conversión entre formatos contenedores y no la reproducción de video IP, añadiendo además, que aun está en fase de prueba, la compatibilidad con algunos protocolos para la transmisión de datos a través de las redes de comunicación como se especificó en el módulo *ffserver*.

LibVLC: Es una biblioteca libre para el trabajo con multimedia desarrollada por el proyecto VideoLan. Es multiplataforma con versiones disponibles para varios sistemas operativos, incluyendo Microsoft Windows, posee funcionalidades para la reproducción de muchos códecs y formatos de audio y video, además tiene la capacidad de realizar difusión de video a través de la red y de recodificar muchos formatos, también dependiendo del sistema operativo. Permite reproducir DVD, VCD, SVCD, CD Audio, DVB (a través de V4l y DirectShow), fuentes RSS/Atom y archivos almacenados en el ordenador.

Permite la decodificación de los siguientes formatos contenedores:

- 3GP, ASF, AVI, FLV, Matroska Video, Musical Instrument Digital Interface (.mid/.midi),QuickTime, MP4, Ogg, OGM, WAV, MPEG-2 (ES, PS, TS, PVA, MP3), AIFF, Raw audio, Raw DV, MXF, VOB.

Es compatible y capaz de difundir y recibir flujo de video e información usando los siguientes protocolos:

- UDP/RTP unicast o multicast, HTTP, FTP, MMS, RTSP, RTMP.

(5)

Esta biblioteca posee módulos que permiten la captura y el trabajo con los fotogramas del video que se está reproduciendo lo que la sitúa en ventaja con las anteriormente expuestas en el documento.

Nota: Los datos y características ofrecidos de cada una de las bibliotecas libres para la reproducción de software fueron tomados de los sitios web pertenecientes a cada uno de los proyectos que las desarrolla, y se encuentran plasmados en la bibliografía del presente documento.

El estudio de estas bibliotecas para la reproducción de contenido multimedia arroja las siguientes conclusiones:

- Todas poseen funcionalidades para decodificar una amplia gama de formatos contenedores de audio y video.
- Cada una de ellas permite la captura de video IP proveniente desde las redes de comunicación.
- FFmpeg permite la captura de los fotogramas del video que se está reproduciendo, pero no está completamente probada su compatibilidad con uno de los protocolos para la transmisión

de video IP a través de las redes de comunicación: el RTSP, muy utilizado por los fabricantes de cámaras IP para el envío de la información desde éstas hasta los sistemas que la controlan.

- Mplayer es compatible con todos los protocolos que se utilizan para la transmisión de video IP, pero no permite la captura de los fotogramas del video que se está reproduciendo.
- La libvlc permite la captura de los fotogramas de video y es altamente compatible con muchos protocolos para la transmisión de datos entre los sistemas de redes, es por ello que se procede a estudiar más profundamente dicha biblioteca.

Se hace especial énfasis en la captura de los fotogramas de video porque es la funcionalidad que permitirá la realización de todo tipo de operación que requiera el pintado sobre el video.

1.3.1. Biblioteca libvlc.

La biblioteca *libvlc* es una API[7] desarrollada por el proyecto VideoLan, que posee todas las funciones que realiza el reproductor del mismo proyecto y que lleva por nombre VLC. Las complejas funcionalidades implementadas en esta biblioteca pueden ser utilizadas por cualquier desarrollador ya que se distribuye como una biblioteca compartida y ahorra el hecho de implementar métodos para el trabajo con medias comenzando desde cero. Puede ser añadida en cualquier aplicación que necesite apoyarse de las facilidades que esta ofrece. (8)

La libvlc está compuesta por un conjunto de módulos que la hacen robusta y útil, a continuación se muestra un gráfico representativo de estos.

[7] Interfaz de programación de aplicaciones, por sus siglas en inglés

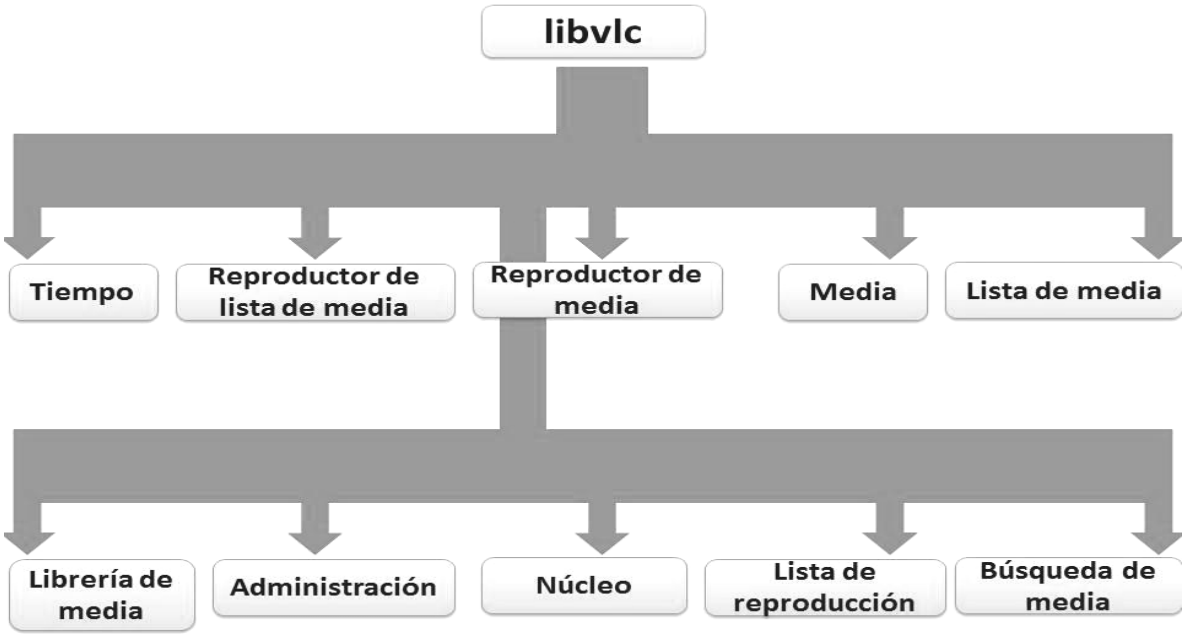

Figura 3. Módulos de la biblioteca libvlc.

Estos módulos tienen implementadas funciones para realizar todo tipo de operaciones sobre archivos de media, entre ellas realizar zoom sobre un área del video que se reproduce y que se seleccione a gusto, y aunque no permite modificar algunas características de dicho zoom digital, o sea agrandar el tamaño de la muestra ampliada o cambiar la posición en que la sitúa, su última versión le ofrece al usuario la posibilidad de capturar los fotogramas que van entrando con el flujo de video proveniente desde la red bajo el protocolo RTSP. Esta ventaja es la que facilitará el desarrollo de la presente investigación para dar cumplimiento al objetivo trazado en la misma, ya que con la captura de dichos fotogramas el equipo de desarrollo podrá reproducirlos en forma de video y realizar operaciones de pintado sobre ellos.

1.4. Video IP

Video IP: Es el video que se envía a través de las diferentes redes de comunicaciones bajo el protocolo TCP/IP o UDP.

1.4.1. Protocolos TCP/IP utilizados para la transmisión de video IP

Tabla 1 - Protocolos TCP/IP utilizados para la transmisión de video IP.

Protocolo	***Protocolo de transporte***	***Puerto***	***Uso común***	***Uso video en red***
FTP File Transfer Protocol	TCP	21	Transferencia de ficheros a través de internet.	Transferencia de imágenes o video desde una cámara de red o servidor de video a un servidor FTP o a una Aplicación.
SMTP Send Mail Transfer Protocol	TCP	25	Protocolo para el envío de e-mails.	Una cámara de red o servidor de video puede enviar imágenes o notificaciones de alarma utilizando su cliente integrado de e-mail.
HTTP Hyper Text Transfer Protocol	TCP	80	Utilizado para navegar en la web.	El modo más común de transferencia de vídeo desde una cámara de red o servidor de vídeo donde el dispositivo trabaja como un servidor web, proporcionando video al usuario o servidor de aplicación.
HTTPS Hypertext Transfer Protocol over Secure Socket Layer	TCP	443	Utilizado para acceder a páginas web de forma segura utilizando encriptación.	La transmisión de vídeo desde una cámara de red o servidor de vídeo puede ser utilizada para autenticar los envíos de la cámara utilizando certificados digitales

				X.509.
RTP Real Time Protocol	UDP/TCP	No definido	Para el envío de vídeo y audio a través de Internet. A menudo utilizado en sistemas multimedia o de video conferencia	Un modo común de transmitir vídeo en red MPEG. La transmisión puede ser *unicast* (uno a uno) o *multicast* (uno a varios).
RTSP Real Time Streaming Protocol	TCP	554	Utilizado para configurar y controlar sesiones multimedia a través de RTP.	

IP utiliza dos protocolos de transporte: Protocolo de Control de Transmisión (TCP) y el Protocolo de Datagramas de Usuario (UDP). TCP ofrece un canal de transmisión fiable basado en la conexión, y gestiona el proceso de convertir grandes bloques de datos en paquetes más pequeños, adecuados para la red física que se utiliza y garantiza que los datos enviados desde un extremo se reciben en el otro. UDP, por otro lado, es un protocolo sin conexión que no garantiza la entrega de los datos enviados, dejando así todo el mecanismo de control y comprobación de errores a cargo de la propia aplicación.

En general, TCP se utiliza cuando se prefiere una comunicación fiable durante el tiempo de espera del transporte. La fiabilidad de TCP a través de la retransmisión puede producir retrasos significativos. Por otro lado, UDP no ofrece retransmisiones de datos perdidos y, en consecuencia, no produce mayores retrasos. (9)

1.4.2. PTZ digital sobre el video IP.

PTZ: Es al acrónimo de movimiento horizontal, movimiento vertical y zoom, por sus siglas en inglés (*pan*, *tilt*, *zoom*) que se le son aplicados generalmente a una videocámara de seguridad.

Cámara PTZ: Son aquellas cámaras utilizadas mayoritariamente para la video vigilancia, a las que se les puede realizar zoom y ser manejadas mediante control remoto. Generalmente están asociadas a sistemas automatizados. Actualmente estas cámaras son capaces de detectar variación en algunos

píxeles en la imagen y enfocarse en esa área, es decir, detectar movimiento en el área que está controlando.

PTZ Digital: Este término será utilizado a lo largo de la investigación para referirse a la operación de moverse horizontal o verticalmente y realizar zoom digital sobre un área específica de los fotogramas del video IP, que se esté reproduciendo.

Zoom óptico: Es una distancia focal de longitud variable, es decir cuando se puede variar a voluntad la distancia focal y en consecuencia el ángulo de visión. Cuando la distancia focal es mayor a 5x se le conoce por súper zoom.

Tiene como propiedades básicas la distancia focal mínima y la distancia focal máxima. Cuando se habla de que el zoom óptico es de 2x, 3x, 10x, se está refiriendo a que la distancia focal máxima es 2, 3 ó 10 veces la distancia focal mínima.

Zoom digital: Es un método que aparentemente disminuye la distancia focal y el ángulo de visión de una imagen fotográfica o de video. Este tipo de zoom se realiza electrónicamente al recortar una imagen con el mismo radio de aspecto que la original (10). Generalmente se interpola el resultado de dicha operación. No se gana resolución óptica con este proceso. Es considerado en muchas ocasiones perjudicial para la calidad de la imagen debido a la acción de interpolación que se realiza a dicha imagen cuando es recortada.

En la siguiente figura se muestra una imagen a la que se le realiza Zoom Digital y Zoom Óptico en dicho orden.

Figura 4. Ejemplo del uso de zoom digital: Se realiza un zoom digital en la imagen central y un zoom óptico en la imagen de la derecha.

Existen términos relacionados con el zoom digital que son fundamentales para el entendimiento de su funcionamiento. Entre ellos se encuentran los que a continuación se relacionan y explican de manera resumida.

- *Radio de aspecto de una imagen:* también conocido como *ratio*, *ratio* de aspecto, proporción de aspecto o razón de aspecto, es la proporción entre su ancho y su alto. Se calcula dividiendo el ancho entre la altura de la imagen y se expresa como "X:Y". (11).
  La figura que se muestra a continuación es un ejemplo de varias relaciones de aspecto.

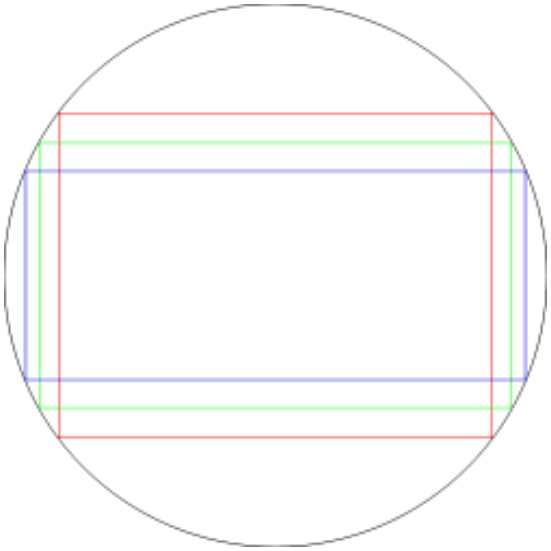

Figura 5. Comparación de las tres relaciones de aspecto más comunes. La morada (2,39:1) y la verde (1,85:1) son las más usadas en el cine. Por su parte, el recuadro rojo (4:3) es la relación más normal en televisión, junto con el «panorámico» (16:9), que ya es habitual en muchos canales y programas de televisión del mundo.

- Interpolación: En el campo de la imagen digital, se aplica el método matemático que lleva este nombre para ampliar la imagen, rellenando los espacios vacíos con datos que son "inventados" a partir de algoritmos específicos.

1.5. Conclusiones

Luego del estudio de las bibliotecas libres de código abierto *Mplayer*, *FFmpeg* y *LibVLC* para el trabajo con multimedia, se llego a la conclusión de que las tres son capaces de decodificar una amplia gama de formatos contenedores y permiten la captura y decodificación de video IP, pero la que más se adapta a las necesidades que presenta el sistema Suria Vision, de un componente de reproducción avanzada de video, es la *LibVLC*, que es la única que ofrece funcionalidades para la captura de los fotogramas del video que se está reproduciendo. Por tanto se define como biblioteca a utilizar para el desarrollo del componente la anteriormente mencionada.

2. HERRAMIENTAS Y TECNOLOGÍAS UTILIZADAS

2.1. Introducción

A la hora de crear una aplicación informática es importante conocer cuáles son las herramientas y tecnologías que brindan las opciones necesarias para obtener los resultados que se esperan. En este capítulo se hará referencia a las herramientas empleadas para el modelado del sistema y las tecnologías utilizadas en el desarrollo y documentación de la aplicación.

2.2. Metodología de desarrollo: XP

Una metodología de desarrollo es una colección de documentación formal referente a los procesos, las políticas y los procedimientos que intervienen en el desarrollo del software. La finalidad de una metodología de desarrollo es garantizar la eficacia (p. ej. cumplir los requisitos iniciales) y la eficiencia (p. ej. minimizar las pérdidas de tiempo) en el proceso de generación de software. Para suplir la falta de requisitos, casos de uso, y demás herramientas; XP utiliza historias de usuarios, la historia de usuario es una fase corta que representa alguna función que realizara el sistema. Cada historia de usuario no puede demorar en desarrollarse más de una semana, si así lo requiera, debe segmentarse. (12)

Se seleccionó XP como metodología de desarrollo porque se adapta en gran medida tanto al tipo de proyecto a desarrollar como a las condiciones de trabajo que se tienen actualmente, en otras palabras el proyecto de sistema de software es pequeño, el cliente es parte del equipo de desarrollo por lo que se mantendrá buena comunicación con el mismo, y los requisitos puede estar cambiando constantemente.

> Programación Extrema (XP por sus siglas en inglés: Xtreme Programming) es un enfoque de la ingeniería de software formulado por Kent Beck, autor del primer libro sobre la materia, *Extreme Programming Explained: Embrace Change (1999).* Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que el resto de éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en el énfasis que hace en cuanto a la adaptabilidad sobre la previsibilidad. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural,

inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.
(13)

Los Valores originales de la programación extrema son: simplicidad, comunicación, retroalimentación, coraje y respeto.

Las características fundamentales de la metodología son las que a continuación se explican:

- Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.

- Pruebas unitarias: continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación. Véase, por ejemplo, las herramientas de prueba JUnit orientada a Java, DUnit orientada a Delphi, NUnit para la plataforma.NET o PHPUnit para PHP. Estas tres últimas inspiradas en JUnit, la cual, a su vez, se insipiró en SUnit, el primer framework orientado a realizar pruebas, elaborado para el lenguaje de programación Smalltalk.

- Programación en parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone así se obtiene mayor calidad del código escrito, de esta manera éste es revisado y discutido mientras se escribe y se evita la posible pérdida de productividad inmediata.

- Frecuente integración del equipo de programación con el cliente o usuario: Se recomienda también que un representante del cliente trabaje junto al equipo de desarrollo.

- Corrección de todos los errores antes de añadir una nueva funcionalidad. Hacer entregas frecuentes.

- Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.

- Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.

- Simplicidad en el código: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

(14)

2.3. Lenguaje de Modelado: UML

El Lenguaje Unificado de Modelado (UML por sus siglas en inglés *Unified Modeling Laguage*) es el lenguaje para modelado de sistemas de software más avanzado y utilizado en la actualidad según datos de la IEEE. Es un lenguaje gráfico que se utiliza para visualizar, especificar, construir y documentar un sistema de software.

Se decide utilizar UML como lenguaje de modelado del visor debido a que la metodología que fue seleccionada XP, utiliza UML para la modelación del sistema, además de que ofrece grandes facilidades para la modelación y diseño del negocio.

UML es un lenguaje para especificar métodos o procesos, no para describirlos, que además se utiliza para definir un sistema de software, para detallar los artefactos del sistema, construirlo y documentarlo. Es el lenguaje en el que esta descrito el modelo. Se puede utilizar en una gran variedad de formas

para darle soporte a una metodología de desarrollo, pero no específica en sí que metodología o proceso usar.

2.4. Lenguaje de programación: C++

> Un lenguaje de programación es un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Es utilizado para controlar el comportamiento físico y lógico de una máquina. (15)
>
> C++: es un lenguaje de programación diseñado a mediados de los años 1980 con la intención de extender al exitoso lenguaje de programación C con mecanismos que permitiesen la manipulación de objetos. Posteriormente se añadieron facilidades de programación genérica, que se sumaron a los otros dos paradigmas que ya estaban admitidos (programación estructurada y la programación orientada a objetos). Por esto se suele decir que el C++ es un lenguaje multiparadigma. Una particularidad del C++ es la posibilidad de redefinir los operadores (sobrecarga de operadores), y de poder crear nuevos tipos que se comporten como tipos fundamentales. C++ permite trabajar tanto a alto como a bajo nivel. (16)

El sistema Suria Vision se ha venido desarrollando en varios lenguajes de programación desde sus inicios hace ya más de cuatro años. Todos los componentes que se están desarrollando para la versión actual del sistema están implementados en C++, por lo que su elección para el desarrollo del componente de reproducción avanzada, facilitará la integración con el resto del sistema. Adicionándole a esta afirmación el hecho de que la biblioteca que será utilizada para el desarrollo del componente también está desarrollada, prácticamente en su totalidad, en el mismo lenguaje, lo que ahorra tiempo y esfuerzo en el trabajo, de no utilizar C++ como lenguaje, seria más difícil el empleo de dicha biblioteca, teniendo que hacer para ello un envoltorio según el lenguaje que se fuese a emplear. Existen además otras características de este lenguaje que podrían ser muy beneficiosas y que se enuncian a continuación:

- Los componentes creados utilizando este lenguaje, son compatibles con la mayoría de los lenguajes de programación.

- Posee bibliotecas para el manejo de las interrupciones de hardware que son muy útiles para el trabajo con video, pues en varias ocasiones se accede a zonas de memoria, que podrían, en caso de no ser utilizadas adecuadamente, causar bloqueos y caídas del sistema operativo.
- Existe suficiente información sobre su uso, errores comunes y ayuda para el desarrollo.
- Existen muchos algoritmos ya desarrollados en C++ que pueden ser reutilizados y amoldarse a la solución deseada.

2.5. Herramienta CASE[8]: Visual Paradigm.

Herramienta CASE: es un conjunto de métodos, utilidades y técnicas que facilitan la automatización del ciclo de vida del desarrollo de sistemas de información, en su totalidad o parcialmente en cada una de sus fases. (17).

Generalmente son las herramientas CASE las que abarcan todos los pasos del desarrollo del software y también aquellas actividades generales que se aplican a lo largo del proceso, dos herramientas de las más utilizadas a nivel mundial son el Rational Rose y el Visual Paradigm.

Se utiliza Visual Paradigm como herramienta de modelado debido a las características que este ofrece y que se enuncian más adelante en la investigación:

Visual Paradigm: es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas, generar documentación. Proporciona además abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. (18)

Características que lo convierten en una poderosa herramienta CASE.

- Modela todos los diagramas de UML.
- Ofrece un diseño centrado en casos de uso y completamente orientado al negocio.

[8] Ingeniería de Software Asistida por Ordenador por sus siglas en inglés.

- Se integra con numerosos IDEs de desarrollo: Visual Studio, NetBeans, JDeveloper, Eclipse, JBuilder.
- Posee capacidades de ingeniería directa e inversa.
- Exportación en diferentes formatos de imagen de los diagramas elaborados: jpg, jpeg, png y svg.
- Genera código de aplicación en lenguajes como java, c++, c#, según el modelado de la aplicación.
- Genera documentación sobre el sistema que se está elaborando en formato PDF, HTML y documento de Office.

2.6. Framework de desarrollo: Qt

Después de haber seleccionado el lenguaje de programación con que se va a trabajar, se hace importante seleccionar un *framework*. El mismo debe de contar con funcionalidades que hagan que el proceso de implementación sea rápido, flexible, eficiente, multiplataforma y completamente de software libre.

Qt es una herramienta para desarrollar interfaces gráficas de usuario y también para el desarrollo de programas sin interfaz gráfica como herramientas para la consola y servidores. Qt utiliza el lenguaje de programación C++ de forma nativa, pero permite usar también C, Python y Perl. Funciona en las principales plataformas de sistemas operativos existentes. Las APIs de Qt cuentan con métodos para el trabajo con objetos gráficos de los formatos más usados en los sistemas operativos, así la gestión de hilos de procesos, soporte de red, el acceso a archivos y el manejo de ficheros, además de utilizar las estructuras de datos tradicionales.

Además de haber sido analizadas las características anteriores se selecciona Qt para el desarrollo de la interfaz gráfica de la aplicación debido a que es multiplataforma, de código abierto y ha sido empleado en toda la implementación de la actual versión del sistema Suria Vision, dicho de esta forma facilita la integración del componente con el resto del sistema. Este *framework* es además completamente gratuito y ofrece funcionalidades que serán de mucha utilidad para el desarrollo del componente y que a continuación serán enunciadas:

- Las herramientas, bibliotecas y clases están disponibles para casi todas las plataformas Unix y sus derivados (como Linux, MacOS X, Solaris, etc) como también para la familia Windows, por

lo que una aplicación puede ser compilada y utilizada en cualquier plataforma sin necesidad de cambiar el código y la aplicación se verá y actuará tan bien como una aplicación nativa.

- Qt tiene para la creación de aplicaciones una extensa biblioteca con clases y herramientas bien documentadas para el desarrollo de aplicaciones que requieran una interfaz gráfica de usuario, sobre la base de la programación orientada a objetos.
- Entre estas bibliotecas cuenta con algunas orientadas completamente a la manipulación de elementos gráficos lo que facilitará el trabajo con los fotogramas de video que se están capturando, y con los que se tiene que realizar todo el proceso de pintado para simular la reproducción del video.

2.7. Entorno de desarrollo: Qt Creator

Entorno de desarrollo integrado: (IDE por sus siglas en inglés) es un programa informático compuesto por un conjunto de herramientas de programación. Es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, es un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Los IDEs pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes que pueden dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder ser utilizado para varios.

Qt Creator es un entorno de desarrollo integrado creado por la compañía Trolltech para el desarrollo de aplicaciones con las bibliotecas Qt, Posee varias versiones, distribuidas a nivel mundial. Se escoge para el desarrollo del componente la versión 4.0.1 Los sistemas operativos que soporta en forma oficial son:

- GNU/Linux 2.6.x, para versiones de 32 y 64 bits con Qt 4.x instalado. Además hay una versión para Linux con gcc 3.3.
- Mac OS X 10.4 o superior, requiriendo Qt 4.x
- Windows XP y superiores, requiriendo el compilador MinGW y Qt 4.4.3 para MinGW.

Se escoge este entorno de desarrollo porque es altamente compatible con el framework de desarrollo Qt, posee una gran colección de ayuda para el programador y además sus ejemplos están muy bien documentados.

2.8. Conclusiones

Luego del análisis de las diferentes tecnologías expuestas anteriormente se escogió como metodología de desarrollo XP porque el equipo de desarrollo para el componente está compuesto por solo una persona, se tiene una interacción permanente con el cliente, y el sistema que se va a desarrollar no va a ser de gran, tamaño ni tiempo de duración. Se llegó a la conclusión de que el uso de C++ como lenguaje de programación facilitará la integración del componente al sistema Suria Vision. Se concluye también que el *framework* Qt ofrece las bibliotecas necesarias para el trabajo que se requiere realizar con los fotogramas del video que se está reproduciendo.

3. PROPUESTA DEL SISTEMA.

3.1. Introducción

En este capítulo se dará a conocer el contexto en el que se expone la solución propuesta. Se describirán los posibles procesos a manejar con el objetivo de comprenderlos y se especificarán aquellos que el sistema deberá soportarar. Se guiará además el desarrollo del software hacia el sistema adecuado, mediante la captura y descripción de las condiciones o capacidades que el sistema debe cumplir.

3.2. Propuesta del sistema

La solución que se propone será un componente para el sistema Suria Vision que brindará la posibilidad de manipular más fácilmente el flujo de video proveniente desde videocámaras de seguridad mediante la obtención y manipulación a conveniencia, y necesidad, de los fotogramas del vídeo que se está recibiendo. El sistema ofrecerá además las opciones de reproducir y detener la reproducción del flujo de video, subir, bajar y deshabilitar el sonido. La operación de más peso será “PTZ Digital”.

3.3. Personas relacionadas con el sistema.

Persona relacionada al sistema: toda aquella que obtiene un resultado del valor de uno o varios procesos que se ejecutan en el mismo.

Tabla 2.Personas relacionadas con el sistema.

Persona relacionada	***Descripción***
Usuario	Es cualquier persona que tenga acceso al mismo. Solamente el administrador puede ofrecer o quitar permisos de acceso al sistema.

3.4. Exploración

La metodología de desarrollo Programación Extrema (XP) comienza con la fase de exploración, fase en la que se define el alcance general del proyecto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto, también se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema. La fase de exploración toma de pocas semanas a algunos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

Es en esta fase que el cliente define lo que necesita mediante la redacción de las “historias de usuarios”. Los programadores estiman los tiempos de desarrollo en base a esta información. Debe quedar claro que las estimaciones realizadas en esta fase son primarias, ya que estarán basadas en datos de muy alto nivel, y podrían variar cuando se analicen más en detalle en cada iteración.

3.5. Historias de usuario

Historia de usuario: en la metodología Programación Extrema, es la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente, quien es el mayor responsable de realizar esta tarea, describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario debe ser lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en tiempo delimitado y generalmente poco prolongado. (13)

Clasificación de las historias de usuario:

- Alta: Se le otorga a las Historias de Usuario que resultan funcionalidades fundamentales en el desarrollo del proyecto, a las que el cliente define como principales para el control integral de proyectos.

- Media: Se le otorga a las Historias de Usuario que resultan para el cliente como funcionalidades a tener en cuenta, sin que estas tengan una afectación directa sobre el proyecto que se esté desarrollando.

- Baja: Se le otorga a las Historias de Usuario que constituyen funcionalidades que sirven de ayuda al control de elementos asociados al equipo de desarrollo, a la estructura y no tienen nada que ver con el proyecto en desarrollo.

A continuación se muestran las Historias de Usuario que fueron detectadas para el desarrollo de la solución propuesta y que reflejan la información brindada por el cliente.

Tabla 3.Reproducir flujo de video RTSP.

Historia de Usuario	
Número: 1	**Nombre Historia de Usuario**: Reproducir un flujo de video RTSP
Usuario: Luis Alberto Terrero Ramírez	
Riesgo en Desarrollo: Alto	**Iteración asignada**: 1
Prioridad en negocio: Alta	**Puntos estimados**: 4
Descripción: El usuario selecciona la opción de reproducir, el sistema captura el flujo de video proveniente desde la videocámara de seguridad, obtiene los fotogramas que contiene dicho flujo, dibuja dichos fotogramas y comienza a reproducirlos en forma de video.	
Observaciones:	

Tabla 4. Pintar sobre el video que se está reproduciendo.

Historia de Usuario	
Número: 2	**Nombre Historia de Usuario**: Pintar sobre el video que se está reproduciendo
Usuario: Luis Alberto Terrero Ramírez	
Riesgo en Desarrollo: Alto	**Iteración asignada**: 2

Prioridad en negocio: Media	**Puntos estimados**: 1
Descripción: Cuando el usuario hace clic sobre el componente y arrastra el puntero, el sistema pinta un rectángulo sobre el área que el usuario está seleccionando.	
Observaciones:	

Tabla 5. Capturar una imagen estática perteneciente al video (snapshot).

Historia de Usuario	
Número: 3	**Nombre Historia de Usuario**: Capturar una imagen estática perteneciente al video (snapshot).
Usuario: Luis Alberto Terrero Ramírez	
Riesgo en Desarrollo: Alto	**Iteración asignada**: 2
Prioridad en negocio: Media	**Puntos estimados**: 1
Descripción: Cualquier usuario con acceso al sistema, tendrá la opción de capturar la imagen actual que se esté observando, del video que se está reproduciendo en el componente. Esta imagen puede ser almacenada o no según la elección del usuario.	
Observaciones:	

Tabla 6. Realizar efecto PTZ Digital.

Historia de Usuario	
Número: 5	**Nombre Historia de Usuario**: Realizar efecto PTZ Digital.
Usuario: Luis Alberto Terrero Ramírez	
Riesgo en Desarrollo: Alto	**Iteración asignada**: 2
Prioridad en negocio: Alta	**Puntos estimados**: 4
Descripción: El usuario debe tener la opción de realizar el efecto de PTZ Digital sobre el video que se está reproduciendo. Al seleccionar dicha opción el usuario observará como en el componente se muestra un zoom digital del video que se está reproduciendo y tendrá la opción de cambiar el área que tiene el foco, es decir, el área que se está observando.	
Observaciones:	

Tabla 7. Activar/Desactivar Sonido.

Historia de Usuario

Número: 6	**Nombre Historia de Usuario**: Activar/Desactivar Sonido.
Usuario: Luis Alberto Terrero Ramírez	
Riesgo en Desarrollo: Alto	**Iteración asignada**: 3
Prioridad en negocio: Baja	**Puntos estimados**: 1
Descripción: El usuario puede activar o desactivar el sonido que se está recibiendo en el video IP con solamente seleccionar la opción Mudo.	
Observaciones:	

Tabla 8. Controlar el volumen.

Historia de Usuario	
Número: 6	**Nombre Historia de Usuario**: Controlar el volumen.
Usuario: Luis Alberto Terrero Ramírez	
Riesgo en Desarrollo: Alto	**Iteración asignada**: 3
Prioridad en negocio: Baja	**Puntos estimados**: 1
Descripción: El usuario puede controlar el nivel del volumen con que se está escuchando el video.	
Observaciones:	

3.6. Fase de planificación

La Programación Extrema plantea la planificación como un permanente diálogo entre la parte empresarial y técnica del proyecto, en la que los primeros decidirán el alcance, ¿qué es lo realmente necesario del proyecto?, la prioridad, qué debe ser hecho en primer lugar, la composición de las versiones, qué debería incluir cada una de ellas y la fecha de las mismas.

En cuanto a los técnicos, son los responsables de estimar la duración requerida para implementar las funcionalidades deseadas por el cliente, de informar sobre las consecuencias de determinadas decisiones, de organizar la cultura de trabajo y, finalmente, de realizar la planificación detallada dentro de cada versión. Durante la fase de planificación se realiza una estimación del esfuerzo que costará implementar cada historia de usuario. En XP las métricas son libres, pudiendo utilizarse cualquier criterio para medir el desarrollo del proyecto.

Una métrica popular es la que utiliza como medida el punto. Un punto se considera como una semana ideal de trabajo, donde los miembros de los equipos de desarrollo trabajan el tiempo planeado sin ningún tipo de interrupción. La estimación incluye todo el esfuerzo asociado a la implementación de la historia de usuario. (13)

3.6.1. Estimación de esfuerzo por Historias de Usuarios.

Con el objetivo de alcanzar el buen desarrollo del sistema propuesto, se realizó una estimación de esfuerzo para cada una de las historias de usuario identificadas y se muestran en la tabla a continuación.

Tabla 9.Estimación de esfuerzo por Historias de Usuario.

Historia de usuario	***Puntos de estimación***
Reproducir flujo de video RTSP	4
Pintar sobre el video que se está reproduciendo	2
Capturar una imagen estática perteneciente al video (snapshot).	1
Realizar efecto PTZ Digital	4
Activar/Desactivar Sonido	1
Controlar el volumen	1

3.6.2. Plan de iteraciones

Luego de haberse identificado las historias de usuario del sistema y haberse estimado el esfuerzo que debe ser empleado en la realización de cada una de estas, se procede a la planificación de la etapa de implementación del proyecto. Sobre la base de lo planteado anteriormente se decide realizar tres iteraciones para alcanzar la solución propuesta. Éstas se detallan a continuación:

Iteración 1:

En esta iteración se desarrollará la historia de usuario que más peso tiene sobre el sistema que se desea desarrollar, el punto de partida para la realización de cada una de las restantes historias de usuario que deben ser realizadas. Al concluir el desarrollo de esta iteración se obtendrá una primera y aun incompleta versión del sistema, que ya puede ser probada.

Iteración 2:

Esta iteración tiene como principal objetivo realizar las historias de usuario de prioridad media que son necesarias para la realización de otras historias de usuario de mayor prioridad que serán realizadas en la misma iteración. Al concluir se obtendrá una nueva versión que será el cliente el responsable de revisar y comunicar al equipo de desarrollo los cambios que son necesarios realizarle a la misma.

Iteración 3:

Durante esta iteración se realizarán las restantes historias de usuario de prioridad baja que aún no se han implementado y que son prescindibles para una versión 1.0 del sistema. Esta versión ya puede ser considerada un componente como tal y por tanto debe ser puesto en funcionamiento por un periodo determinado de tiempo y de esta forma ser probado por varios usuarios.

3.6.3. Plan de duración de las iteraciones.

Tabla 10. Plan de duración de las iteraciones.

Iteración	***Orden de las HU[9]***	***Duración de las iteración***
Iteración 1	-Reproducir flujo de video RTSP.	4
Iteración 2	Dibujar sobre el video que se está reproduciendo. Capturar una imagen estática perteneciente al video (snapshot). Realizar efecto PTZ Digital sobre el video.	7
Iteración 3	Activar/Desactivar sonido. Controlar el volumen del sonido.	2

[9] Historias de usuario

Tabla 11. Plan de entregas del sistema propuesto.

No. Iteración	***Duración total***	***Fecha inicio***	***Fecha fin***
Iteración 1	4	20/2/2012	20/3/2012
Iteración 2	6	22/3/2012	1/5/2012
Iteración 3	2	10/5/2012	20/5/2012

3.7. Diseño

Para el diseño de las aplicaciones, se utilizó la metodología XP, esta no requiere la representación del sistema mediante diagramas de clases utilizando notación UML. En su lugar se usan otras técnicas como las tarjetas CRC como una extensión informal a UML. La técnica de representación mediante tarjetas CRC se puede usar para guiar el sistema a través de análisis basados en la responsabilidad. Las clases se examinan, se filtran y se refinan sobre la base de sus responsabilidades con respecto al sistema, y las clases con las que necesitan colaborar para completar las mismas (Wake, 1999).

3.7.1. Tarjetas CRC

Tabla 12. Tarjeta CRC NPlayer

NPlayer	
Superclases: QFrame, IPlayerWidget	
Subclases:	
Responsabilidades	***Colaboraciones***
NPlayer(QWidget* parent = 0) bool openVideoUrl(const QString& videoUrl) void play() void pause() void unPause() void togglePause() void stop() void setScaleMode(Qt::AspectRatioMode scaleMode = Qt::KeepAspectRatio) virtual QSize sizeHint() void setFillBackground(bool backgroundFilled = true) void* lockCB(void** pixelPlane) void unlockCB(void* picture, void*const *pixelPlane) void displayCB(void* picture) static void* lockCBWrapper(void* widget, void** pixelPlane) static void unlockCBWrapper(void* widget, void* picture, void*const *pixelPlane)	libvlc_instance_t QImage QMutex QSize QRect QFrame QResizeEvent

static void displayCBWrapper(void* widget, void* picture) virtual void paintEvent(QPaintEvent *e) QSize videoSourceSize() void videoResizeEvent(GLResizeEvent* event) void displaySizeHelper() void resizeEvent(QResizeEvent *e) void activateZoom() void mousePressEvent(QMouseEvent *event) void mouseMoveEvent(QMouseEvent *event) void mouseReleaseEvent(QMouseEvent *event) bool event(QEvent *e)	

Tabla 13 Tarjeta CRC QPlayer

QPlayer	
Superclases: QGLWidget, IPlayerWidget	
Subclases:	
Responsabilidades	***Colaboraciones***
NPlayer(QWidget* parent = 0) bool openVideoUrl(const QString& videoUrl) void play() void pause() void unPause() void togglePause() void stop() void setScaleMode(Qt::AspectRatioMode scaleMode = Qt::KeepAspectRatio) virtual QSize sizeHint() void setFillBackground(bool backgroundFilled = true) void* lockCB(void** pixelPlane) void unlockCB(void* picture, void*const *pixelPlane) void displayCB(void* picture) static void* lockCBWrapper(void* widget, void** pixelPlane) static void unlockCBWrapper(void* widget, void* picture, void*const *pixelPlane) static void displayCBWrapper(void* widget, void* picture) virtual void paintEvent(QPaintEvent *e) QSize videoSourceSize() void videoResizeEvent(GLResizeEvent* event) void displaySizeHelper() void resizeEvent(QResizeEvent *e) void activateZoom() void mousePressEvent(QMouseEvent *event) void mouseMoveEvent(QMouseEvent *event) void mouseReleaseEvent(QMouseEvent *event) bool event(QEvent *e)	libvlc_instance_t QImage QMutex QSize QRect QFrame QResizeEvent

Tabla 14 Tarjeta CRC IPlayerWidget

IPlayerWidget	
Superclases:	
Subclases: NPlayer, QPlayer	
Responsabilidades	***Colaboraciones***
void play() void pause() void unPause() void togglePause() void stop() QSharedPointer<QImage> pan_tilt (const QImage& original,const QRect& rect = QRect()) QSharedPointer<QImage> zoom(const QImage& original,const QRect& rect = QRect()) void activateZoom() void Mute() void setpanning()	QRect libvlc_media_player_t QWidget

Tabla 15 Tarjeta CRC GLResizeEvent

GLResizeEvent	
Superclases: QEvent	
Subclases:	
Responsabilidades	***Colaboraciones***
GLResizeEvent (const QSize& newSize, const QSize& oldSize) QSize newSize() QSize oldSize()	QSize

Tabla 16 Tarjeta CRC MainWindow

MainWindow (Form)	
Superclases: QMainWindow, Ui::MainWindow	
Subclases:	
Responsabilidades	***Colaboraciones***
MainWindow(QWidget *parent = 0) void on_actionOpen_triggered() void on_actionMudo_triggered() void on_actionZoom_triggered() void on_actionPan_Tilt_triggered()	IPlayerWidget

No obstante el uso de los diagramas de clases UML puede aplicarse siempre y cuando mejore la comunicación, no sea un peso su mantenimiento, no sean extensos y se enfoquen en la información importante.

3.8. Patrones de diseño

Para organizar el desarrollo del componente se aplicaron diversos patrones de diseño, los cuales tiene como objetivo mantener guiado el desarrollo de un sistema de software. A continuación se describen los mismos:

***Patrones GRASP*:**

- *Experto*: Este patrón, tiene como objetivo que la responsabilidad de realizar una labor sea de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. Si se realiza bien la aplicación de este, los sistemas tienden a ser más fáciles de entender, mantener y ampliar, y existen más oportunidades para reutilizar componentes en futuras aplicaciones. Este patrón es utilizado en la aplicación cuando se está dando todas las funcionalidades a las clases que se crean. Ejemplo de esto lo demuestra en la *IPlayerWidget* que tiene un método que es *setZoomPos* que lo realiza porque cuenta con todos los parámetros que necesita para ello.

- *Creador*: Guía la asignación de responsabilidades relacionadas con la creación de objetos, una tarea muy común. La intención básica del patrón Creador es encontrar un creador que necesite conectarse al objeto creado en alguna situación. Eligiéndolo como el creador se favorece el bajo acoplamiento. Este se utiliza cuando se crea algún objeto que sea o no de la clase en la que se quiere crear, pues va a contar con todos los valores posibles para crear dicho objeto, así como para poder utilizarlo.

- *Bajo Acoplamiento*: Debe haber pocas dependencias entre las clases. Si todas las clases dependen de todas no se puede extraer software de forma independiente y reutilizarlo. Este patrón es un principio que asigna la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Esto facilita la centralización de las actividades. (19)

3.9. Conclusiones

Se concluye que XP como metodología de desarrollo ágil se adapta casi perfectamente a las condiciones existentes para el desarrollo del componente ya que no es un sistema de software de gran

tamaño, el equipo de desarrollo estaba compuesto únicamente por una persona y había además constante comunicación con el cliente. Se determinó que la utilización de C++ como lenguaje de programación en el entorno de desarrollo Qt facilitaría la integración con la versión actual del sistema Suria Vision que se ha desarrollado en su totalidad utilizando estas dos herramientas. Por su parte el uso de Qt como framework para el desarrollo, ofrece funcionalidades que facilitarán el pintado con un nivel medio de rendimiento para las imágenes que se obtendrán del video IP.

4. IMPLEMENTACIÓN

4.1. Introducción

En este capítulo se explicarán los estándares para la codificación utilizados durante el desarrollo del software. También se explicará cómo se planificó cada una de las tareas necesarias para darle solución a los objetivos propuestos en la introducción a la investigación.

4.2. Estándares de codificación

Nombres:

- Escoger nombres lo suficientemente largos que expresen correctamente el sentido de lo que se quiere, pero evitando manejar nombres que dificulten la labor de implementación.
- Evitar nombres que permitan una interpretación subjetiva de esta forma se evita la ambigüedad y se asegura abstracción.
- Limitar el uso de nombres cortos para las variables cuando se encuentren dentro de funcionalidades pequeñas, donde esté claro su significado.
- Minimizar el uso de abreviaturas. En caso de ser requeridas, debe ser consistente en su uso y cada abreviatura debe significar solo una cosa.
- Los nombres de los métodos son frases que incluyen verbos.
- Los nombres de los atributos y parámetros son frases con sustantivos.
- Evitar redundancia no repitiendo nombres de clases en sus elementos.
- Dado que los nombres generalmente son el producto de concatenar varias palabras, se debe emplear la primera palabra en minúscula, mayúscula para denotar la letra de inicio de cada una de las palabras restantes por las que esté formado y minúscula para las letras intermedias en el caso de los nombres de métodos y funciones.
- Para el caso de los nombres de variables y atributos debe aplicarse la misma convención.
- Los nombres de las clases son sustantivos singulares.
- Los nombres de clases y objetos deben reflejar qué hacen y no cómo lo hacen.
- Los nombres de constantes deben contener solo letras mayúsculas.

Codificación:

- Emplear correctamente los tipos de ciclos: si es al menos una vez usar do-while, si es ninguna o más veces usar while-do, y si se conoce el número exacto de ciclos usar for.

- Inicializar todas las variables.
- Emplear líneas en blanco para separar pasos lógicos.
- Emplear líneas en blanco para organizar el código, permitiendo crear párrafos de código para una mejor lectura.
- Evitar colocar más de una sentencia por línea.
- Emplear constantes en sustitución de números o cadenas de caracteres literales.
- Alinear secciones del código.
- Alinear verticalmente llaves de apertura y cierre.
- Usar espacios antes y después de los operadores que el lenguaje de programación permita.
- Minimizar el alcance de las variables para evitar confusión y facilitar el mantenimiento.
- Emplear cada variable y rutina solo para un propósito.
- Emplear las letras i, j, k, l, m, p, q, r para contadores en ciclos.
- Mantener la modularidad del código bajo el criterio de la lógica que encierra, no exagerar la modularidad.

4.3. Tareas de la ingeniería para dar solución a las historias de usuario.

Durante el inicio de cada iteración, se lleva a cabo una revisión del plan de iteraciones y se modifica de ser necesario. Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable. Estas tareas, pueden escribirse utilizando un lenguaje técnico y no necesariamente deben ser entendibles para el cliente. (13)

4.3.1. Primera iteración.

Tabla 17. Tarea 1-Reproducir flujo de video RTSP.

Tarea	
Numero de tarea: 1	**Número de historia:** 1
Nombre de tarea: Reproducir flujo de video RTSP	
Tipo de tarea: Desarrollo	**Puntos estimados:** 4
Fecha de inicio: 20/2/2012	**Fecha de fin:** 20/3/2012
Programador responsable: Luis Alberto Terrero Ramirez	
Descripción: Esta funcionalidad será implementada para capturar los fotogramas que vienen en el flujo de video IP, estos fotogramas luego serán dibujados y reproducidos secuencialmente en forma de video.	

4.3.2. Segunda iteración

Tabla 18. Tarea 2-Dibujar sobre el video que se está reproduciendo

Tarea	
Numero de tarea: 2	**Número de historia:** 2
Nombre de tarea: Pintar sobre el video que se está reproduciendo	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 21/3/2012	**Fecha de fin:** 28/3/2012
Programador responsable: Luis Alberto Terrero Ramirez	
Descripción: Esta funcionalidad será implementada con el objetivo de que el cliente pueda seleccionar un área determinada sobre el video que se está reproduciendo, mostrándole con un rectángulo pintado sobre el video el área que seleccionó.	

Tabla 19 Tarea 3. Capturar una imagen estática perteneciente al video.

Tarea	
Numero de tarea: 3	**Número de historia:** 3
Nombre de tarea: Capturar imagen estática	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 21/3/2012	**Fecha de fin:** 28/3/2012
Programador responsable: Luis Alberto Terrero Ramirez	
Descripción: Esta funcionalidad será implementada con el objetivo de que el usuario pueda almacenar un fotograma en forma de imagen, del video que se está reproduciendo.	

Tabla 20 Tarea 4. Realizar efecto PTZ Digital sobre el video.

Tarea	
Numero de tarea: 4	**Número de historia:** 4
Nombre de tarea: Realizar PTZ Digital sobre el video	
Tipo de tarea: Desarrollo	**Puntos estimados:** 4
Fecha de inicio: 28/3/2012	**Fecha de fin:** 30/04/2012
Programador responsable: Luis Alberto Terrero Ramirez	
Descripción: Esta funcionalidad tiene como objetivo de implementación que el usuario pueda realizar zoom y movimientos horizontales, y verticales dentro del video que se está reproduciendo.	

4.3.3. Tercera iteración

Tabla 21 Tarea 5. Activar/Desactivar sonido.

Tarea	
Numero de tarea: 5	**Número de historia:** 5
Nombre de tarea: Activar/Desactivar sonido.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 1/5/2012	**Fecha de fin:** 7/5/2012
Programador responsable: Luis Alberto Terrero Ramirez	
Descripción: Esta funcionalidad será implementada para brindarle al usuario la posibilidad de escuchar o no el audio incluido en el flujo de media.	

Tabla 22 Tarea 6. Controlar el volumen del sonido.

Tarea	
Numero de tarea: 6	**Número de historia:** 6
Nombre de tarea: Controlar el volumen del sonido.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 1/5/2012	**Fecha de fin:** 7/5/2012
Programador responsable: Luis Alberto Terrero Ramirez	
Descripción: Esta funcionalidad será implementada para brindarle al usuario la posibilidad de controlar el volumen con que se está escuchando el archivo de media.	

4.4. Fase de Prueba

La Programación Extrema tiene entre sus mayores pilares las pruebas de software. XP incita a probar tanto y con la mayor frecuencia posible, es por ello que se obtiene gran calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. Permite además aumentar la seguridad y evitar efectos colaterales no deseados a la hora de realizar modificaciones a las versiones de software existentes.

Las pruebas realizadas por XP se dividen en dos grupos: pruebas unitarias que son las encargadas de verificar el código, son diseñadas por los programadores, y las pruebas de aceptación o pruebas funcionales como también suele llamárseles, destinadas a evaluar si al final de cada una de las iteraciones se consiguió la funcionalidad requerida diseñadas por el cliente final.

Siempre que se finaliza una o varias historias de usuario, se presenta el software debidamente probado al cliente, quien es el encargado de realizar las pruebas de aceptación a la(s) funcionalidad(es) implementada(s). Cuando se reúnen un número suficiente de funcionalidades que representan una versión útil y parcialmente completa de la aplicación se produce una liberación de software, o sea se convierte en una versión funcional de la aplicación que aporta valor al negocio y que debe ser mantenida mientras se desarrollan las próximas funcionalidades.

A continuación se explica detalladamente las pruebas que le fueron realizadas al componente de visualización.

4.4.1. Pruebas unitarias

Durante el proceso de implementación de un sistema, cada desarrollador esta en la obligación de ir probando constantemente el resultado que va obteniendo en la funcionalidad exigida por el cliente. Estas son las pruebas conocidas como Unitarias, estas no generan artefactos y no son directamente palpables para el cliente pero son de mucha importancia para el desarrollo satisfactorio de un proyecto.

4.4.2. Pruebas de aceptación

Las pruebas de aceptación son las realizadas por el cliente y usuarios finales de la aplicación. En estas serán probadas las funcionalidades exigidas por el cliente y descritas en las historias de usuario, además de los aspectos de seguridad requeridos. Luego de haber superado las pruebas de aceptación podrá considerarse que la aplicación es apta para el uso y despliegue dentro del proyecto.

Como resultado de las pruebas de aceptación se obtendrán artefactos descritos en tablas, estas contarán con los siguientes campos:

- Código: servirá como identificador de la prueba realizada, a su vez será sugerente al nombre de la prueba a la que hace referencia. UH: tendrá el nombre de la historia de usuario a la que hace referencia la prueba a realizar.
- Nombre: nombre que se le da a la prueba a realizar.
- Descripción: se describe la funcionalidad que se desea probar.

- Condiciones de Ejecución: mostrará las condiciones que deben cumplirse para poder llevar a cabo el caso de prueba.
- Entradas / Pasos de Ejecución: se hará la descripción de cada uno de los pasos seguidos durante el desarrollo de la prueba, se tendrá en cuenta cada una de las entradas que hace el usuario con el objetivo de ver si se obtiene el resultado esperado.
- Resultado esperado: se hará una breve descripción del resultado que se espera obtener con la prueba realizada.
- Evaluación de la prueba: acorde al resultado de la prueba realizada se emitirá una evaluación sobre la misma. Esta evaluación tendrá uno de los tres resultados siguientes: Bien, Parcialmente Bien y Mal.

A continuación se muestran las pruebas de aceptación que le fueron realizadas al sistema:

Tabla 23. Caso de prueba de aceptación HU- Reproducir flujo de video RTSP.

Caso de Prueba de Aceptación	
Código: 1	**Historia de Usuario:** 1
Nombre: Reproducir flujo de video RTSP.	
Descripción: Prueba para verificar la correcta reproducción de un flujo de video RTSP.	
Condiciones de ejecución: Cualquier usuario puede comenzar la reproducción de un flujo de video.	
Entrada/Pasos de ejecución: Se introduce la dirección utilizando el protocolo “rtsp” a reproducir y se intenta comenzar la reproducción.	
Resultado esperado: Se comienza la reproducción del flujo de media mostrándole al usuario el video IP que está recibiendo desde la video cámara de seguridad.	
Evaluación de la prueba: Bien	

Tabla 24. Caso de prueba de aceptación HU-Pintar sobre el video que se está reproduciendo.

Caso de Prueba de Aceptación	
Código: 2	**Historia de Usuario:** 2
Nombre: Pintar sobre el video que se está reproduciendo.	
Descripción: Prueba para verificar la funcionalidad “Pintar sobre el video que se está reproduciendo”.	

Condiciones de ejecución: Cualquier usuario puede pintar sobre un área determinada del video que se está reproduciendo.
Entrada/Pasos de ejecución: Se hace clic sobre un área del video y se arrastra el cursor intentando pintar un área en forma de rectángulo.
Resultado esperado: Se observa como dicha área queda encerrada por una figura geométrica en forma de rectángulo que se va pintando sobre el video mientras se arrastra el cursor dentro del mismo.
Evaluación de la prueba: Bien

Tabla 25. Caso de prueba de aceptación HU-Capturar una imagen estática perteneciente al video.

Caso de Prueba de Aceptación	
Código: 3	**Historia de Usuario: 3**
Nombre: Capturar una imagen estática perteneciente al video.	
Descripción: Prueba para verificar la funcionalidad “Capturar una imagen estática perteneciente al video”.	
Condiciones de ejecución: Cualquier usuario puede capturar una imagen estática perteneciente al video que se encuentra en reproducción.	
Entrada/Pasos de ejecución: Se selecciona del menú la opción “Snapshot” y se selecciona en la ventana de diálogo que aparece la ubicación que se desea para almacenar la imagen extraída desde el video.	
Resultado esperado: Se almacena la imagen con el nombre, formato y dirección que el usuario seleccionó.	
Evaluación de la prueba: Bien	

Tabla 26. Caso de prueba de aceptación HU- Realizar efecto PTZ Digital sobre el video.

Caso de Prueba de Aceptación	
Código: 4	**Historia de Usuario:** 4
Nombre: Realizar efecto PTZ Digital sobre el video.	
Descripción: Prueba para verificar la funcionalidad “Realizar efecto PTZ Digital sobre el video”.	
Condiciones de ejecución: Todo usuario puede realizar el efecto de PTZ Digital sobre el video que se está reproduciendo.	
Entrada/Pasos de ejecución: Se selecciona del menú la opción “Zoom” o “Pan-Tilt”	

Resultado esperado: El sistema obtiene un rectángulo de la esquina superior izquierda del video y lo muestra al usuario, brindándole además la opción de agrandar un área de la imagen digitalmente y cambiar el área que tiene el foco, es decir moverse dentro del video.
Evaluación de la prueba: Parcialmente bien

Tabla 27. Caso de prueba de aceptación HU-Activar/Desactivar sonido.

Caso de Prueba de Aceptación	
Código: 5	**Historia de Usuario:** 5
Nombre: Activar/Desactivar sonido.	
Descripción: Prueba para verificar si el sistema es capaz de desactivar el sonido que se está reproduciendo.	
Condiciones de ejecución: Todo usuario activar o desactivar el sonido que se está recibiendo de conjunto con el video.	
Entrada/Pasos de ejecución: Se selecciona del menú la opción “Mudo”	
Resultado esperado: El sistema silencia el video en caso de estarse escuchando, en caso contario lo reactiva.	
Evaluación de la prueba: Bien	

Tabla 28. Caso de prueba de aceptación HU-Controlar el volumen del sonido.

Caso de Prueba de Aceptación	
Código: 6	**Historia de Usuario:** 6
Nombre: Controlar el volumen del sonido.	
Descripción: Prueba para verificar la funcionalidad “Controlar el volumen del sonido”.	
Condiciones de ejecución: Todo usuario puede alzar o disminuir el volumen del sonido que se está escuchando sincronizado al video.	
Entrada/Pasos de ejecución: Se selecciona del menú la opción “Incrementar volumen” o “Disminuir volumen”.	
Resultado esperado: El sistema incrementa y disminuye, según la opción seleccionada por el usuario, el volumen del sonido.	
Evaluación de la prueba: Bien	

4.5. Conclusiones

Se concluye que con la implementación de las tareas de ingeniería propuestas se da solución a cada una de las historias de usuario definidas durante la etapa de diseño. Por otra parte el uso de los estándares de codificación propuestos organizarán el código y lo harán mucho más entendible para su futura reutilización o modificación. Se comprobó además mediante las pruebas aplicadas y expuestas en el capítulo, la efectividad de la realización de cada una de las tareas de ingeniería, dando lugar a un componente completamente funcional.

CONCLUSIONES GENERALES

Con la realización del presente trabajo se ha dado cumplimiento al objetivo general propuesto para esta investigación, se concluye además que:

- Luego del estudio de las diferentes bibliotecas libres de código abierto para el trabajo con multimedia, la más factible para dar solución a la problemática planteada es la libvlc, ya que brinda las funcionalidades necesarias para capturar los fotogramas del video IP que se está reproduciendo y utilizarlos para satisfacer cada una de las historias de usuario definidas en la investigación.
- Se determinó que XP como metodología de desarrollo ágil se adapta casi perfectamente a las condiciones necesarias para el desarrollo del componente, ya que no se iba a desarrollar un sistema de software de gran tamaño, el equipo de desarrollo estaba compuesto únicamente por una persona y había, además, constante comunicación con el cliente
- Se determinó que la utilización de C++ como lenguaje de programación en el entorno de desarrollo Qt Creator, facilitaría la integración con la versión actual del sistema Suria Vision que ha sido desarrollado en su totalidad utilizando estas dos herramientas. Por su parte el uso de Qt como framework para el desarrollo ofrece funcionalidades que facilitaron el pintado con un nivel medio de rendimiento para las imágenes que se capturan del flujo de video IP.
- Fueron realizadas, exitosamente, las tareas definidas para llevar a cabo cada una de las historias de usuario, evidenciándose su efectividad en las pruebas de aceptación que le fueron aplicadas al sistema.

RECOMENDACIONES

A continuación se listan algunos aspectos que se deberían tener en cuenta para posibles futuras versiones del sistema para la reproducción de video:

- Profundizar en la investigación de los componentes de reproducción de video para los sistemas de video vigilancia para incluirle funcionalidades como: seguimiento de objetos dentro del video y "congelar" una zona dentro del video que se está reproduciendo.
- Mejorar la implementación del proceso de pintado del video, para obtener mejor calidad en la imagen que se está observando.
- Utilizar algoritmos para la reconstrucción de imágenes, en busca de mejorar la calidad visual de las imágenes que se almacenan al hacer uso de la funcionalidad "*Snapshot".*

REFERENCIAS BIBLIOGRÁFICAS

1. **Edmis Deyvis Semanat Aldana, Leonor Verdecia Four.** *Sistema de video vigilancia.* La Habana : s.n., 2009.
2. **Colomer, Antonio Albior.** *Seguimiento de objeto en secuencia de video.* España : s.n., 2003.
3. Electronic Dreams. Cámara IP. [En línea] [Citado el: 23 de febrero de 2012.] http://www.camara-ip.es.
4. **Jonás A. Montilva C., Nelson Arapé, Juan Andrés Colmenares.** *Desarrollo de Software basado en Componentes.* Venezuela : s.n.
5. VideoLan - Free Open Software and Open Source video streaming for every OS. [En línea] Video Lan Project. [Citado el: 7 de octubre de 2011.] http://www.videolan.org/videolan. 1.
6. MPlayer - The Movie Player. [En línea] MPlayer. [Citado el: 22 de octubre de 2011.] http://www.mplayerhq.hu/design7/news.html. 4.
7. FFmpeg Documentation. [En línea] [Citado el: 25 de febrero de 2012.] http://ffmpeg.org/ffmpeg.html.
8. **Roque, Serguey Corvo.** *Interfaz de comunicación con la librería libvlc para las aplicaciones de reproducción y transmisión de media del departamento de Señales Digitales.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2010. Tesis de Diploma. 5.
9. Axis Communications - Métodos de transporte de datos. [En línea] [Citado el: 25 de febrero de 2012.] http://www.axis.com/es/products/video/about_networkvideo/data_transport_methods.htm.
10. Glosario de términos para cámaras digitales. [En línea] [Citado el: 5 de noviembre de 2011.] http://www.shopmania.es. 8.
11. Entendiendo mejor la relación de aspecto y la resolución de una pantalla. [En línea] Walhez. [Citado el: 7 de noviembre de 2011.] http://walhez.com/2008/12/entendiendo-mejor-la-relacion-de-aspecto-y-la-resolucion-de-una-pantalla. 9.
12. Metodologías de Desarrollo. [En línea] [Citado el: 20 de enero de 2012.] http://www.marblestation.com/?p=644. 12.
13. **Beck, Kent.** *Extreme Programming Explained: Embrace Change, Addison–Wesley.*
14. **Stephens, Matt y Rosenberg, Doug.** Extreme Programming Refactored. [En línea] [Citado el: 5 de febrero de 2012.] http://www.softwarereality.com/ExtremeProgrammingRefactored.jsp.
15. Historia del computador y evolución, arquitectura, lenguajes de programación y algoritmos. [En línea] [Citado el: 15 de febrero de 2012.] http://nectacellcomputer.jimdo.com/historia-del-computador-arquitectura-.

16. Lenguaje de programación. [En línea] [Citado el: 25 de enero de 2012.] http://www.scribd.com/doc/13719562/Lenguaje-de-Programacion. 15.
17. Ingenieria de Software I. [En línea] [Citado el: 2012 de enero de 25.] http://www.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE.
18. Visual Paradigm for UML. [En línea] [Citado el: 25 de enero de 2012.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.
19. **Hall, Larman - Prentice.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.*
20. Acerca de la transmisión de unidifusión. [En línea] Microsoft Corporation, 2007. [Citado el: 28 de octubre de 2011.] http://technet.microsoft.com/es-es/library/cc772130%28WS.10%29.aspx. 6.
21. Español - The KMPlayer´s Forum. [En línea] Pandora TV. [Citado el: 15 de octubre de 2011.] http://www.kmplayer.com/forums. 2.
22. Microsoft Corporation. [En línea] Microsoft Corporation. [Citado el: 18 de octubre de 2011.] http://www.microsoft.com/en-US. 3.
23. Enrutamiento unicast. [En línea] Verisign. [Citado el: 2 de noviembre de 2011.] http://www.verisigninc.com/es_ES/products-and-services/network-intelligence-availability/managed-dns/dns-query-routing/unicast/index.xhtml. 7.
24. *Servicios de vídeo sobre redes móviles de nueva generación.* **Bernárdez, Rosa Maria, y otros, y otros.** 10, 2001, Comunicaciones Teléfonica I+D.
25. Real Time Streaming Protocol. [En línea] [Citado el: 11 de noviembre de 2011.] http://www.ietf.org/rfc/rfc2326.txt. 11.
26. **Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides - Addison Wesley.** *Design Patterns. Elements of Reusable Object-Oriented Software.*
27. **Sánchez, Frank Emilio Hernández.** *Manejador de datos del servidor de streaming distribuido Alffrey´s.* Universidad de las Ciencias Informáticas. La Habana : s.n., 2011. Tesis de diploma. 13.

BIBLIOGRAFÍA

1. Servicios de vídeo sobre redes móviles de nueva generación. Bernárdez, Rosa Maria, y otros. 1, 2001, Comunicaciones Teléfonica I+D.

2. Real Time Streaming Protocol. Network Working Group. Request for Comments: 2326. [En Linea]. Disponible en: http://www.ietf.org/rfc/rfc2326.txt

3. Transmisión de Audio e Imagen en Tiempo Real a través de Internet. [En Línea]. Disponible en: http://futura.disca.upv.es/~radiotv/sagra/videos_de_sagra/pagina/Memoria%20Radiotv.htm

4. Reserva Eficiente de Recursos en Redes para Transmisión en Tiempo Real. Enrique Hernández Orallo. Departamento de Informática de Sistemas y Computadores. Universidad Politécnica Valencia. Valencia-España, 2.001.

5. Video Streaming – En directo, [En línea]. Disponible en: http://www.webstudio.es/streaming/video-directo.html

6. Streaming video hosting services, [En línea], Disponible en http://www.streamingvideoprovider.co.uk/

7. VideoLan – Streaming Video Features, [En línea]. Disponible en http://www.videolan.org/streaming-features.html

8. Society of Motion Picture and Television Engineers (SMPTE). [En línea]. Disponible en: www.smpte.org

9. Intellectual Properties. [En Línea]. Disponible en: http://wiki.videolan.org/Intellectual_Properties

10. VLC playback features. [En Línea]. Disponible en: http://www.videolan.org/vlc/features.html

11. MPlayer – The Movie Player. [En línea]. Disponible en: http://www.mplayerhq.hu

12. The KMPlayer´s forum. [En Linea]. Disponible en : http://www.kmplayer.com/fórums

13. Windows Media – Microsoft Windows. [En línea]. Disponible en: http://windows.microsoft.com/en-US/windows/products/windows-media

14. Windows Media Technical Articles Archive. [En Línea]. Disponible en: http://www.microsoft.com/windows/windowsmedia/knowledgecenter/technicalarticles.aspx#mediaandentertainment

15. Ciudad Ricardo, Febe Angel. ¿CÓMO CONFECCIONAR UN ESTADO DEL ARTE? Ciudad de la Habana : s.n., 2008.

16. VideoLan - Free Open Software and Open Source video streaming for every OS. [En línea] Video Lan Project. [Citado el: 7 de octubre de 2011.] http://www.videolan.org/videolan.

17. Español - The KMPlayer´s Forum. [En línea] Pandora TV. [Citado el: 15 de octubre de 2011.] http://www.kmplayer.com/forums.

18. Microsoft Corporation. [En línea] Microsoft Corporation. [Citado el: 18 de octubre de 2011.] http://www.microsoft.com/en-US.

19. MPlayer - The Movie Player. [En línea] MPlayer. [Citado el: 22 de octubre de 2011.] http://www.mplayerhq.hu/design7/news.html.

20. Roque, Serguey Corvo. Interfaz de comunicación con la biblioteca libvlc para las aplicaciones de reproducción y transmisión de media del departamento de Señales Digitales. Universidad de las Ciencias Informáticas. La Habana : s.n., 2010. Tesis de Diploma.

21. Acerca de la transmisión de unidifusión. [En línea] Microsoft Corporation, 2007. [Citado el: 28 de octubre de 2011.] http://technet.microsoft.com/es-es/library/cc772130%28WS.10%29.aspx.

22. Enrutamiento unicast. [En línea] Verisign. [Citado el: 2 de noviembre de 2011.] http://www.verisigninc.com/es_ES/products-and-services/network-intelligence-availability/managed-dns/dns-query-routing/unicast/index.xhtml.

23. Glosario de términos para cámaras digitales. [En línea] [Citado el: 5 de noviembre de 2011.] http://www.shopmania.es

24. Entendiendo mejor la relación de aspecto y la resolución de una pantalla. [En línea] Walhez. [Citado el: 7 de noviembre de 2011.] http://walhez.com/2008/12/entendiendo-mejor-la-relacion-de-aspecto-y-la-resolucion-de-una-pantalla.

25. Servicios de vídeo sobre redes móviles de nueva generación. Bernárdez, Rosa Maria, y otros, y otros. 10, 2001, Comunicaciones Teléfonica I+D.

26. Real Time Streaming Protocol. [En línea] [Citado el: 11 de noviembre de 2011.] http://www.ietf.org/rfc/rfc2326.txt.

27. Metodologías de Desarrollo. [En línea] [Citado el: 20 de enero de 2012.] http://www.marblestation.com/?p=644.

28. Beck, Kent. Extreme Programming Explained: Embrace Change, Addison–Wesley.

29. Historia del computador y evolución, arquitectura, lenguajes de programación y algoritmos. [En línea] [Citado el: 15 de febrero de 2012.] http://nectacellcomputer.jimdo.com/historia-del-computador-arquitectura-.

30. Lenguaje de programación. [En línea] [Citado el: 25 de enero de 2012.] http://www.scribd.com/doc/13719562/Lenguaje-de-Programacion.

31. Ingenieria de Software I. [En línea] [Citado el: 2012 de enero de 25.] http://www.scribd.com/doc/3062020/Capitulo-I-HERRAMIENTAS-CASE.

32. Visual Paradigm for UML. [En línea] [Citado el: 25 de enero de 2012.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

GLOSARIO DE TÉRMINOS

PTZ: Es al acrónimo de movimiento horizontal, vertical y realizar efecto zoom, por sus siglas en inglés (pan, tilt, zoom) que se le son aplicados generalmente a una videocámara de seguridad.

PTZ Digital: Este término será utilizado a lo largo de la investigación para referirse a la operación de moverse horizontal o verticalmente y realizar zoom digital sobre un área específica de un cuadro de video.

Reproducción avanzada: Hecho de reproducir un contenido multimedia y realizar sobre el mismo el efecto PTZ Digital:

Video IP: Contenido multimedia (audio y video) que es transmitido a través de redes bajo el protocolo TCP/IP.

Video sensor: es una herramienta de análisis de video digital que ofrece información significativa proveniente de una secuencia de video. El desarrollo de video sensores tiene su base en el procesamiento digital de la imagen.

Cámara IP: Puede describirse como una cámara y un ordenador combinados para formar una única unidad inteligente. Captura y envía video en vivo, a través de una red IP, como una LAN, Intranet o Internet, permite a los usuarios ver o gestionar la cámara con un navegador Web estándar o con software de gestión de video en cualquier equipo local o remoto conectado a una red. Permite a usuarios autorizados de distintas ubicaciones acceder simultáneamente a las imágenes captadas por la misma cámara IP de red.