Universidad de las Ciencias Informáticas

Facultad 6

Título:

Componente para elaboración de resúmenes de video.

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

Autor: Ronald Alfonso Sentmanat

Tutor: M.Sc. Abel Díaz Berenguer

La Habana, Junio 2015.

"Año 57 de la Revolución"

"Hay una fuerza motriz más poderosa que el vapor, la electricidad y la energía atómica: la voluntad."

Albert Einstein

Declaración de autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma con carácter exclusivo; para hacer uso de la misma en su beneficio.

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

_____ _____

Autor: Ronald Alfonso Sentmanat. **Tutor:** M.Sc. Abel Díaz Berenguer.

Datos de contacto

Tutor: M.Sc. Abel Díaz Berenguer

Categoría Científica: Master en Informática Aplicada.

Centro Laboral: Universidad de las Ciencias Informáticas (UCI).

Cargo: Subdirector del centro Geoinformática y Señales Digitales

Correo: aberenguer@uci.cu.

Dedicatoria

A mis padres y mi madrina, como los principales promotores de mi superación, especialmente a mi madre por estar en todo momento disponible para ayudarme.

A mi mujer por convertirse en solo 2 años en la razón por la que me levanto en las mañanas.

A "El Sergo" por ser el hermano que nunca tuve durante los últimos cinco años.

A todas las personas que me brindaron su apoyo incondicional.

Agradecimientos

Agradezco a mi mamá, por ayudarme durante todos estos años a estudiar y formar mi pensamiento, y por su preocupación y cariño sin medidas.

A mi papá por, impulsarme a superarme en todo momento y por ser el promotor de muchos de mis valores.

A mi madrina que desde la distancia me ha ayudado tanto que no tengo palabras para agradecerle.

Un agradecimiento especial a mi mujer Daimé, por el cariño incondicional que me brinda, por su incansable deseo de hacer lo correcto, por abrirme caminos que siempre pensé bloqueados para mí y por su ayuda en todos aspectos de mi vida.

A Serguey, por ser el hermano con el que siempre he podido contar y por nuestras conversaciones que hacen tolerable el día a día.

A mis amigos y compañeros de la UCI, en especial para Yoan, Reidel, Eduardo, Rolando, Yulina, Elibetsy, Noslen, Daldis, Yaneisy, Pedro, Arian, Marquito, Alexis y para Mike dondequiera que esté en estos momentos.

A mi tutor, por guiarme durante todo el proceso de tesis.

A todos los profesores que de alguna forma son responsables de que yo me haya graduado de Ingeniero en Ciencias Informáticas.

A todos aquellos que me ayudaron a lograr que este día fuera posible.

Y a la UCI, por brindarme una experiencia inolvidable.

Resumen

El gran volumen de información audiovisual existente dificulta su gestión. Surgen los resúmenes de video con el objetivo de obtener representaciones sintetizadas de los mismos para lograr cierta perspectiva de un archivo de video sin visualizar su contenido completamente. Dichos resúmenes de videos pueden ser representados de forma dinámica, como una colección de secuencias extraídas del video original, o bien, estática, como una recopilación de imágenes fijas.

La presente investigación propone la creación de un componente para la elaboración automática de resúmenes escalables de video fácilmente adaptables a una duración conforme a las necesidades de cada usuario. Para dar cumplimiento a este objetivo, el componente implementado garantiza el procesamiento del video siguiendo tres pasos: segmentación del video en tomas, selección de los segmentos a incluir en el resumen, unión de los segmentos. Durante la segmentación se procesa el video mediante histogramas de color en el espacio HSV, descriptores de puntos de interés y contornos de la imagen. Luego en la selección se obtiene una lista organizada, de las tomas que componen el video original, por criterios de significación-impacto de la cual se eliminan las tomas que poseen contenido muy similar. Se guardan los resultados de la etapa de análisis en una base de datos externa, donde los mismos pueden ser obtenidos en el momento deseado para la generación de los resúmenes escalables, considerando la duración requerida.

El componente obtenido presenta buenos resultados para las diferentes codificaciones de video probadas, así mismo permite generar resúmenes escalables que brindan la cobertura necesaria para determinar el contenido del video original.

Palabras Clave: componente, descriptores, escalable, resumen, video.

Abstract

The large volume of existing audiovisual information difficult it's manage. Video summaries arise in order to obtain synthesized representations of video sequences, to gain some perspective of a video file without displaying its full content. These video summaries can be represented dynamically, as a collection of sequences extracted from original video, or static, as a collection of still images.

This research proposes the creation of a component for the automatic production of scalable video summaries easily adaptable to a duration according to the needs of each user. To fulfill this objective, the implemented component guarantees the video processing following three steps: segmentation of video shots, selection of segments to include in the summary, union of the segments. During segmentation, the video is processed using color histograms in HSV space, descriptors of points of interest and image contours. With the selecting process an organized list of the shots that make up the original video is obtained, in which of the ones that have similar content are removed, by an impact-significance criteria. The results of the analysis stage are stored in a external database where they can be obtained in the desired moment for generation of the scalable summaries, considering the required duration.

The component obtained has good results for the different video encodings tested, also can generate scalable summaries that provide the coverage needed to determine the content of the original video.

Keywords: component, descriptors, scalable, summary, video.

Índice general

Índice de figuras

Índice de tablas

Introducción

La revolución en video digital presenciada en los últimos años ha sido impulsada por el rápido desarrollo en diversas áreas de la infraestructura informática como aumento de la potencia de procesamiento, de la capacidad de los dispositivos de almacenamiento, y de la velocidad de las redes. (Truong y Venkatesh, 2007)

Dicha revolución ha traído consigo la aparición de nuevas aplicaciones y, como consecuencia, la investigación de nuevas tecnologías que apuntan a mejorar el proceso de captura, almacenamiento, análisis, catalogación e indexación de video, así como a incrementar la usabilidad de los videos almacenados. (Díaz Berenguer, 2014)

En el centro Geoinformática y Señales Digitales (GEYSED) de la Universidad de las Ciencias Informáticas (UCI), se desarrollan un conjunto de aplicaciones de software de gestión, procesamiento y transmisión de contenidos audiovisuales. A los usuarios administradores de dichos software se les complejiza la aprobación de los audiovisuales a transmitir o publicar, debido a que normalmente implica un proceso de pre-visualización del material. De igual forma se complejizan los procesos de identificación y catalogación de los audiovisuales de preferencia para cada usuario cliente, donde estos últimos, al no conocer el contenido del material a consumir se ven obligados a la reproducción completa del mismo haciendo uso del ancho de banda disponible.

Para facilitar la recuperación de los materiales se realiza una descripción manual de estos, para ello los encargados de la descripción necesitan visualizar completa o parcialmente cada audiovisual hasta encontrar los fragmentos representativos para el material en su conjunto.

El gran volumen de información de video existente dificulta su gestión, por lo tanto hay una fuerte demanda de un mecanismo que permita obtener cierta perspectiva de un archivo de video sin visualizar su contenido completamente. Según (Truong y Venkatesh, 2007), esto último ha alimentado un área de investigación que evoluciona rápidamente conocida como **abstracción de video**, que estudia los métodos y técnicas de procesamiento de video para generar automáticamente resúmenes de los mismos.

Según (Díaz Berenguer, 2014) la mayoría de las aproximaciones que se observan en la literatura para generar resúmenes automáticos de videos, a partir de una secuencia original, generan un único resumen, lo que a veces puede ser insuficiente ya que en ocasiones es deseable lograr una presentación personalizada para cada usuario; afirmación que se considera acertada. Dicha presentación personalizada

puede obtenerse mediante la generación de resúmenes escalables de video. (Valdés y Martínez, 2008) (Herranz y Martínez, 2010) (López Meneses, 2013) (Díaz Berenguer, 2014)

Atendiendo a la situación planteada anteriormente, se ha identificado el siguiente **problema a resolver:** la determinación del contenido de un video se dificulta debido a que se requiere navegar aleatoriamente sobre la secuencia o visualizar completamente la misma.

Por lo que se delimita como **objeto de estudio de la investigación** los procedimientos para obtener resúmenes de videos, enmarcado en el **campo de acción** de los procedimientos para generar resúmenes escalables de videos.

Para dar solución al problema a resolver en la investigación se establece como **objetivo general**: desarrollar un componente de software que permita generar resúmenes escalables de videos.

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

- ¿Cuáles son los fundamentos teóricos que rigen el desarrollo de los procedimientos que permiten obtener resúmenes de videos?
- ¿Cómo ha evolucionado el desarrollo procedimientos para generar resúmenes escalables de videos?
- ¿Qué métodos, proceso y herramientas hay que tener en cuenta para el desarrollo de un componente de generación de resúmenes escalables de videos?

Para dar cumplimiento al objetivo general de la presente investigación se proponen las siguientes tareas de investigación:

1. Caracterización de los métodos y técnicas para la obtención de resúmenes de videos.
2. Determinación y fundamentación de los métodos y técnicas factibles para realizar resúmenes escalables de video.
3. Determinación de las herramientas y tecnologías de desarrollo a utilizar en la construcción del componente.
4. Elaboración de la documentación y elementos de configuración necesarios según el proceso de desarrollo utilizado.
5. Implementación de las funcionalidades determinadas.
6. Validación funcional del componente de software desarrollado.

Como métodos teóricos de la investigación científica se emplearon:

El método **analítico-sintético** con el objetivo de analizar y procesar la bibliografía relativa al objeto de estudio, lo que permitió obtener un conocimiento concreto durante la investigación para definir la propuesta del componente a desarrollar.

El **histórico-lógico** para realizar un análisis profundo sobre la evolución y comportamiento de los procedimientos para generar resúmenes escalables de video e identificar las tendencias generales de los mismos.

El método **inductivo-deductivo** el cual posibilitó valorar los métodos y técnicas para realizar resúmenes de video, apoyándose en conclusiones obtenidas a partir del análisis del objeto de estudio.

La **modelación** para la realización de los modelos necesarios para una correcta construcción del componente.

El presente documento está constituido por los siguientes capítulos:

Capítulo 1: Fundamentos teóricos. Métodos, proceso y herramientas. Se explican los diferentes conceptos asociados a la solución del problema que permiten comprender la solución propuesta, así como el análisis de soluciones existentes. Además se analiza el objeto de estudio y se caracterizan la metodología, tecnologías, lenguajes de programación y herramientas para el desarrollo del componente de software.

Capítulo 2: Análisis y diseño del componente. Se desarrolla el análisis y diseño del componente, siguiendo las pautas de la metodología seleccionada. Se define la arquitectura del componente y los patrones de diseño a utilizar.

Capítulo 3: Implementación y prueba del componente. En este capítulo se plantea la construcción de la solución propuesta en el capítulo anterior. Se presenta el modelo de implementación de la solución y el diagrama de despliegue. Además se realizan las descripciones de las pruebas y se exponen los resultados de las mismas.

1. Capítulo I: Fundamentos teóricos. Métodos, proceso y herramientas.

Introducción

Este capítulo tiene como objetivo fundamental abordar los principales conceptos y enfoques de la abstracción de video para lograr una mejor comprensión del contenido de la investigación y su objeto de estudio. Se realiza un estudio sobre las soluciones existentes. Finalmente se describen y caracterizan los métodos, el proceso y las herramientas para desarrollar la solución propuesta.

1.1. Términos asociados al dominio del problema

Los conceptos asociados al problema son explicados a continuación, lo que permitirá tener un dominio teórico del objeto de estudio.

1.1.1. Componente

Aplicado a las ciencias informáticas **componente[1] de software** es un elemento de un sistema software que ofrece un conjunto de servicios o funcionalidades, a través de interfaces definidas. Sus principales características son (Aritza Rojas, 2004):

- Auto contenido: un componente no debe requerir de la utilización de otros para finiquitar la función para la cual fue diseñado.
- Puede ser remplazado por otro componente: se puede remplazar por nuevas versiones u otro componente que lo remplace y mejore.
- Con acceso solamente a través de su interfaz: debe asegurar que estas no cambiaran a lo largo de su implementación.
- Sus servicios no varían: las funcionalidades ofrecidas en su interfaz no deben variar, pero su implementación sí.
- Es genérico: sus servicios deben ser útiles para varias aplicaciones.
- Independiente de la plataforma: hardware, software, sistema operativo.

[1] Componente: Según (RAE): “que forma parte de un todo”.

1.1.2. Escalabilidad

En informática, la escalabilidad es la capacidad de un sistema de adaptarse a diferentes situaciones o parámetros sin perder calidad. (ALEGSA.COM.AR, 2015)

Según (Díaz Berenguer, 2014): *"…algo es escalable cuando es capaz de adaptarse adecuadamente a diferentes condiciones."* La presente investigación hace uso de este término más ajustado al marco de la misma.

1.1.3. Video

Según (Digitales, 2008-2009): *"…suele llamarse video a la captura, grabación, almacenamiento, y reconstrucción de una serie de imágenes y sonidos, las cuales representan escenas en movimiento."*

Otra definición de video es el procesamiento, almacenamiento, transmisión y reconstrucción por medios electrónicos digitales o analógicos de una secuencia de imágenes que representan escenas en movimiento. (Thakre, 2010)

En esta investigación se utiliza la definición de video de (Thakre, 2010), ajustada a los medios electrónicos digitales para referirse a los materiales audiovisuales que se usan.

Transiciones

En la edición de video una **transición** es lo que el editor muestra entre dos de las partes que componen la secuencia para hacer un cambio de contexto. La transición por **corte** es aquella que se hace sin efectos especiales o artísticos. (Salvat Salvat y otros, 2010)

En el Anexo I se muestran ejemplos del comportamiento de algunos de dichos efectos de transición.

Estructura del contenido de un video

La estructura de un video puede descomponerse de forma jerárquica ascendente en fotograma, toma, y escena respectivamente. (Díaz Berenguer, 2014)

El **fotograma** es la unidad básica del video, una imagen sin movimiento, una matriz de píxeles[2] que recrean el contenido de esta imagen. (Díaz Berenguer, 2014)

[2] Píxeles: (acrónimo del inglés *picture element*, "elemento de imagen") es la menor unidad homogénea en color que forma parte de una imagen digital.

Una **toma** es una colección de fotogramas consecutivos o segmento de video, sin cortes ni transiciones, resultantes de una acción de grabación continua; por lo cual generalmente posee un fondo consistente característico. (Díaz Berenguer, 2014)

Los cambios de tomas pueden dividirse en **abruptos**[3] (el cambio entre dos fotogramas consecutivos es total) o **graduales**[4] (efectos de edición tales como desvanecimiento de entrada y de salida, disolvencias o barridos). (Díaz Berenguer, 2014)

A su vez una **escena** es una colección de una o más tomas que se encuentran en un espacio temporal cercano y tienen relación semántica entre ellas. (Díaz Berenguer, 2014)

Según (Díaz Berenguer, 2014), la toma constituye un límite físico por lo que se pueden identificar los cambios con mayor facilidad, a diferencia de la escena, ya que la identificación de esta última depende de una apreciación semántica del contenido del video.

Para identificar correctamente la estructura de un video hay que tener en cuenta primeramente el tipo de contenido del mismo.

Las modalidades en cuanto a tipo de contenido se dividen en **basado en guion**: el video tiene una estructura bien definida (Ej.: los seriales, las películas, noticias o documentales) y **no basado en guion**: los eventos ocurren de forma aleatoria (Ej.: los eventos deportivos, películas caseras o grabaciones de cámaras de video protección) (Díaz Berenguer, 2014).

La presente investigación se enfoca en la modalidad basada en guion debido a que en este tipo de audiovisual es menos compleja la identificación de sus tomas.

1.1.4. Resumen de video

Un resumen de video es una representación sintetizada del contenido del mismo. Dicho resumen es un conjunto determinado de imágenes o una secuencia de video de menor tiempo de duración, pero con representaciones de la información del contenido original. (Truong y Venkatesh, 2007) (Herranz Arribas, 2010)

Tipos de resúmenes de video

[3] Abrupto. (Loc. lat.; literalmente, 'con brusquedad'). 1. loc. adv. De repente, de improviso. (RAE)
[4] Gradual. (Del lat. gradus, grado). 1. adj. Que va de grado en grado. (RAE)

- Resumen estático o *storyboard*[5]: Según (Truong y Venkatesh, 2007): *"…consiste en una colección de fotogramas claves (keyframes*[6]*) o imágenes fijas extraídas del video original"*
- Resumen dinámico o s*kim*[7]: Según (Truong y Venkatesh, 2007): *"…consiste en una colección de segmentos de video extraído del video original."*

Ambas formas de abstracción de video, **resumen estático** y **resumen dinámico**, son ventajosas dependiendo de las necesidades específicas de cada usuario. El resumen estático tiene como características principales su pequeño espacio de almacenamiento y tiempo de visualización; por su parte el resumen dinámico brinda al usuario un entendimiento diferente sobre el argumento del video original.

Los algoritmos de creación de resúmenes de video requieren de gran capacidad de procesamiento y por lo general generan un único resumen a partir de una secuencia original. En vista del aprovechamiento de la capacidad de procesamiento y la obtención de resultados personalizables de dichos algoritmos es recomendable usar resúmenes escalables. (Herranz Arribas, 2010)

- Resumen escalable: Según (Díaz Berenguer, 2014), generación de resúmenes escalables de video es la aptitud de un algoritmo para generar múltiples secuencias sintetizadas de un video; ajustando las secuencias resultantes a preferencias o condiciones pre-establecidas, donde estas últimas pueden ser, por ejemplo, la resolución, la tasa de bits por segundo, la longitud o duración.

Para la investigación se asume por resumen escalable a cada una de dichas secuencias sintetizadas, teniendo en cuenta como condición de escalabilidad la longitud o duración del resumen.

1.2. Descripción actual del dominio del problema

El centro GEYSED cuenta con una variada producción de software de gestión, transmisión y publicación de audiovisuales en los cuales se dificultan los procesos de clasificación y aprobación de los materiales a publicar o transmitir. Dichos procesos se obstaculizan en el sentido de que los responsables de los mismos deben visualizar de forma parcial o total los materiales a transmitir o publicar en la web.

[5] *Storyboard*: Proveniente de la cultura anglosajona. Guion gráfico.
[6] *Keyframe*: Proveniente de la cultura anglosajona. *Key*: Clave. *Frame*: Marco o fotograma. También llamado fotograma representativo.
[7] *Skim*: Proveniente de la cultura anglosajona. Porción, extracto, también llamado *storyboard* en movimiento o sumario de secuencia.

Según (López Meneses, 2013), una variante para solucionar este problema es buscar la sinopsis textual del video, pero la misma puede no existir o carecer de veracidad. A su vez, una sinopsis es un resumen de texto, por lo tanto hay muchas características de una imagen o de un video que para describir textualmente dependen de que una persona visualice al menos los fragmentos representativos del material en cuestión.

Los usuarios consumidores de servicios de dichos software también dependerían de la sinopsis para obtener una idea del contenido de los audiovisuales, de lo contrario necesitarían visualizarlo en su totalidad y así identificar los de su preferencia, lo cual requiere tiempo y ancho de banda disponible. Por lo tanto sería ventajoso la utilización de otra forma de generar resúmenes que puedan ser obtenidos automáticamente y aporten más información al cliente.

Se propone la aplicación de técnicas de resumen de video para proporcionar una secuencia con menor tiempo de duración, pero con las representaciones significativas del contenido original, que podría ser utilizado como pre-visualizaciones, facilitando los procesos de identificación, catalogación y aprobación de materiales audiovisuales.

1.3. Descripción general del objeto de estudio

De los estudios (Truong y Venkatesh, 2007) (López Meneses, 2013) (Díaz Berenguer, 2014) previamente realizados sobre las técnicas de abstracción de video han surgido algunos procesos con el objetivo de modulizar dichas técnicas. En (Truong y Venkatesh, 2007) se propone un proceso genérico para la generación de resúmenes dinámicos, del cual se deriva la propuesta explicada a continuación.

1.3.1. Proceso de generación de resúmenes de video

Se identificó un proceso de tres pasos para la generación automática de resúmenes dinámicos de video:

1. Segmentación del video en tomas.
2. Selección de los segmentos a incluir en el resumen.
3. Unión de los segmentos. Incluye una posible integración de transiciones y efectos de audio o video para hacer más agradable el resultado final al usuario.

En el caso de tratarse de un resumen estático se selecciona uno o varios fotogramas claves de cada segmento para ser incluidos en el resultado final.

En la práctica la utilización de diferentes técnicas de abstracción de video puede obviar o adicionar algún paso o combinarlos en diferentes variaciones (Truong y Venkatesh, 2007).

1.3.2. Segmentación del video

En los trabajos (López Meneses, 2013) (Díaz Berenguer, 2014), la segmentación del video o detección de tomas es realizada mediante el uso de diferentes descriptores visuales para retornar una lista de las tomas que componen el video original y las características de las mismas. Dicha lista resultante se utilizará en las etapas restantes del proceso de generación de resúmenes.

Descriptor visual

Según (RAE), descriptor es el término o símbolo válido y formalizado que se emplea para representar inequívocamente los conceptos de un documento o de una búsqueda.

En el campo del procesamiento de video se le denomina descriptores visuales a aquellos que son utilizados para destacar un conjunto de píxeles (características) de un fotograma que son apreciables solo desde el punto de vista humano. Algunas de estas características que son extraídas de las imágenes con descriptores son color, textura, forma o movimiento (López, 2011).

En la literatura se pueden observar varias taxonomías para los descriptores visuales. Una de estas se basa en la región de incidencia sobre la imagen, para lo cual se dividen en 2 grupos: descriptores globales y descriptores locales.

Los **descriptores globales** realizan un análisis completo del área de una imagen y retornan un vector de características que describe la misma en su totalidad. (Quintana Rondón y Hernández Heredia, 2012).

En cambio los **descriptores locales** retornan un vector de características de una región predefinida de la imagen y por lo general son utilizados para obtener particularidades de las regiones de los bordes, o para identificación de rostros u objetos en regiones específicas. (Tuytelaars y Mikolajczyk, 2008)

Histogramas de color

Según (Bradski y Kaehler, 2008), un histograma es una recolección de conteos o estadísticas de algún dato, agrupadas en un número predeterminado de divisiones, llamadas *bins* en la literatura anglosajona. El **histograma de color** es una representación de la distribución del color en una imagen. Aplicado a una imagen digital, el histograma de color sería un vector que representa el número de píxeles que tienen colores en cada rango de los colores de la imagen. Los histogramas de color son un método efectivo para la

detección de cambios de tomas abruptos. La diferencia entre dos fotogramas consecutivos permite detectar un cambio de toma abrupto si la **distancia** entre los vectores de los histogramas de color de las mismas sobrepasa un umbral definido. (Nagasaka y Tanaka, 1992) (Swanberg, Shu, y Jain, 1993) (Ueda, Miyatake, y Yoshizawa, 1991)

Entre las desventajas del histograma de color para la detección de cambios de tomas se encuentra la alta sensibilidad a los cambios de iluminación. También algunos efectos de edición de los videos o transiciones como son: disolver, desvanecimiento y el barrido de la imagen, causan falsos positivos. (Valdés y Martínez, 2007)

Los histogramas de color pueden ser extraidos en los diferentes espacios de color existentes. En (Díaz Berenguer, 2014) se recomienda el uso del espacio de color HSV[8]. (Díaz Berenguer, 2014) explica las ventajas del uso de este modelo, entre ellas señalar que es un espacio de color perceptualmente uniforme donde la componente de intensidad se separa de la información de color y las otras dos componentes proveen una representación del color que constituye la que más se asemeja al sistema humano de percepción del color.

Contornos

Entiéndase como contorno a las líneas que permiten representar la forma de los objetos presentes en una imagen. La detección de los contornos puede ser usada para detectar movimientos de cámaras, cambios de iluminación y transiciones de imágenes en los cambios de tomas graduales. La comparación de dos fotogramas consecutivos permite detectar que se está produciendo un cambio de toma gradual si la distancia entre los contornos que entran y salen es mayor que un umbral definido. (Lienhart, 1997) (Zabih, Miller y Mai, 1993).

Este método implica un gran costo computacional por lo que se recomienda usar el algoritmo propuesto por (Canny, 1986). Dicho algoritmo obtiene resultados incluso cuando la imagen tiene ruido por lo que califica como aproximadamente óptimo (Chandrakar y Bhonsle, 2012).

[8] HSV: Matiz, Saturación y Valor de intensidad, del inglés, *Hue, Saturation and Intensity Value*.

Descriptores SIFT y SURF

SIFT (*Scale Invariant Feature Transform*)[9] y **SURF** (*Speeded Up Robust Features*)[10] son descriptores locales a diferencia de los antes descritos, los mismos extraen puntos o regiones características de las imágenes y conforman con estos un vector que las describe. Dichos descriptores son muy utilizados para la detección de objetos en un fotograma por ser muchos más precisos, pero tienen un costo computacional más alto que los globales. (Quintana Rondón y Hernández Heredia, 2012)

El descriptor SIFT, fue desarrollado para detectar puntos característicos estables en una imagen. Dichos puntos se muestran invariables frente a diferentes transformaciones como traslación, rotación, escala, iluminación y transformaciones afines. Originalmente fue desarrollado para el reconocimiento de objetos de manera general y realiza la correspondencia entre puntos basada en los vectores de características de cada punto que componen el descriptor de la imagen. (Boullosa García, 2011)

El descriptor SURF fue desarrollado como un detector de puntos de interés robusto. Este guarda cierta similitud con la filosofía del descriptor SIFT, aunque presenta notables diferencias con respecto al anterior. Los autores afirman que este detector y descriptor presenta principalmente tres mejoras resumidas en los siguientes conceptos, por lo que es recomendado su uso (Boullosa García, 2011):

- Velocidad de cálculo considerablemente superior sin ocasionar pérdida del rendimiento.
- Mayor robustez ante posibles transformaciones de la imagen.
- Genera menor cantidad de información a partir del análisis de una imagen.

Se selecciona SURF como descriptor local a usar por sus características de mayor robustez, mejores tiempos de cómputo y por generar menos datos.

Similaridad y distancia entre fotogramas

La **similaridad** (S) es una medida que denota que tan parecidos son dos objetos x e y, a su vez la **distancia** (D) entre dichos objetos denota que tan disimilares son los mismos. Para una representación numérica de estas medidas se tomaron valores normalizados entre 0 y 1, donde para la similaridad valores más cercanos a 1 representan mayor parecido y para la distancia los valores más cercanos a 0, respectivamente.

[9] Características no variantes por transformación de escala
[10] Características robustas aceleradas.

Dada la siguiente taxonomía de operaciones binarias, (Véase Tabla 1), propuesta por (Choi y otros, 2010):

Tabla #1 Taxonomía de operaciones binarias.

i / j	1	0	Sum
1	$a= i \cap j$	$b= \sim i \cap j$	$a+b$
0	$c= i \cap \sim j$	$d= \sim i \cap \sim j$	$c+d$
Sum	$a+c$	$b+d$	$n= a+b+c+d$

Donde i y j son dos vectores binarios de características, n el número de características o dimensión del vector, a es el número de características donde los valores de i y j coinciden, b es el número de características ausentes en i y presentes en j, c es el número de características ausentes en j y presentes en i y d es la ausencia de la características en ambos vectores.

Para el cálculo de la similaridad entre las imágenes binarias (fotograma en blanco y negro), resultantes de la detección de contornos, se propone la fórmula 3W–JACCARD descrita por (Choi y otros, 2010):

$$(1) \qquad S_{3W-JACCARD} = \frac{3a}{3a+b+c}$$

Donde la característica a tener en cuenta corresponde a la existencia o no de un pixel negro (Ej.: a pasaría a ser la existencia de un pixel negro coincidente en ambas imágenes).

Para el cálculo de la similaridad entre los vectores pertenecientes a la extracción de las características SURF de dos fotogramas se propone también la fórmula 1, donde la característica a tener en cuenta corresponde a la existencia o no de una característica SURF (Ej.: a pasaría a ser la existencia de una característica SURF coincidente y en ambas imágenes).

Para el cálculo de la similaridad entre los vectores pertenecientes a la extracción de los histogramas de color de dos fotogramas se propone la fórmula de **correlación.** (OPENCV.ORG, 2015)

Dado dos vectores *n-dimensionales* H_1 y H_2 la correlación se define como:

$$(2) \qquad S_{CORRELACION}(H_1, H_2) = \frac{\sum_i (H_1(i)-\overline{H_1})(H_2(i)-\overline{H_2})}{\sqrt{\sum_i (H_1(i)-\overline{H_1})^2 \sum_i (H_2(i)-\overline{H_2})^2}}$$

Donde:

$$(3) \qquad \overline{H_k} = \frac{1}{N}\sum_j H_k(j)$$

Nota: N es el tamaño del vector.

Finalmente, para el cálculo de la distancia entre los vectores pertenecientes a la extracción de los histogramas de color de dos fotogramas se propone la fórmula de **distancia de Minkowski.** (Kruskal, 1964)

Dado dos vectores n-dimensionales i y j, la distancia de Minkowski se define como:

$$(4) \qquad D_{MINKOWSKI}(i,j) = \sqrt[\lambda]{\sum_{k=0}^{n-1} |y_{i,k} - y_{j,k}|^{\lambda}}$$

Donde k es el índice del valor, y el vector correspondiente, n el tamaño del vector y λ el orden de la métrica de Minkowski. Aunque por definición $\lambda > 0$, raramente es usado para valores diferentes de 1 y 2, donde para el orden 1 se obtiene el resultado de la distancia Manhattan y para el orden 2 el resultado de la distancia Euclidiana.

1.3.3. Método de selección

El método de selección asegura obtener las tomas más representativas del video. Luego de este proceso se retorna una lista de las tomas que compondrán el resumen final.

Calificación

La selección de tomas a incluir en el resumen resultante debe hacerse con un correcto balance de significación-impacto (Truong y Venkatesh, 2007). Otorgando una **calificación** en base a estos dos indicadores se puede conformar una jerarquía o lista organizada por la cual seleccionar las tomas de acuerdo a su importancia. Para lograr dicha jerarquía se combinaron varias características de las tomas tales como (Herranz y Martínez, 2010) (Valdés y Martínez, 2008):

- Disimilitud: característica de significación que indica qué tanto difiere una toma del resto.
- Variabilidad: característica de significación que indica qué tanto varia una toma en su interior.
- Duración: característica de la toma la cual indica su longitud o tiempo de duración que por lo general es deseable que sea lo menor posible para disminuir su impacto sobre el resumen final.

Selección

En (Truong y Venkatesh, 2007), se mencionan variadas formas de selección de tomas a incluir con diferentes complejidades. En algunas aproximaciones existentes para generar resúmenes automáticos de video prevalecen los enfoques que basan su selección en la utilización de agrupamiento jerárquico (Valdés y Martínez, 2008) (Herranz y Martínez, 2010) (López Meneses, 2013) (Díaz Berenguer, 2014), pero se considera que los métodos de agrupamiento requieren un gran costo computacional. Por ejemplo, en (Babaguchi, 2000) simplemente se seleccionan los eventos en orden decreciente de impacto para el resumen final. Con el objetivo de reducir el costo computacional se propone una estrategia basada en fotogramas clave; comparar los fotogramas clave de dos tomas, como representación de las tomas en cuestión; en caso que la similitud entre ambos supere un umbral definido solo se selecciona la toma de mayor calificación, en caso contrario, ambas tomas son seleccionadas.

1.3.3. Unión

Este paso asegura la escalabilidad del resultado; esta investigación fue enfocada a la duración final del resumen. En (Truong y Venkatesh, 2007) se proponen dos formas de determinar la duración del video; puede ser especificado un valor predeterminado (a priori), o dejar desconocido hasta el final del proceso (a posteriori). Según (Díaz Berenguer, 2014), un resumen escalable debería lograr un número de escalas que le permita adaptarse a situaciones en las que, por ejemplo, la duración del mismo puede limitarse a un valor específico o porcentual. Se propone seleccionar la duración del resumen final como un valor a priori, definido como un porciento de la duración total del video original.

1.3.4. Método de selección del fotograma clave.

En la investigación de (Truong y Venkatesh, 2007) se explican diferentes mecanismos para la extracción de fotogramas clave, entre dichos mecanismos el más simple y computacionalmente eficiente es el muestreo uniforme.

Luego de segmentar el video en tomas desaparece la principal desventaja de este mecanismo, la cual se encuentra en que puede no generar fotograma clave para una toma corta, pero semánticamente representativa, a la vez que puede extraer varios para una toma relativamente estática de mayor tamaño.

En algunas aproximaciones que utilizan este mecanismo se establece como fotograma clave el primero, el último o un fotograma de la toma seleccionado de forma aleatoria lo cual no garantiza que el fotograma seleccionado sea realmente representativo de la toma en su conjunto. (Díaz Berenguer, 2014)

Teniendo en cuenta la definición de toma, ofrecida en el epígrafe 1.1.3 del presente capítulo, se propone:

(a) *Sea la toma T una colección de fotogramas f, donde n es la cantidad de fotogramas de la toma en cuestión,* $T = (f_1, f_2, f_3, \ldots, f_n)$, $\boldsymbol{C} = \frac{\boldsymbol{n}}{\boldsymbol{2}}$, $\boldsymbol{A} = \frac{\boldsymbol{C}}{\boldsymbol{2}}$ *y* $\boldsymbol{B} = \frac{\boldsymbol{n+C}}{\boldsymbol{2}}$; $\{\boldsymbol{C}, \boldsymbol{A}, \boldsymbol{B}\} \in \mathbb{N}$

Comparar los fotogramas f_A y f_B con f_C, respectivamente, y seleccionar el que obtenga mayor similaridad de los contornos, ya que se considera que el fondo consistente característico de la toma puede estar mayormente representado de la mitad en adelante o de la mitad hacia atrás, o simplemente ser constante para toda la toma.

1.4. Análisis de las soluciones existentes.

En los últimos años se documentan aproximaciones para la generación de resúmenes de video. Varios autores han publicado propuestas de técnicas y algoritmos que se recogen en estudios anteriores (Truong y Venkatesh, 2007). A continuación se caracterizan algunas soluciones para crear resúmenes escalables de videos.

1.4.1. Generación de resúmenes de video en línea basado en árboles binarios

Esta solución de los autores (Valdés y Martínez, 2008) propone la utilización de un algoritmo basado en la generación dinámica de un árbol binario de tomas que emula la posibilidad de inclusión o exclusión de cada toma del video. Evalúa los caminos posibles con la información que provee el árbol para decidir entre seleccionar o descartar cada segmento de video y se selecciona iterativamente el mejor camino en este árbol binario desechando todos los otros caminos.

La relevancia de la toma detectada se calcula de acuerdo a los siguientes criterios:

- Duración: a menor duración mayor será su prioridad en el resumen.
- Continuidad: se toma en consideración que un resumen sería más agradable si se lograra que las tomas seleccionadas tuvieran la mayor continuidad posible.
- Disimilitud: a mayor distancia con respecto a las otras tomas mayor cobertura de información.

- Redundancia: indicador de similitud entre dos fotogramas en una misma secuencia.

La escalabilidad de esta solución está presente en el hecho de que cada camino final representa un resumen diferente del video dando así la posibilidad de obtener un resultado diferente al establecer criterios diferentes de inclusión y exclusión de tomas, pero se considera que los criterios usados en la selección de tomas no aseguran una buena eliminación de la redundancia de tomas por similitud, ya que la continuidad de tomas implica normalmente que pertenezcan a una misma escena.

1.4.2. Framework para la generación de resúmenes escalables de videos

Esta solución de los autores (Herranz y Martínez, 2010) propone la generación de resúmenes escalables fácilmente adaptables a una duración predeterminada de acuerdo a los requisitos de cada caso. El algoritmo de análisis de la secuencia de video utiliza un procedimiento de *ranking*[11] iterativo en el cual cada resumen es el resultado de la extensión del anterior.

Este procedimiento balancea la cobertura de información y el placer visual para el usuario. Para lograr el balance primero buscan seleccionar la mayor cantidad de información representativa posible al tiempo que eliminan la redundancia, de esta forma el resumen es más cómodo y agradable de ver. Esto se logra seleccionando la toma que tenga mayor disimilitud con las seleccionadas anteriormente teniendo en cuenta también la duración de la misma para dar una puntuación o nivel de relevancia a la misma. Así a medida que se incluyen las tomas van quedando las de menor relevancia.

La segmentación del video utiliza una característica de los formatos MPEG-1 y MPEG-2 para una eficiente detección de fotogramas clave, pero restringe el uso del algoritmo para cualquier otro formato. Sin embargo, el sistema de puntuación es fácilmente implementable y con el mismo los autores obtuvieron buenos resultados.

1.4.3. Componente para elaborar resúmenes de video escalables automáticamente

El componente presentado por (López Meneses, 2013) construye un resumen escalable utilizando un algoritmo de ranking iterativo que hace uso de distintos criterios para dar una puntuación a las tomas y agregarlas a una lista, en la que, según el orden de aparición en la misma, será su relevancia; logrando de esta forma una representación visual escalable del material audiovisual.

[11] Posición en una escala de logro o estado, una clasificación.

Se toman en cuenta para la segmentación de tomas la utilización de los histogramas de color en el espacio de color RGB[12], la detección de contornos y el descriptor SURF.

El uso del descriptor SURF le otorga buenos resultados, pero esta solución está restringida al uso de los formatos MPEG4-Xvid y MPEG-1, aparte de que carece de una personalización paramétrica, o sea, el algoritmo usado no puede ser configurado para otros parámetros o umbrales.

1.4.4. Método para la generación automática de resúmenes escalables de videos

En (Díaz Berenguer, 2014) se propone dividir el procesamiento de los videos en dos etapas: análisis y generación.

Los datos resultantes de la etapa de análisis son almacenados en disco lo cual garantiza la escalabilidad del método ya que una vez que se realice dicha etapa a la secuencia de video, solamente sería necesario realizar nuevamente la etapa de generación en caso de requerir otro resumen del video con diferente duración; pero la recuperación de los resultados de la etapa de análisis implica el acceso a dispositivos de almacenamiento de forma reiterada lo que puede influir en el rendimiento.

También son extraídos los descriptores de contornos y los histogramas de color, pero en estos últimos, utiliza el modelo de color HSV y propone una división de los fotogramas en regiones de diferentes importancias según su actividad visual, (Véase Tabla 2), la misma se considera acertada. No obstante, se debe lograr una personalización paramétrica para garantizar la adaptación a distintos contenidos de video.

Tabla #2 Regiones del fotograma

Esquina	Triángulo	Esquina
Lateral	Centro	Lateral
Triángulo	Base	Triángulo

En esta distribución se le otorga mayor importancia a la región etiquetada Centro, luego la Base, Lateral, Triángulo y Esquina, respectivamente en dicho orden.

[12] RGB: Rojo, Verde y Azul, del inglés, *Red, Green and Blue.*

1.5. Caracterización de los métodos, proceso y herramientas.

Para lograr desarrollar el componente de resúmenes de videos propuesto es necesario conocer los métodos, el proceso y las herramientas que se van a emplear para alcanzar el objetivo trazado. A continuación se presentan las tecnologías de desarrollo a utilizar en esta investigación.

1.5.1. Metodología de desarrollo RUP[13]

El Proceso de Desarrollo Unificado (RUP) es un proceso de ingeniería de software que proporciona un enfoque disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo de software. Su objetivo es garantizar la producción de software de alta calidad que satisfaga las necesidades de sus usuarios finales. RUP se fundamenta en seis mejores prácticas: desarrollo iterativo, administración de requisitos, uso de arquitectura basada en componentes, modelamiento visual, verificación continua de la calidad y administración del cambio. (Torres, 2009)

Características principales de RUP

- Dirigido por casos de uso

Los casos de uso representan una forma particular de ver el sistema, especifican qué debe hacer el mismo para cada tipo de usuarios. A partir de los casos de uso se realiza el diseño, la implementación y las pruebas, con lo cual se puede decir que los casos de uso guían el proceso de desarrollo, además de guiar la arquitectura y a su vez esta arquitectura incide en la selección de los casos de uso. (Leterlier, 2013)

- Centrado en la arquitectura

La arquitectura define los componentes más relevantes y conforma una visión más clara del sistema que se desea desarrollar. La arquitectura involucra los aspectos estáticos y dinámicos más significativos del sistema, está relacionada con la toma de decisiones que indican cómo tiene que ser construido el sistema y ayuda a determinar en qué orden. (Leterlier, 2013)

- Iterativo e incremental

El proceso iterativo e incremental consta de una secuencia de iteraciones. Cada iteración aborda una parte de la funcionalidad total, pasando por todos los flujos de trabajo relevantes y refinando la arquitectura. Cada iteración se analiza cuando termina. Se puede determinar si han aparecido nuevos requisitos o han

[13] RUP: Proceso de Desarrollo Unificado, del inglés, *Rational Unified Process.*

cambiado los existentes, afectando a las iteraciones siguientes. Toda la retroalimentación de la iteración pasada permite reajustar los objetivos para las siguientes iteraciones. Se continúa con esta dinámica hasta que se haya finalizado por completo con la versión actual del producto. (Leterlier, 2013)

El uso de esta metodología se debe a su adaptación a las necesidades del proyecto de desarrollo para obtener el componente, ya que:

- Permite su utilización en proyectos donde las personas no cuentan con experiencia en la producción de software.
- No depende de la presencia constante del cliente.
- Permite el desarrollo de proyectos a largo plazo.
- Cuenta con una documentación detallada.

1.5.2. Lenguaje de modelado

El lenguaje de modelado de objetos es un conjunto estandarizado de símbolos utilizados para modelar un diseño de software orientado a objetos. Con la utilización del lenguaje de modelado se puede llegar a obtener diseños sólidos que sirvan como guía a seguir por los analistas, desarrolladores y clientes. Logrando unificar la visión y proyección del equipo de desarrollo hacia un producto de calidad.

Lenguaje Unificado de Modelado (UML) 2.0

UML es un lenguaje gráfico para especificar, visualizar, construir y documentar los artefactos que modelan un sistema de software. El mismo cuenta con las notaciones estándar y semánticas esenciales. Fue diseñado para ser un lenguaje de modelado de propósito general, por lo que puede utilizarse para especificar la mayoría de los sistemas basados en objetos o en componentes, y para modelar aplicaciones de muy diversos dominios de aplicación y plataformas de objetos distribuidos. (Fuentes y otros, 2004)

1.5.3. Herramienta de modelado de software

Las herramientas CASE[14], son utilizadas para realizar la representación de un producto de software, mediante diagramas que se van desarrollando durante los diferentes ciclos de vida del software. Dichas herramientas facilitan las tareas de coordinación de los eventos en el ciclo de desarrollo de software.

[14] CASE: Ingeniería de Software Asistida por Computación, del inglés, *Computer Aided Software Engineering*.

Visual Paradigm 8.0

Visual Paradigm permite modelar todos los artefactos que se obtienen a partir del análisis del negocio y el sistema. Es una herramienta multiplataforma que soporta completamente el ciclo de desarrollo de un software: análisis y diseño, construcción, pruebas y despliegue. Posibilita el modelado de base de datos, requisitos, proceso de negocio, permite realizar todo tipo de diagramas de clases, ingeniería inversa, generar código desde diagramas y generar documentación. (Visual Paradigm, 2015)

Características que presenta *Visual Paradigm* (Fuentes y otros, 2004):

- Se pueden realizar distintos tipos de diagramas, por ejemplo: diagramas de Clases, de Casos de Uso, de Secuencia y de Componentes.
- Exportación en imágenes de los diagramas que son elaborados, con formato jpg, png y svg.
- Genera la documentación del sistema en formato PDF, HTML y Microsoft Word.
- Genera el código de aplicación según el modelado de la aplicación.
- Genera la documentación de lo que se está modelando.
- Modela todos los diagramas de UML.

Se utiliza Visual Paradigm como herramienta CASE ya que ofrece salida de documentación flexible y de alta calidad, soporta la metodología RUP en su totalidad, está disponible en múltiples plataformas, lo cual facilita que la realización de diagramas de modelado no dependan del sistema operativo, y posee, además de su licencia comercial, una licencia gratuita.

1.5.4. Sistema de Gestión de Base de Datos (SGBD)

Una base de datos es una entidad en la cual se pueden almacenar datos de manera estructurada, con la menor redundancia posible. (Communitic International, 2013)

Un SGBD es un programa capaz de gestionar adecuadamente las bases de datos, permitiendo a su vez:

- Definir una base de datos: especificar tipos, estructuras y restricciones de datos.
- Construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD.
- Manipular la base de datos: realizar consultas, actualizarla, generar informes.

Entre los gestores de base de datos más comunes se encuentra MySQL[15], PostgreSQL[16] y SQLite.

SQLite 3

SQLite es un proyecto de dominio público, su creador es D. Richard Hipp, el cual implementa una pequeña biblioteca de aproximadamente 500kb, programado en el lenguaje C.

Algunas características de SQLite son (Maldonado, 2008):

- Soporta múltiples tablas, índices, triggers (o disparador) y vistas.
- No necesita un proceso separado funcionando como servidor, pues lee y escribe directamente sobre archivos que se encuentran en el disco duro.
- El formato de la base de datos es multiplataforma e indistintamente se puede utilizar el mismo archivo en sistemas de 32 y 64 bits.
- SQLite emplea registros de tamaño variable, de forma tal que se utiliza el espacio en disco que es realmente necesario en cada momento.

El uso de SQLite es debido a que no necesita un proceso separado funcionando como servidor, lo que hace a dicho SGDB portable, característica necesaria para su utilización en un componente de software.

1.5.5. Biblioteca

Según (ALEGSA.COM.AR, 2015) una biblioteca es una colección o conjunto de subprogramas usados para desarrollar software. En general, las bibliotecas no son ejecutables, pero sí pueden ser usadas por ejecutables que las necesitan para poder funcionar correctamente. La mayoría de los sistemas operativos proveen bibliotecas que implementan la mayoría de los servicios del sistema.

OpenCV 2.4.11

Open Source Computer Vision Library es una biblioteca de código abierto de tratamiento de imágenes, destinada principalmente a aplicaciones de visión por computador en tiempo real. (Igual y Medrano, 2008)

Está escrita en C y C++ y se ejecuta bajo Windows, Linux y Mac OS X. Fue diseñada para ser computacionalmente eficiente y principalmente con un enfoque para aplicaciones en tiempo real según

[15] MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario

[16] PostgreSQL es un gestor de bases de datos muy conocido y usado en entornos de software libre que posee un conjunto de funcionalidades avanzadas.

(Arteaga Domínguez y otros, 2011). Tiene cinco módulos fundamentales, donde cada uno tiene asignada una función específica:

- **core**: define las estructuras de datos básicas.
- **imgproc**: módulo de procesamiento de imágenes que incluye el filtrado lineal y no lineal, transformaciones geométricas de la imagen, conversiones de color, histogramas, etc.
- **video**: módulo de análisis de video que incluye estimación de movimiento, extracción de trasfondo y algoritmos de rastreo de objetos.
- **features2d**: módulo que detecta características, contiene descriptores y macheadores de características.
- **nonfree**: módulo que contiene las funcionalidades de los descriptores SIFT y SURF.

Se usa OpenCV ya que está bajo licencia BSD[17], lo que permite total libertad para usos académicos o comerciales, puede utilizarse en Windows o Linux y cuenta con las funcionalidades que se necesitan para implementar la solución propuesta.

1.5.6. Lenguaje de programación

Se define como lenguaje de programación al elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que se pone a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes (ALEGSA.COM.AR, 2015).

C++

Es un lenguaje de programación basado en C que soporta directamente conceptos de la Orientación a Objetos. Desde el punto de vista de la Programación Orientada a Objetos (POO) ofrece una buena cantidad de recursos, como son las formas de encapsulamiento, la sobrecarga de operadores y funciones, la herencia y el polimorfismo. Los mayores beneficios se aprecian a largo plazo: alta calidad del software, mayor reutilización de código y mayor facilidad de adaptación, esta última apreciada sobre todo en grandes

[17] La licencia BSD es la licencia de software libre. Esta licencia tiene menos restricciones en comparación con otras como la GPL, estando muy cercana al dominio público.

proyectos. Puede emplearse mediante programación basada en eventos para crear programas que usen interfaz gráfica de usuario. (C++, 2015)

Las principales ventajas que presenta son (C++, 2015):

- Difusión: al ser uno de los lenguajes más empleados en la actualidad, posee un gran número de usuarios y existe una gran cantidad de libros, cursos y páginas web dedicados a su estudio.
- Versatilidad: es un lenguaje de propósito general, por lo que se puede emplear para resolver cualquier tipo de problema.
- Portabilidad: está estandarizado y un mismo código fuente se puede compilar en diversas plataformas.
- Eficiencia: es uno de los lenguajes más rápidos en cuanto a tiempo de ejecución.
- Herramientas: existe una gran cantidad de compiladores, depuradores y bibliotecas para el trabajo con él.

C++ es el lenguaje de programación principal de OpenCV 2.x. Debido a sus características se usa como lenguaje de programación utilizado para la implementación del componente.

1.5.7. Marco de trabajo

En el desarrollo de software, un marco de trabajo o *framework* (en inglés) es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado.

Los marcos de trabajo suelen incluir:

- Soporte de programas.
- Bibliotecas.
- Software para desarrollar y unir diferentes componentes de un proyecto de desarrollo de programas.

Estos facilitan el desarrollo de software permitiendo concentrar más esfuerzo y tiempo en identificar los requerimientos de software (ALEGSA.COM.AR, 2015).

Qt 5.2

Qt es una aplicación multiplataforma y un *framework* de interfaz de usuario para los desarrolladores que utilizan C ++, CSS y JavaScript como lenguaje. Presenta un gran número de características entre las que se destacan las siguientes (Qt Project, 2015):

- Es portable desde sistemas de escritorio hasta sistemas operativos embebidos.
- Herramientas y entornos de desarrollo multiplataforma.
- Alto rendimiento en todas las plataformas.

Cuenta con abundante documentación, arquitectura que soporta el uso de plugins y hasta el momento ha sido liberado bajo dos licencias, la LGPL[18] y la comercial (Qt Project, 2015). Presenta varios componentes que facilitan el trabajo a los desarrolladores, los cuales se mencionan a continuación (Qt Project, 2015):

- Las bibliotecas Qt: clases escritas en C++ que facilitan el desarrollo.
- QtDesigner: para diseñar formularios visualmente.
- QString: funcionalidades de cadenas de caracteres.
- QSqlDatabase: funcionalidades de conexión con diferentes tipos de bases de datos.
- Qmake: simplifica el proceso de construcción de proyectos en las diferentes plataformas soportadas.

1.5.8. Entorno integrado de desarrollo

Un *IDE*[19] es un programa compuesto por una serie de herramientas que utilizan los programadores para escribir códigos. Esta herramienta puede estar pensada para su utilización con un único lenguaje de programación o bien puede dar cabida a varios de estos. Las herramientas que normalmente componen un entorno integrado de desarrollo son las siguientes: un editor de texto, un compilador, un intérprete, unas herramientas para la automatización, un depurador, un sistema de ayuda para la construcción de interfaces gráficas de usuario y, opcionalmente, un sistema de control de versiones. (Techopedia, 2015)

QtCreator 3.2

Es un IDE multiplataforma, adaptado a las necesidades de los desarrolladores, permite crear aplicaciones de escritorio y plataformas de dispositivos móviles a través del *framework* Qt. Sus características más relevantes se detallan a continuación (Nokia, 2011):

[18] LGPL: Licencia Publica General Menor, del inglés, Lesser *General Public License*.
[19] IDE: Entorno Integrado de Desarrollo, del inglés, *Integrated Development Environment*.

- Editor de código sofisticado: ofrece soporte para la edición de C++, ayuda sensible al contexto, completamiento de código y buena navegación.
- Control de versiones: se integra con la mayoría de los populares sistemas de control de versiones.
- Diseño de interfaz de usuario integrado: proporciona dos editores visuales integrados: QtDesigner para la creación de interfaces de usuario y Qt Quick Designer para el desarrollo de interfaces de usuario animadas con el lenguaje QML.
- Proyecto y gestión de construcción: si se importa un proyecto existente o se crea uno desde cero, genera todos los archivos necesarios.
- Escritorio y objetos móviles: ofrece soporte para crear y ejecutar aplicaciones para equipos de escritorio y dispositivos móviles.

1.6. Conclusiones del capítulo

Con el estudio de los conceptos asociados al objeto de estudio, el campo de acción y las soluciones similares existentes, se ha obtenido una visión más clara de los elementos y características que se deben tener en cuenta durante el diseño y construcción de un componente de resúmenes escalables de videos. Por lo que se llega a las siguientes conclusiones:

1. La utilización de diferentes técnicas de detección de tomas en combinación aseguran la correcta segmentación del video.
2. La construcción del resumen final debe de generarse en ambas formas de resúmenes de video ya que cada una posee ventajas propias.
3. El estudio de las soluciones existentes aportó conocimientos y técnicas útiles para el desarrollo del componente; tales como el uso de un *ranking* de tomas, división del proceso en etapas, uso del espacio de color HSV, regiones del fotograma, descriptores locales y criterios de calificación o puntuación de tomas.
4. Se identificó la necesidad del almacenamiento en base de datos del resultado de cada etapa con el objetivo de permitir continuar el proceso desde cualquier punto deseado.

2. Capítulo II: Análisis y diseño del componente

Introducción

Una vez conocidos los conceptos asociados al tema, las herramientas y las tecnologías a utilizar en el proceso de creación del componente, es necesario efectuar el análisis y diseño del mismo. A partir del análisis se extrajeron los requisitos funcionales y no funcionales que responden a las necesidades existentes. Seguidamente se muestra el estilo y patrón de arquitectura empleados, los patrones de diseño, y por último, se desarrolla el modelo de implementación compuesto por los diagramas de componentes y el diagrama de despliegue.

2.1. Modelo de dominio

El modelo de dominio es el punto de partida para lograr el diseño adecuado del sistema que se desea desarrollar ya que expresa el razonamiento obtenido del problema. Este permite la visualización de clases conceptuales desde el punto de vista del mundo real y no de las que componen el sistema. Utilizando UML, un modelo del dominio se puede representar como un conjunto de diagramas de clases en los que no se define ninguna operación. Dichos diagramas representan (Larman, 2008):

- Objetos del dominio o clases conceptuales.
- Asociaciones entre las clases conceptuales.
- Atributos de las clases conceptuales.

2.1.1. Descripción del modelo de dominio

El flujo que se obtiene a partir del dominio existente, (Véase Fig. 1), es el siguiente: El cliente solicita un determinado video. Este video se encuentra dentro de un servidor de contenidos audiovisuales. A su vez el contenido de este video fue descrito y aprobado por la persona encargada.

- **Cliente:** Sistema, componente o usuario que necesita obtener un video.
- **Servidor de almacenamiento:** Estación con gran capacidad de almacenamiento que provee contenidos audiovisuales.

- **Video:** Conjunto de imágenes y sonidos agrupados en tomas, que tienen un nombre que los identifica, una determinada duración y una frecuencia de fotogramas por segundo (Fps).
- **Catalogador:** Usuario que visualiza el contenido del video para describir, catalogar y aprobar su publicación en el “Servidor de almacenamiento”.

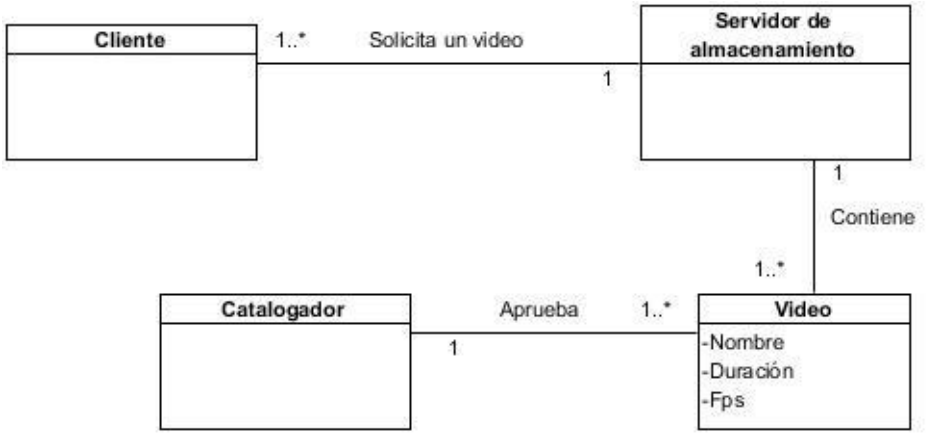

Fig. 1 Modelo del dominio

2.2. Requisitos

En el presente epígrafe se estará abordando sobre los requisitos del componente. Los requisitos no son más que las condiciones necesarias que debe cumplir el sistema para alcanzar los objetivos trazados.

2.2.1. Requisitos funcionales

Los requisitos funcionales (RF) son declaraciones de servicios que el sistema debe proporcionar, define la manera en que éste debe reaccionar a determinadas entradas y cómo se debe comportar en situaciones particulares. (Pressman, 2012)

Con el objetivo de desarrollar la solución se identificaron los siguientes requisitos funcionales:

RF 1 Segmentar el video en tomas: El componente debe permitir efectuar la segmentación del video o separación del mismo en las tomas que lo componen.

RF 2 Almacenar tomas en base de datos: El componente debe almacenar en la base de datos los resultados retornados por el RF1.

RF 3 Extraer las características de las tomas: El componente debe permitir obtener la duración, variabilidad, fotograma clave y distancia de las tomas.

RF 4 Calificar las tomas: El componente debe permitir obtener una puntuación para cada toma en dependencia de sus características.

RF 5 Seleccionar las tomas: El componente debe seleccionar las tomas más relevantes para conformar los resúmenes escalables.

RF 6 Generar el resumen escalable: El componente debe permitir generar el resumen escalable de acuerdo a la longitud o duración solicitada, utilizando la puntuación dada a las tomas en el RF 3. .

RF 7 Ordenar tomas seleccionadas: El componente debe organizar la aparición de las tomas en el resumen de video siguiendo el flujo temporal que tengan en el video original.

RF 8 Construir el resumen de video: El componente debe permitir efectuar la concatenación de las tomas o fotogramas claves según el tipo de resumen de video final a obtener.

2.2.2. Requisitos no funcionales

Los requisitos no funcionales (RNF) son las restricciones de los servicios o funciones ofrecidas por el sistema, restringen el espacio de posibles soluciones. (Pressman, 2012)

Para el correcto funcionamiento de la aplicación deben tenerse en cuenta los siguientes requisitos no funcionales:

Soporte

RNF 1 Requisitos de software:

- Sistema operativo: Windows versiones XP o superior. Ubuntu versiones 12.04 o superior.
- Biblioteca: OpenCV versión 2.4.11.

RNF 2 Requisitos de hardware:

- Procesador Intel Core i3 o superior.
- RAM: 4Gb o superior.
- Disco Duro: 80Gb o superior.

Restricciones de diseño

RNF 3 Estándares:
Se deben emplear estándares para la codificación del componente, estos se pueden encontrar en el Anexo II del documento.

Usabilidad

RNF 4 Configuración:
El cliente debe poder personalizar la ejecución del componente a través de parámetros.

Confiabilidad

RNF 5 Interrupción de flujo:
Si se interrumpe la energía o la red en la estación de procesamiento donde está corriendo el componente no se debe perder el resultado de las etapas vencidas. Una vez reanudada la energía en la estación debe ejecutarse manualmente el componente para que inicie sus funcionalidades.

RNF 6 Excepciones:
El sistema debe garantizar un tratamiento adecuado cuando ocurra una excepción, enviando mensajes al cliente informando las causas del error.

2.3. Casos de uso

Los casos de uso (CU) se utilizan con el objetivo de representar como debe de trabajar un sistema, por lo que son descripciones de las funcionalidades del mismo. Independiente de la implementación, describen el comportamiento de un sistema desde la perspectiva del cliente bajo la forma de acciones y relaciones.

2.3.1. Definición del actor

Según (Pressman, 2012), un actor del sistema no es más que un conjunto de roles que los usuarios desempeñan cuando interactúan con los casos de uso. El actor **identificado** para el sistema se describe en la Tabla 3.

Tabla #3 Descripción del actor del sistema

Actor	Descripción

Cliente	Representa al sistema o el usuario que accede a las funcionalidades que provee el componente.

2.3.2. Diagrama de casos de uso

El diagrama de CU representa la forma en cómo un actor opera con el sistema, además de la forma y orden en como los elementos interactúan.

A continuación se presenta el “Diagrama de casos de uso”, (Véase Fig. 2), del componente propuesto:

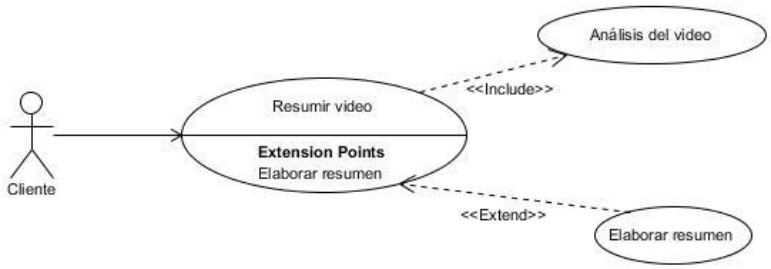

Fig. 2 Diagrama de casos de uso

Al recibir una solicitud de “Resumir video”, el componente invoca al CU “Análisis del video” y al CU “Elaborar resumen” solo en caso de ser precisado por el actor al enviar la solicitud.

2.3.3. Descripción de los casos de uso

Tabla #4 Descripción del CU “Resumir video”

Caso de uso	Resumir video
Actores	Cliente
Resumen	El Caso de uso inicia cuando el actor solicita resumir un video. Esta petición es recibida por el componente a través de la interfaz de comunicación y es efectuada. Se finaliza cuando el componente actualiza su estado a “Tarea finalizada”.
Complejidad	Alta
Prioridad	Crítica
Pre-condiciones	Conectividad con la base de datos.
Post-condiciones	Capacidad disponible en el disco para almacenar el resumen.
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**

1. El cliente manda una solicitud de “Resumir video”, en la que especifica la ruta del video. En caso de desear un resumen se especifica también el tipo y la duración del mismo.		
	2. Comprueba que no exista análisis previo del video registrado en la base de datos.	
	3. Invoca al CU “Análisis del video”.	
	4. Comprueba que se desea elaborar un resumen.	
	5. Invoca al CU “Elaborar resumen”.	
	6. Informa al usuario de que el proceso ha concluido. Termina así el CU.	
Flujo alterno “Existe análisis previo del resumen”		
	3a. Se pasa al paso 4 del flujo normal de eventos.	
Flujo alterno “No se desea elaborar un resumen”		
	4b. Se pasa al paso 6 del flujo normal de eventos.	
Relaciones	**CU Incluidos**	Análisis del video
	CU Extendidos	Elaborar resumen

Tabla #5 Descripción del CU “Análisis del video”

Caso de uso	Análisis del video
Actores	Cliente
Resumen	El Caso de uso inicia cuando es invocado. Se finaliza cuando el componente actualiza su estado a “Análisis realizado”.
Complejidad	Alta
Prioridad	Crítica
Pre-condiciones	Conectividad con la base de datos. Se conoce la ruta del video a analizar.
Post-condiciones	
Referencias	RF 1, RF 2, RF 3, RF 4, RF 5
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
	1. Comprueba que no haya una segmentación previa del video registrada en la base de datos.

	2. Extrae las tomas existentes en el video original, efectuando la detección de tomas basada en histogramas, en contornos y en SURF.
	3. Almacena en base de datos las tomas detectadas, incluyendo su puntuación y número de fotograma clave.
	4. Selecciona las tomas más representativas del video en su totalidad. Termina así el CU.
Flujo alterno "Existe segmentación previa"	
	3a. Se pasa al paso 5 del flujo normal de eventos.

Tabla #6 Descripción del CU "Elaborar resumen"

Caso de uso	Elaborar resumen
Actores	Cliente
Resumen	El CU inicia si el actor solicitó elaborar un resumen de video y especificó el tipo y la longitud del mismo. Se finaliza cuando el componente devuelve el resumen de video.
Complejidad	Media
Prioridad	Crítica
Pre-condiciones	Conectividad con la base de datos. Análisis previo registrado en la base de datos. Se conoce la ruta del video original y el tipo de resumen que se desea.
Post-condiciones	Capacidad disponible en el disco para almacenar el resumen.
Referencias	RF 6, RF 7, RF 8
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
	1. Genera el resumen escalable de acuerdo a la duración especificada.
	2. Construye el resumen y se almacena en disco. Termina así el CU.
Flujo alterno "No existe análisis previo del video"	
	2a. Termina así el CU.

2.4. Descripción de la propuesta de algoritmo

Para una mejor comprensión se explica a continuación como se pretende implementar la propuesta de algoritmo a partir de los requisitos funcionales identificados. En la Fig. 3 se pueden observar las diferentes etapas englobadas en cada funcionalidad concreta del componente.

Durante el análisis del video se evidencian los pasos de segmentación y selección del proceso de generación de resúmenes de video; y durante la elaboración, la generación del resumen escalable de acuerdo a la duración deseada y la unión de los segmentos correspondientes.

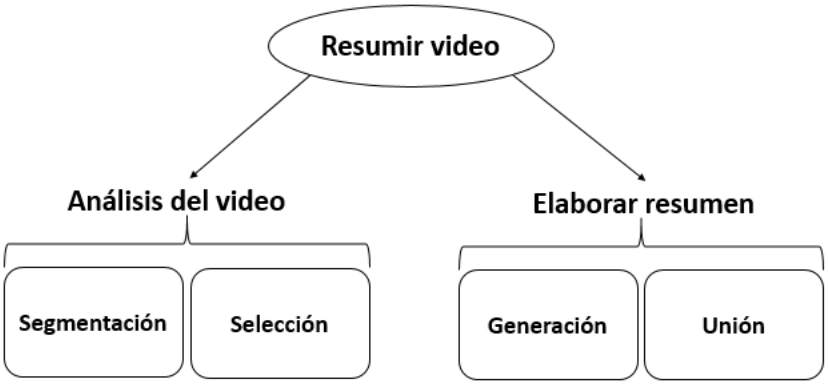

Fig. 3 Descripción del proceso

2.4.1. Segmentación en tomas

El algoritmo comienza cuando se recibe el video del cual se quiere obtener el resumen (Véase Fig. 4). Una vez recibido el video se pasa a decodificar y extraer los fotogramas del mismo. Para lograrlo se utilizan funcionalidades de la biblioteca OpenCV. Primeramente es necesario crear una estructura de tipo **VideoCapture** encargada de decodificar el video y un objeto de tipo **Mat** que contendrá el fotograma extraído. Luego se utiliza la función **read** de la estructura que contiene el video para guardar el próximo fotograma en el objeto. Este segmento de código debe de estar encapsulado dentro de un ciclo, de manera que se efectúe el proceso para todos los fotogramas del video.

Una vez capturados los fotogramas se pasa a la extracción de los histogramas de color. Primeramente se convierte el espacio de color de los fotogramas extraídos de RGB a HSV haciendo uso de la función

cvtColor de OpenCV. Se calcula el histograma para cada región de cada fotograma y su contiguo, se almacenan en estructuras de tipo **MatND**, y se calcula la distancia entre los vectores n-dimensionales resultantes mediante la fórmula 3. Se multiplican por la importancia establecida para cada región y se divide entre la sumatoria de las importancias para normalizar el resultado. Se establecen dos umbrales, máximo y mínimo, entre las comparaciones, y se tienen en cuenta las siguientes condiciones:

- Detectar todas las distancias que sobrepasen el umbral máximo como un posible cambio de toma y pasar a comprobar la similaridad SURF.
- Continuar la decodificación si la distancia es menor que el umbral mínimo.
- Comprobar la correlación si la distancia se encuentra entre los dos umbrales.

En caso de que la distancia (D) se encuentre en un rango de duda, o sea, el mínimo umbral < D < máximo umbral, se calcula el nivel de correlación entre los histogramas mediante la fórmula 2, teniendo en cuenta también la importancia de cada región para el par de fotogramas que se estén analizando. A diferencia de la comprobación anterior, la correlación da una medida de qué tan similares son los histogramas de color. Seguidamente se establecen dos umbrales, máximo y mínimo, y se tienen en cuenta las siguientes condiciones:

- Detectar como un posible cambio de toma si la correlación se encuentra entre los dos umbrales y se pasa a comprobar la similaridad SURF.
- Continuar la decodificación si la correlación es menor que el umbral mínimo.
- Se concatena la toma actual con la siguiente, o sea, se considera no existe cambio de toma, si la correlación es mayor que el umbral máximo.

El uso de la similaridad SURF garantiza la precisión de las técnicas basadas en histogramas de color, ya que estas últimas se ven afectadas por repentinos cambios de iluminación. Para la comparación se utiliza la fórmula 1, se establece un umbral mínimo y se tienen en cuenta las siguientes condiciones:

- Detectar como un cambio de toma si la similaridad SURF es menor que el umbral establecido.
- Se concatena la toma actual con la siguiente, o sea, se considera no existe cambio de toma, si la similaridad SURF es mayor o igual que el umbral.

Una vez terminado el ciclo de decodificación el video queda segmentado en las tomas que lo componen y seguidamente se pasa a filtrar las tomas no significantes para el resumen.

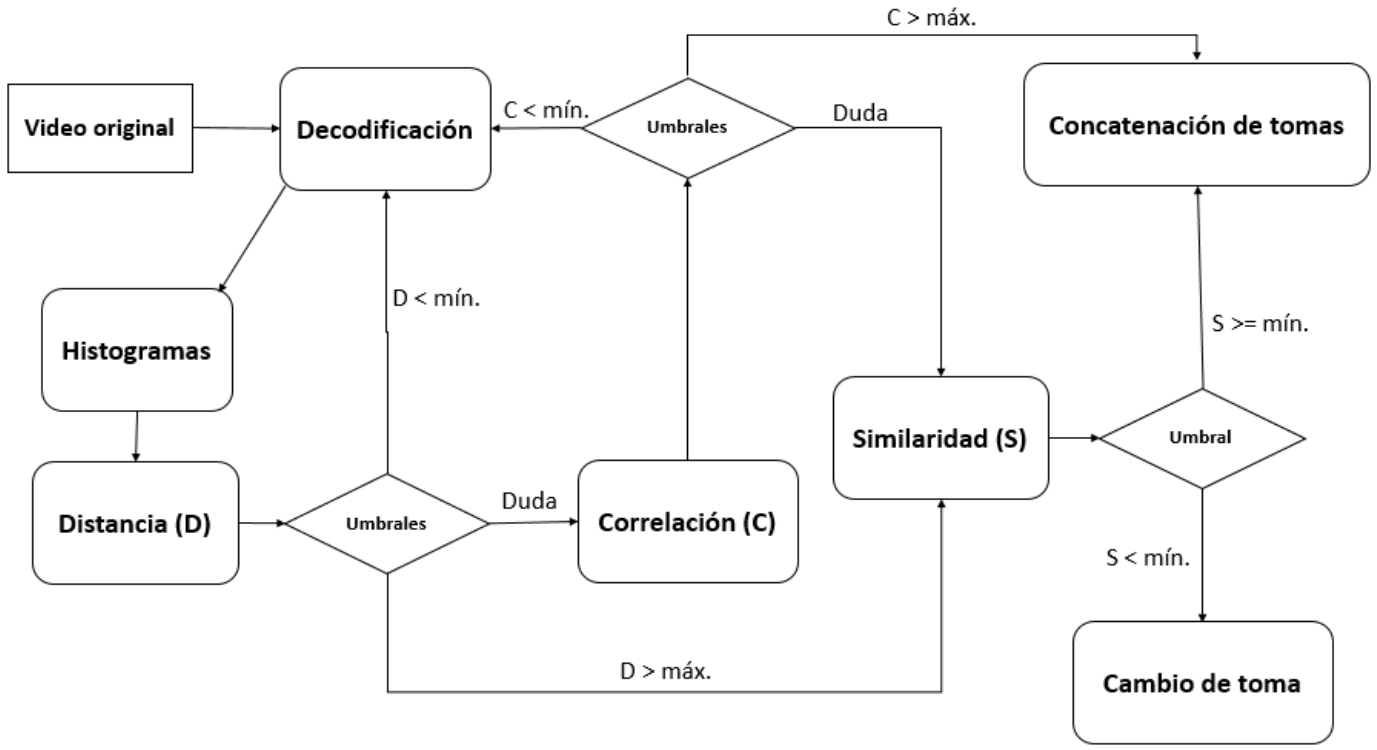

Fig. 4 Descripción de la segmentación de tomas

2.4.2. Filtrado de tomas y extracción del fotograma clave

Se considera que una toma no aporta información relevante para el resumen si contiene menos de 35 fotogramas, o sea, poco más de un segundo del video. También son descartadas las tomas con fotogramas vacíos, o sea, negros o blancos; las mismas se encuentran normalmente entre dos tomas unidas por una transición de desvanecimiento, o bien, puede tratarse de pantallas de créditos del video.

Seguidamente, toda toma relevante pasa por un proceso de extracción de su fotograma clave, el cual se explica en el epígrafe 1.3.4. Para la comparación de la similaridad de los contornos de dos fotogramas se siguen los siguientes pasos:

1. Convertir los fotogramas a escala de grises.
2. Aplicar una función de detección de contornos.
3. Expandir los contornos resultantes.
4. Calcular la similaridad de los contornos.

5. Comparar los resultados contra un umbral establecido.

El primer paso se logra utilizando la función **cvtColor** de OpenCV. El mismo prepara la imagen para efectuarle la detección de contornos **Canny**. El resultado es luego expandido mediante el método **dilate** para disminuir la sensibilidad al movimiento y se utiliza la fórmula 1 para dar una medida de similaridad.

Una vez finalizado el filtrado de las tomas y la extracción del fotograma clave se almacena en la base de datos la colección de posibles tomas a incluir en los resúmenes escalables.

2.4.3. Selección de las tomas

A partir del conjunto de tomas filtradas se hace necesario eliminar la redundancia de información. En la Fig. 5 se muestra visualmente el proceso explicado en el epígrafe 1.3.3, donde el fotograma clave de la toma aspirante a ser seleccionada es comparado con los fotogramas claves de las tomas seleccionadas.

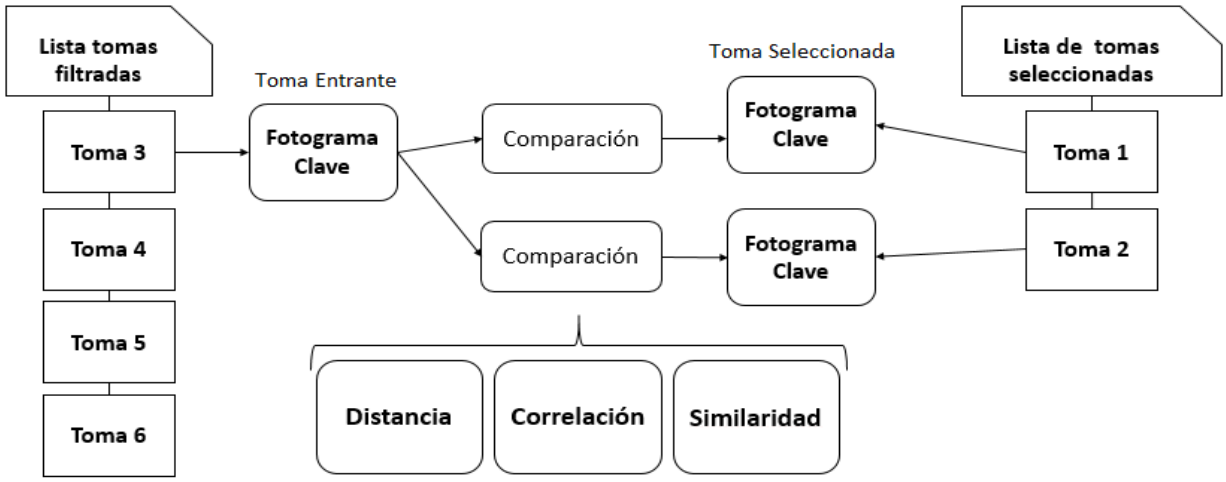

Fig. 5 Descripción de la selección de tomas

La comparación realizada es también efectuada por las medidas de distancia, correlación y similaridad SURF explicadas previamente, pero con umbrales diferentes establecidos específicamente para la selección.

La etapa de análisis concluye cuando se actualiza en la base de datos cuáles tomas serán incluidas en los diferentes resúmenes del video.

2.4.4. Generación del resumen y unión de las tomas

Para la generación del resumen se crea un *ranking* con las tomas más relevantes ordenadas según su puntuación. Se establece la longitud que se desea que tenga el video resumen resultante. Se seleccionan, comenzando por el principio del *ranking*, las tomas que contiene el mismo, hasta que se satisfaga la duración requerida. Habiendo seleccionado las tomas que formarán parte del resumen se ordenan siguiendo el flujo temporal del video original.

Luego, para la construcción de un resumen dinámico, se concatenan dichas tomas mediante el uso de una transición de corte simple, o bien, si se necesita un resumen estático, se muestran cada uno de los correspondientes fotogramas claves durante el tiempo establecido para los mismos en este tipo de resumen. Las tomas seleccionadas son extraídas del video original lo que permite que, teniendo la secuencia original y el componente, se pueda construir el resumen deseado sin necesidad de tener los segmentos previamente construidos, ya que se considera que almacenando cada segmento o toma para cada audiovisual se duplicaría el volumen de información de video, lo que reduciría la capacidad del servidor de almacenamiento considerablemente.

Esta etapa puede ser utilizada para obtener resúmenes de cualquier longitud deseada, por tanto, se puede repetir todas las veces que sea necesario.

2.5. Diseño del componente

Para definir cómo se va a desarrollar la aplicación se deben tener en cuenta los siguientes puntos:

2.5.1. Estilo arquitectónico

Los estilos arquitectónicos son modelos que describen a los patrones y las interacciones entre ellos. Estos ayudan a lograr un tratamiento estructural que se aplican más bien en la teoría, la investigación académica y la arquitectura en el nivel de abstracción más elevado. (Cavenago y otros, 2007)

El estilo seleccionado para efectuar el diseño de la aplicación es el de **llamada y retorno**, lo cual garantiza la fácil modificación y la escalabilidad de las aplicaciones del componente.

Patrón de arquitectura

Los patrones de arquitectura son la herramienta básica de un arquitecto a la hora de dar forma a la arquitectura de una aplicación. Un patrón de arquitectura se puede entender como un conjunto de principios

que definen a alto nivel un aspecto de la aplicación. También se entienden como indicaciones abstractas de cómo dividir en partes el sistema y de cómo éstas partes deben interactuar. Dentro de los patrones de arquitecturas que se pueden utilizar en el estilo seleccionado se encuentra la arquitectura **N-Capas**. (Llorente y otros, 2010)

El patrón de arquitectura en capas (N-Capas) se basa en una distribución jerárquica de los roles y las responsabilidades para proporcionar una división efectiva de los problemas a resolver. Los roles indican el tipo y la forma de interacción con otras capas y las responsabilidades de la funcionalidad que implementan. (Somasegar y otros, 2009)

Sus principales características son que los componentes de cada capa se comunican con los componentes de otras capas a través de interfaces bien conocidas y que cada nivel agrega las responsabilidades y abstracciones del nivel inferior. (Llorente y otros, 2010)

La utilización de esta arquitectura ofrece varias ventajas. A continuación se hace referencia a las ventajas que se manifiestan en el componente (Llorente y otros, 2010):

- Aislamiento: se pueden realizar actualizaciones en el interior de las capas sin que esto afecte al resto de la aplicación.
- Rendimiento: distribuyendo las capas en distintos niveles físicos se puede mejorar la escalabilidad, la tolerancia a fallos y el rendimiento.
- Testeabilidad: cada capa contiene una interfaz bien definida sobre la que realizar pruebas y la habilidad de cambiar entre diferentes implementaciones de la misma.

Se dividieron las funcionalidades en tres capas (Véase Fig. 6):

- Interfaz: compuesta por los métodos que garantizan la comunicación con otros componentes.
- Lógica de negocio: en esta capa se ejecutan las funcionalidades principales del componente, la misma incluye todo el proceso de generación de resúmenes.
- Acceso a datos: provee las funcionalidades de acceso a datos.

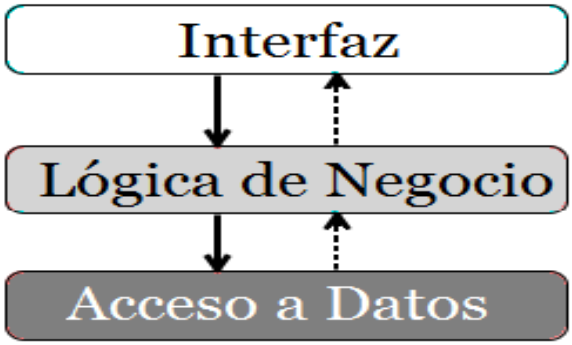

Fig. 6 Representación de la arquitectura en tres capas

2.5.2. Patrones de diseño

Un patrón es una pareja de problema/solución con un nombre y que es aplicable a otros contextos, con una sugerencia sobre la manera de usarlo en situaciones nuevas. (Larman, 2008)

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. Brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. Poseen una serie de elementos como: un nombre, el problema que indica cuándo aplicar el patrón y la solución que es la descripción abstracta del problema.

Patrones GRASP[20]

Con el propósito de desarrollar el componente para la elaboración de resúmenes de video se tuvieron en cuenta los patrones GRASP, debido a que describen los principios fundamentales de la asignación de responsabilidades a objetos expresados en forma de patrones.

Se utilizó el patrón **experto** debido a que plantea que se debe asignar la responsabilidad al experto en información, que en este contexto, sería la clase que cuenta con la información necesaria para cumplir la responsabilidad. (Larman, 2008)

[20] Patrones Generales de Software para Asignación de Responsabilidades.

Se empleó el patrón **creador**, el cual se encarga de asignar a una clase la responsabilidad de crear una instancia de otra. Es de gran importancia saber asignar la responsabilidad de crear instancias de una clase sólo a aquella que la requiera. (Larman, 2008)

Otro patrón utilizado es **alta cohesión,** que consiste en asignar una responsabilidad de modo que la cohesión siga siendo alta. La cohesión es la medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. (Larman, 2008)

El patrón **bajo acoplamiento,** que tiene como propósito evitar la dependencia excesiva entre las clases del sistema, garantizando que un cambio en una clase no provoque grandes cambios en otras. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, conque las conoce y recurre a ellas. (Larman, 2008)

Otro patrón que se aplica es el **controlador** ya que durante el diseño es necesario contar con una clase controladora; la cual tiene la responsabilidad de manejar los eventos del sistema. Generalmente los eventos del sistema son generados por actores externos. (Larman, 2008)

***Patrones GoF*[21]**

Los patrones **GoF** se clasifican según tres propósitos: creacional, estructural y de comportamiento. (Gamma y otros, 2004)

Los patrones de diseño creacionales se centran en resolver problemas acerca de cómo crear instancias de las clases de una determinada aplicación. Dentro de los mismos se utiliza el patrón **instancia única** debido a que restringe la creación de objetos pertenecientes a una clase a un único objeto. Esto garantiza que una clase solo contenga una instancia y un punto de acceso global a la misma. (Rojas y Olivares, 2010)

Los patrones de diseño estructurales son los que plantean las relaciones entre clases, las combinan y forman estructuras mayores. Dentro de los mismos se utiliza el patrón **fachada** debido a que simplifica el acceso a un conjunto de clases proporcionando una única clase que los programadores utilicen para comunicarse con dicho conjunto de clases. (Rojas y Olivares, 2010)

Los patrones de comportamiento tratan la interacción y la cooperación entre clases. Dentro de los mismos se utiliza el patrón **observador** el cual define una dependencia “uno a muchos” entre objetos, para que

[21] Pandilla de los cuatro, del inglés, *Gang of Four.*

cuando uno de ellos cambie su estado, todos los que dependan de él sean avisados y puedan actualizarse convenientemente. (Gamma y otros, 2004)

2.5.3. Clases del diseño

Una clase de diseño es una abstracción de una clase o construcción similar en la implementación del sistema y tiene operaciones, parámetros, atributos y tipos que son necesarios para su implementación en el lenguaje de programación elegido. El diagrama de clases de diseño (DCD) representa un nivel de detalle alto, pues se relaciona con el lenguaje de programación del cual se hará uso en la implementación del sistema. (Carvajal, 2010)

Los diagramas a continuación, muestran la distribución de las clases en el componente para cada CU, donde se evidencia la arquitectura en tres capas, sin mostrar sus atributos y métodos (Fig. 7 y Fig. 8; las versiones completas para cada caso de uso pueden encontrarse en el Anexo III).

En la capa “Interfaz” la clase **CInterfaz** utiliza el patrón fachada para proveer a los usuarios que utilicen el componente las funcionalidades de las demás clases con las que esta se relaciona, sin necesidad de que el usuario acceda directamente a las mismas. Dicha clase **CIntefaz** es capaz de brindar actualizaciones, en intervalos definidos, del estado y el porciento en que se encuentra el proceso de generación de resúmenes, haciendo uso del patrón observador.

En la capa “Lógica de negocio” se observa como la clase controladora principal **CResumidorVideoControladora** implementa el patrón instancia única para **CResumidorVideo** donde esta última muestra el uso de patrón controlador, ya que es la encargada de proveer los métodos necesarios para manejar los eventos del sistema.

En la *Fig. 7* se puede observar que la clase **CResumidorVideo** se compone por **CVideoProcesador**, **Ranking** la cual hereda de la clase lista (**ListaSE**) de tomas (**CToma**), y **CConexionControladora** (la cual agrega **CConexionSQL** implementando el patrón instancia única) desde la capa de “Acceso a datos”, donde la primera se encarga de las dos primeras etapas del proceso devolviendo una lista rankeada de tomas y con la última se garantiza la persistencia de este análisis en una base de datos.

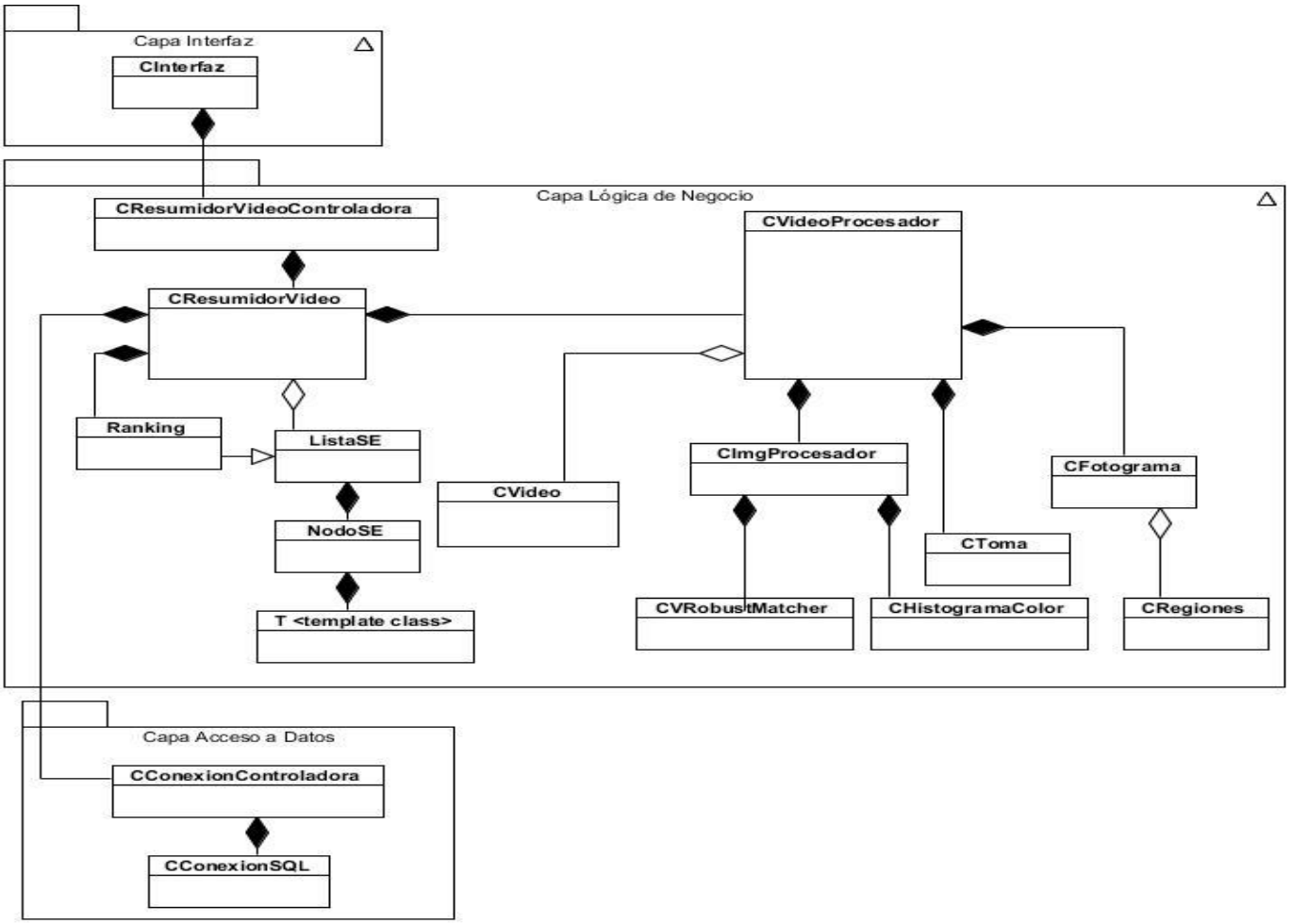

Fig. 7 Diagrama de clases de diseño del CU "Análisis del video"

La clase **CVideoProcesador** agrega el uso de la clase **CVideo** para las operaciones de lectura y escritura de fotogramas (**CFotograma**), y se compone de la clase **CImgProcesador** que es la encargada de las operaciones de cálculo de distancia, de la similaridad, de los distintos descriptores (**CHistogramaColor, CVRobustMatcher** para la detección SURF; ambos, ejemplos de clase que utilizan el patrón experto) y de comparación entre fotogramas, teniendo en cuenta la importancia de sus regiones (**CRegiones**). La existencia de la clase **CRegiones** demuestra el uso del patrón alta cohesión.

Finalmente, en la *Fig. 8* se observa el uso de la clase **CGeneradorResumenes** (ejemplo del patrón creador donde **CResumidorVideo** es el encargado de instanciarla), la cual provee los métodos de generación de

resúmenes estáticos y dinámicos; esta clase pone de manifiesto el patrón bajo acoplamiento ya que dicha clase recurre al uso de la menor cantidad de clases posibles.

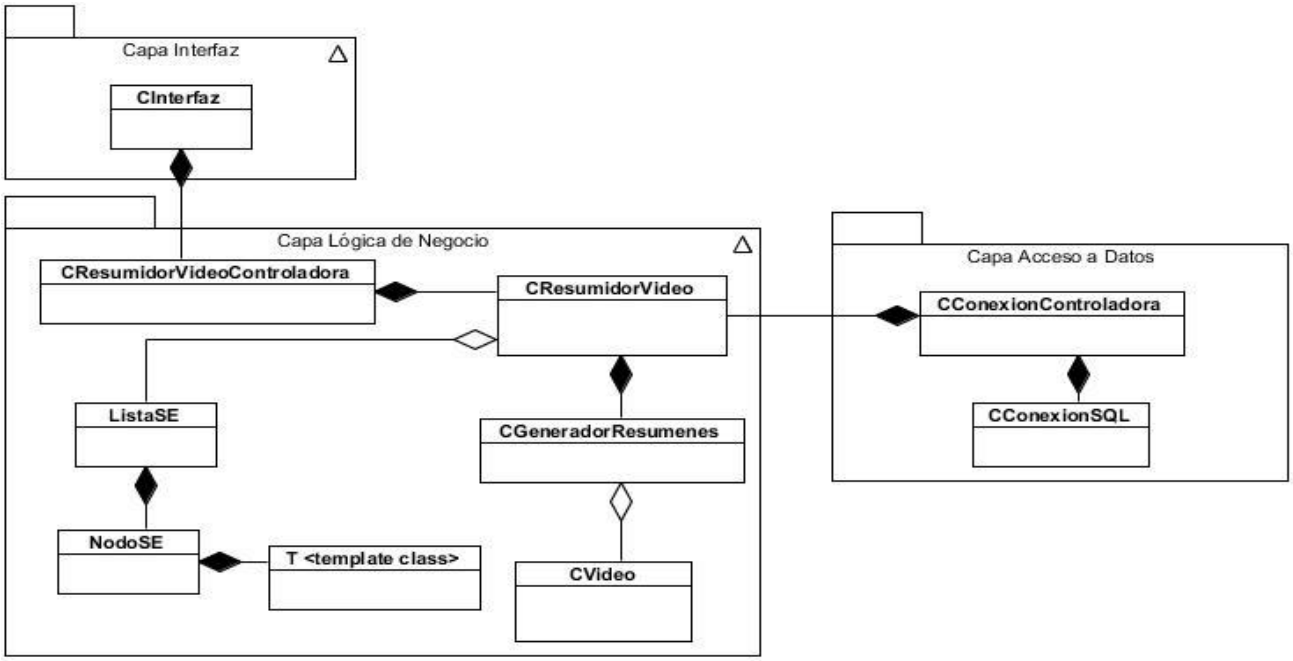

Fig. 8 Diagrama de clases de diseño del CU "Elaborar resumen"

2.5.4. Diagrama de secuencia del diseño.

Un diagrama de secuencia es una forma de diagrama de interacción que muestra los objetos como líneas de vida a lo largo de la página y con sus interacciones en el tiempo, representadas como mensajes dibujados como flechas desde la línea de vida origen hasta la línea de vida destino. Los diagramas de secuencia no están pensados para mostrar lógicas de procedimientos complejos. (Jacobson y otros, 2005)

Los diagramas de secuencia pertenecientes al diseño del componente pueden encontrarse en el Anexo IV.

2.6. Conclusiones del capítulo

En este capítulo se presentaron los principios de diseño que determinaron las bases de la arquitectura propuesta. Se arribó a las siguientes conclusiones:

1. El levantamiento de requisitos funcionales, adecuado y con calidad, garantiza un desarrollo de software también con calidad y eficiencia, ya que los mismos son la base de las funcionalidades del sistema y de no definirse correctamente, este último no podrá satisfacer las necesidades del cliente.
2. La selección de una arquitectura en capas permite tener una correcta estructura para los elementos de software; su empleo simplifica su complejidad y organiza los elementos del mismo.
3. La utilización del patrón de arquitectura en tres capas asegura un correcto aislamiento en la solución final.
4. La elaboración de los artefactos que genera la metodología RUP permite un mejor dominio del sistema a implementar en cada fase del desarrollo de software.
5. La utilización de los patrones de diseño (alta cohesión, bajo acoplamiento, controlador, experto, creador e instancia única) asegura estandarizar la propuesta de solución, y obtener una arquitectura basada en buenas prácticas para no cometer errores ante la solución de problemas ya conocidos y solucionados anteriormente.

3. Capítulo III: Implementación y prueba

Introducción

En el presente capítulo se desarrolla el modelo de implementación compuesto por los diagramas de despliegue y el diagrama de componente. Finalmente, una vez implementada la propuesta se hace necesaria la utilización de varias pruebas para encontrar posibles errores, conocer las limitaciones del software, entre otras. Las pruebas utilizadas para validar el componente son las conocidas como pruebas del sistema.

3.1. Modelo de implementación

El modelo de implementación no es más que la representación de la composición física de la implementación del sistema, comprendido por los subsistemas y componentes del mismo. Con el objetivo de describir la relación que existe entre estos componentes y subsistemas se efectuó el diagrama de despliegue. Dicho artefacto tiene una gran importancia porque permite a los desarrolladores comprender el funcionamiento del sistema desde el punto de vista de sus componentes y sus relaciones.

3.1.1. Diagrama de despliegue

El diagrama de despliegue, (Véase Fig. 9), muestra las relaciones físicas de los distintos nodos que componen un sistema o subsistema y el reparto de los mismos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria. (Marca Huallpara, 2010)

El nodo servidor debe ser una estación con las características especificadas en los requisitos no funcionales. El mismo debe conocer la ubicación de un servidor de archivos audiovisuales, para obtener el contenido de video a procesar, que especifique el nodo cliente. La comunicación con el nodo cliente sería efectuada a través de los protocolos XML-RPC[22], SOAP[23] o ICE[24].

[22] Forma fácil y rápida de hacer llamadas a procedimientos. Convierte la llamada en un XML y lo envía usando HTTP, la respuesta es también es un XML.
[23] Protocolo de comunicación que permite la comunicación mediante el intercambio de archivos XML con un formato definido.
[24] Plataforma orientada a objetos para la comunicación entre aplicaciones que provee de herramientas, APIs y biliotecas para apoyar la construcción de aplicaciones de cliente-servidor orientada a objetos.

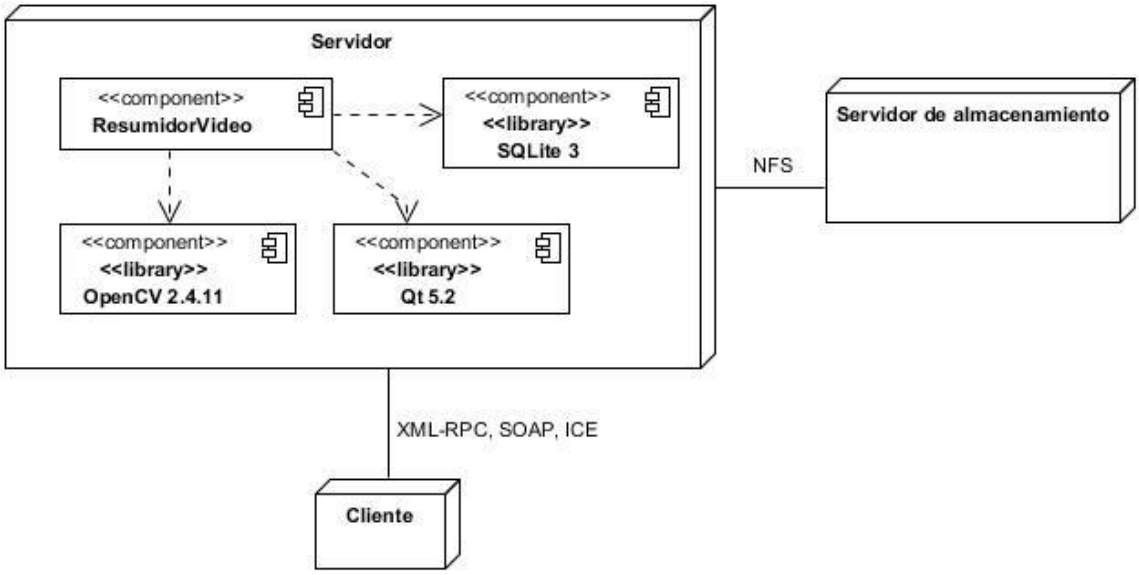

Fig. 9 Diagrama de despliegue

3.1.2. Diagrama de componentes

Un diagrama de componentes muestra las dependencias lógicas entre los componentes de una solución, sean estos de código fuente, componentes del código binario o componentes ejecutables. Su creación asegura un mejor entendimiento y estructuración de la implementación. (Jacobson y otros, 2005)

El diagrama a continuación, (Véase Fig. 10), muestra las relaciones entre el ejecutable, las bibliotecas y los datos usados solamente, ya que el código fuente ha sido estructurado según las clases, y las relaciones entre estas ya fueron especificadas en el “Diagrama de clases del diseño”.

3.2. Modelo de datos

El modelo de datos describe la representación lógica y física de los datos persistentes usados por la aplicación. Es una colección de conceptos que se emplean para describir la estructura de una base de datos. La misma incluye entidades, atributos y relaciones. (ALEGSA.COM.AR, 2015)

Para este caso el modelo se obtuvo a partir de las necesidades del componente quedando representado como se muestra en la Fig. 11.

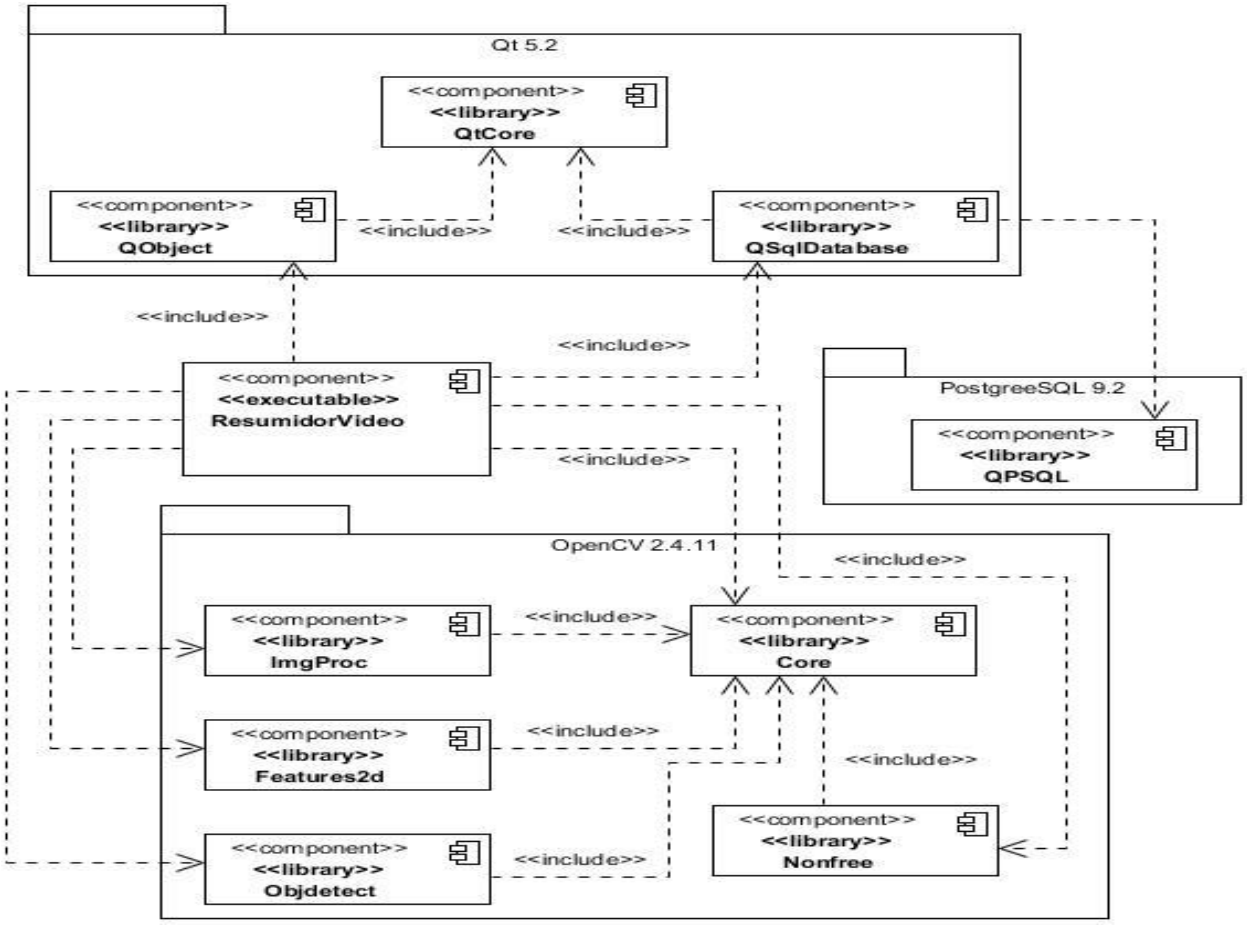

Fig. 10 Diagrama de Componentes

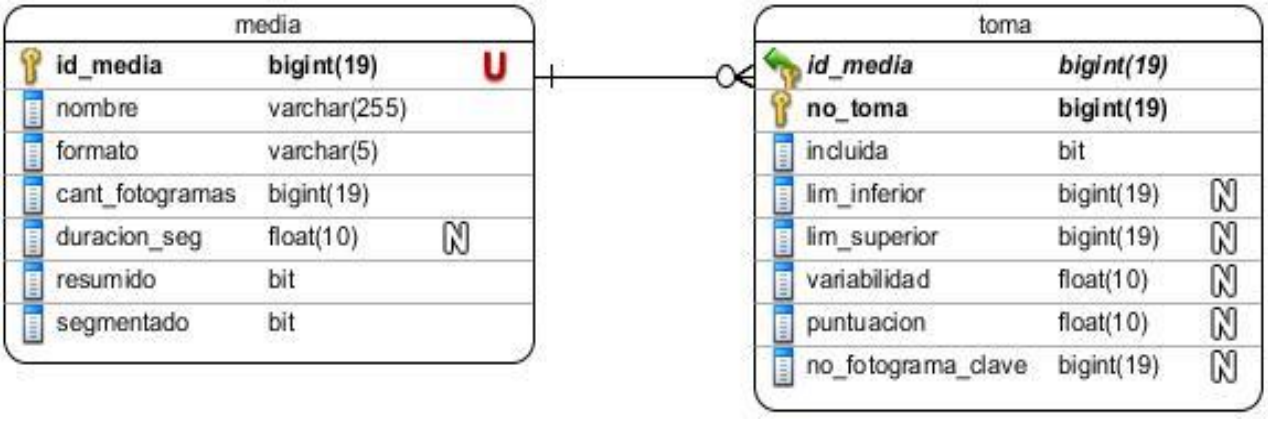

Fig. 11 Modelo de datos

3.3. Pruebas del sistema

Según (Pressman, 2012), las pruebas de software son un elemento crítico para la garantía de la calidad de software y representan una revisión final de las especificaciones del diseño (Caja negra) y de la codificación (Caja blanca).

La prueba de caja negra se centra principalmente en los requisitos funcionales del software. Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos. (Richards y otros, 2006)

Con el objetivo de verificar el ingreso, procesamiento y recuperación apropiada de los datos se realizaron pruebas al sistema enfocadas en requisitos funcionales tomados directamente de casos de uso y funciones de negocio.

3.3.1. Soporte de formatos

Con el objetivo de comprobar el comportamiento de la solución implementada ante videos de características distintas se conformó un caso de prueba que tiene como objetivo probar 5 videos de características distintas (Véase Tabla 7) para comprobar el comportamiento de la solución implementada. Entre las características que los distinguen se encuentran el formato, codificador, *bitrate*[25], fotogramas por segundo (FPS) y el tamaño de los mismos.

Durante la investigación se identificó la necesidad de utilizar diferentes parámetros para flexibilizar el algoritmo implementado. Dichos parámetros fueron configurados para las pruebas, (Véase Tabla 8). En la Tabla 9 se muestra el resultado de las pruebas de soporte de formato para la ejecución del componente.

Tabla #7 Descripción de los videos de prueba

No.	Nombre	Contenedor	Cantidad de Fotogramas / Duración en segundos(s)	Dimensiones	Contenido
1	La teoría del Big Bang	Mkv	5760 / 240s	856x480	Segmento de serie.
2	Amigos	Mpg	9498 / 316s	352x240	Segmento de serie.
3	El experto	Webm	11402 / 456s	852x476	Cortometraje.
4	Trompetas	mp4	5685 / 237s	856x480	Video musical.
5	El Coro - Teléfono	Avi	1581 / 66s	632x352	Segmento de serie. Musical.

[25] Frecuencia de bits por segundo

Tabla #8 Descripción de los parámetros de configuración del componente

Nombre	Dominio	Descripción	Epígrafes de referencia	Rango [Mín. - Máx.]	Valor usado
desdeFotograma	Natural	Empezar a procesar desde el fotograma indicado.	-	0 – *	0
minUmbralDistDuda-maxUmbralDistDuda	Real	Umbrales de distancia para segmentación. Min-Máx.	1.3.2, 2.4.1	0 – 1	0.13 – 0.42
minUmbralCorrDuda-maxUmbralCorrDuda	Real	Umbrales de correlación para segmentación. Min-Máx.	1.3.2, 2.4.1	0 – 1	0.906 – 0.987
minUmbralSimDuda	Real	Umbral de similaridad SURF para segmentación.	1.3.2, 2.4.1	0 – 1	0.43
umbralDistMin-unbralDistMax	Real	Umbrales de distancia para selección. Min-Máx.	1.3.3, 2.4.3	0 – 1	0.08 – 0.20
umbralCorr	Real	Umbral de correlación para selección.	1.3.3, 2.4.3	0 – 1	0.91
umbralSim	Real	Umbral de similaridad SURF para selección.	1.3.3, 2.4.3	0 – 1	0.10
divColores	Natural	Factor de reducción de colores	-	0 – 255	18
ladoLargo	Porciento	Por ciento del total para lado del centro	1.4.4, 2.4.1	0 – 98	63
umbralCanny1-umbralCanny2	Natural	Rango de detección de contornos del algoritmo Canny	1.3.2, 1.3.4, 2.4.2	0 – 255	125 – 150
ordenDistancia	Natural	Orden de Minkowski	1.3.2	1 – 2	2
siAltoMasAncho	Natural	Máximo de suma de las dimensiones del fotograma.	-	1000 – 5000	1200
factorReduccion	Natural	Factor de reducción de las dimensiones del fotograma.	-	1 – 10	2
binesHistograma	Natural	Cantidad de colores a tener en cuenta en el histograma	1.3.2	1 – 255	56
impCentral-impBase-impLaterales-impTriangular-impEsquinasSup	Real	Importancias de las regiones del fotograma.	1.4.4	0 – 5	2 – 1.5 – 0.5 – 0.1 – 0.01
dinamico	Binario	Modalidad de resumen.	1.1.4	-	-
porciento	Natural	Por ciento de la duración total para el resumen deseado.	-	1 - 100	-
tiempoPorFCSeg	Natural	Cantidad de tiempo que es mostrado cada fotograma clave	1.1.4, 1.3.4	1 – 5	1

Tabla #9 Resultado de la prueba de formato.

No.	Formato	Codificador	Bitrate(KB/s)	FPS	Resultado
1	Matroska	V_MPEG4	1152	24	Procesado satisfactoriamente
2	MPEG	MPEG-1	1150	30	Procesado satisfactoriamente
3	WebM	VP-8	1782	25	Procesado satisfactoriamente
4	MPEG-4	H263	1460	24	Procesado satisfactoriamente
5	AVI	XVID	1035	24	Procesado satisfactoriamente

3.3.2. Algoritmo de detección de tomas

La segmentación del video en tomas representa un paso crítico dentro del proceso de generación de resúmenes debido a que como resultado de esta tarea se obtienen las unidades básicas para efectuar el procesamiento de resumen. Las pruebas expuestas a continuación buscan valorar la fidelidad de las detecciones durante este proceso. La forma más utilizada para medir el buen funcionamiento de los algoritmos de detección de cambios de tomas consiste en calcular la métrica *recall*[26] (R) y la métrica *precision*[27] (P). (Hernández Heredia, 2013)

$$(5) \qquad R = \frac{DC}{DC+FN}$$

$$(6) \qquad P = \frac{DC}{DC+FP}$$

Donde DC representa los verdaderos positivos o detecciones correctas, FN son los falsos negativos o cambios de toma que debía detectar el sistema y no detectó, FP son los falsos positivos o cambios de toma que el sistema detectó y no debía detectar.

Tabla #10 Resultado de las medidas de recall y presicion.

Video	Cortes	Detectados	DC	FN	FP	*Recall*	*Precision*
1	66	70	66	0	4	1	0.943
2	85	82	82	3	0	0.965	1
3	91	97	91	0	6	1	0.938
4	86	91	86	0	5	1	0.945
5	23	23	23	0	0	1	1

El componente presentó dificultad en la detección de efectos de edición disolver y barrido, así como algunos efectos de cambios de iluminación repentinos.

[26] Indicador de los cortes entre tomas que no detecta el algoritmo.
[27] Indicador de los cortes entre tomas incorrectos o falsos que detecta el algoritmo.

3.3.3. Selección de tomas

La selección de tomas es también crucial para lograr el balance necesario entre cobertura de información y placer visual para el usuario. Luego de la segmentación del video de prueba No. 5, (Véase Tabla 6), con los parámetros descritos en la Tabla 8, la ejecución del método propuesto seleccionó, de las 24 tomas detectadas, 12 significantes. Las tomas eliminadas contenían información redundante. A continuación se muestran los fotogramas clave de las 12 tomas seleccionadas y posibles a incluir en un resumen de dicho video, (Véase Fig. 9).

Se considera que los fotogramas seleccionados por el algoritmo brindan un buen entendimiento sobre la actividad humana manifestada en el video de prueba elegido. A partir de dicho entendimiento es posible al menos, identificar y aprobar el contenido de este material audiovisual.

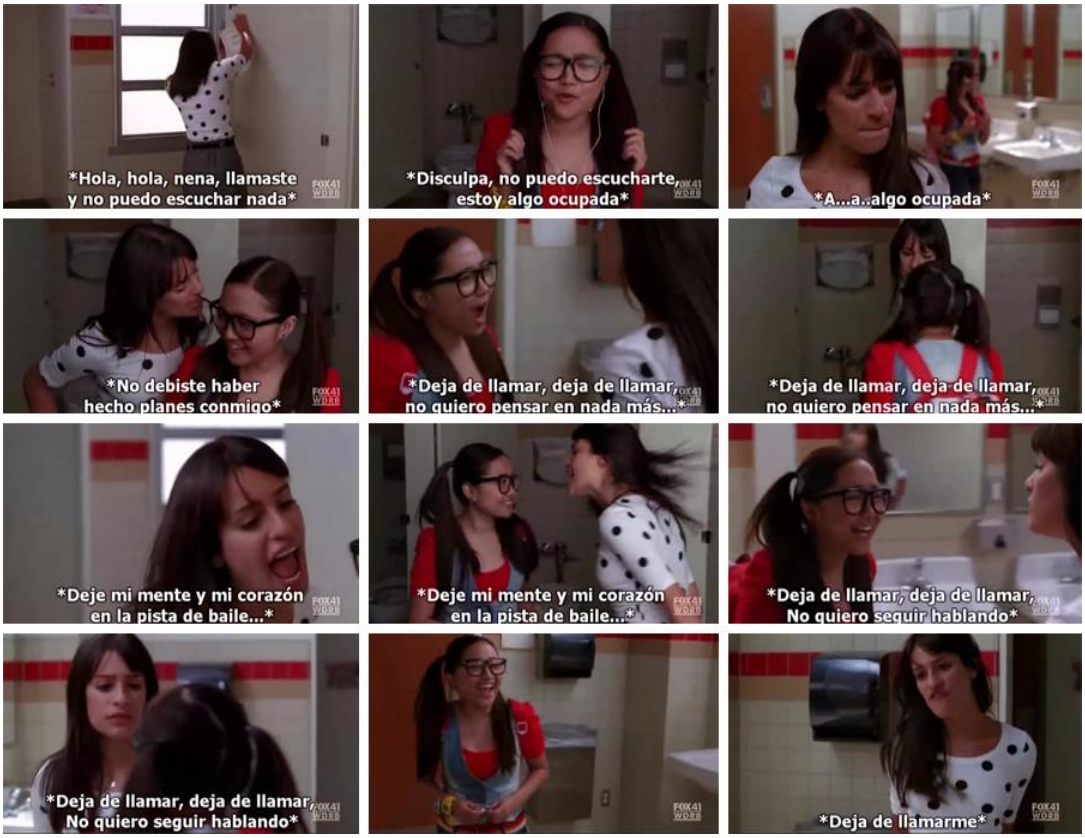

Fig. 12 Fotogramas clave de las tomas seleccionadas

3.3.4. Rendimiento

Se conformó un caso de prueba de rendimiento para mostrar el resultado del componente en cuanto al tiempo, en segundos, requerido para la obtención de resúmenes de distintas escalas y modalidades de una secuencia de video de 10 minutos. Se ejecutó la prueba sobre un procesador Intel Core i3 a 1.9 GHz. Las mediciones fueron registradas en la Tabla 11.

Tabla #11 Tiempo de procesamiento (en segundos) para una secuencia de 10 minutos

Análisis [Segmentación(248) + Selección(33)]	**Elaboración**							
	Resumen estático (#)		***Resumen dinámico (%)***					
	5	**10**	**1**	**5**	**10**	**20**	**50**	**100**
281	0.52	1.42	1.21	2.01	2.93	3.50	5.23	11.05

Evaluando las soluciones similares analizadas en el epígrafe 1.4, es posible observar que el componente propuesto muestra mejores valores promedio para las medidas *recall (0.99)* y *presicion (0.96)*, (Véase Tabla 12), pero posee un rendimiento inferior a los de las soluciones (Herranz y Martínez, 2010) (Díaz Berenguer, 2014).

Tabla #12 Promedio de valores de las medidas recall y presicion de las soluciones similares

Referencia	***Recall***	***Precision***
(Herranz y Martínez, 2010)	0.81	0.93
(López Meneses, 2013)	0.98	0.93
(Díaz Berenguer, 2014)	0.86	0.89

3.4. Conclusiones del capítulo

Después de la realización las pruebas del sistema se arribaron a las siguientes conclusiones:

- Se realizaron satisfactoriamente las pruebas de compatibilidad con diferentes formatos lo que evidencia que es posible utilizar el componente para videos con distinta codificación.

- La segmentación del video mostró valores por encima del 93% en las medidas de *precision* y *recall* y el método de selección implementado balancea cobertura de información y el placer visual lo que garantiza una correcta etapa de análisis.

Conclusiones generales

Una vez culminada la investigación asociada al desarrollo del componente se concluye que:

- El uso de métodos teóricos permitió la creación de las bases para el desarrollo de la presente investigación científica.
- El estudio de las soluciones similares permite determinar características positivas para ser tomadas como premisas en el desarrollo de componentes de software.
- La selección de las tecnologías y herramientas en función de sus características permitió una mejor integración con otros sistemas del centro GEYSED.
- La fase de diseño permitió crear las bases para la implementación del componente, evaluando y restringiendo cada posibilidad de error.
- Las pruebas del sistema presentaron resultados satisfactorios por lo tanto se concluye que el componente soluciona la situación problemática planteada.

Recomendaciones

Durante el desarrollo del componente de software surgieron algunas ideas teniendo en cuenta que la solución constituye una primera versión que estará sujeta a futuros cambios, por lo que se recomienda:

- Implementarle al componente la posibilidad de la elección entre diferentes algoritmos de segmentación, selección y unión añadidos como plugins.
- Integrar el componente a la plataforma de publicación de medias del centro GEYSED.

Bibliografía y referencias bibliográficas.

ALEGSA.COM.AR. 2015. ALEGSA.COM.AR. [En línea] 2015. [Citado el: 1 de 29 de 2015.] http://www.alegsa.com.ar/Dic/framework.php.

Ariza Rojas, Maribel y Molina García, Juan Carlos. 2004. Introducción y principios básicos del desarrollo de software basado en componentes. 2004.

Arteaga Domínguez, Giovanna, y otros. 2011. Clasificación automática de imágenes de cubiertos. [En línea] 2011. [Citado el: 28 de 1 de 2015.] http://www.acmor.org.mx/sites/default/files/102cubiertos.pdf.

Babaguchi, N. 2000. Towards abstracting sports video by highlights. New York : s.n., 2000.

Boullosa García, Óscar. 2011. Estudio comparativo de descriptores visuales para la detección de escenas. . 2011.

Bradski, G. y Kaehler, A. 2008. Learning OpenCV: Computer Vision with the OpenCV Library. 2008.

Bradski, Gary y Kaebler O'Reilly, Adrian. 2008. *Computer vision with the OpenCV library.* 2008.

C++. 2015. C Plus Plus. [En línea] 2015. [Citado el: 20 de 1 de 2015.] http://www.cplusplus.com/.

Canny, John. 1986. A computational approach to edge detection. Pattern Analysis and Machine Intelligence, IEEE Transactions on, (6):679–698. 1986.

Carvajal, Erlys. 2010. Análisis y diseño del subsistema de Análisis de Resultados de un Simulador de Procesos Químicos. La Habana : s.n., 2010.

Cavenago, Adriana, Almeira, Sandra y Perez, Vanina. 2007. Arquitectura de Software: Estilos y Patrones. 2007.

Chandrakar, N y Bhonsle, D. 2012. Study and comparison of various image edge detection techniques. s.l. : International Journal of Managment, IT and Engineering, 2(5):499–509., 2012.

Choi, Seung-Seok, Cha, Sung-Hyuk y C, Charles. 2010. A Survey of Binary Similarity and Distance Measures. 2010.

Communitic International. 2013. Kioskea.net. [En línea] 2013. [Citado el: 1 de 4 de 2015.] http://es.kioskea.net/contents/bdd/bddintro.php3.

Díaz Berenguer, Abel. 2014. Método para la generación automática de resúmenes escalables de videos. s.l. : Tesis de Maestría, 2014.

Digitales, Señales. 2008-2009. *Conferencia: "Características del video. Conceptos".* La Habana, Cuba : UCI, 2008-2009.

Fuentes, Lidia y Vallecito, A. 2004. Una Introducción a los Perfiles UML. 2004.

Ginestá, Marc Gibert y Pérez Mora. , Oscar. 2011. Scribd. [En línea] 2011. [Citado el: 2 de 4 de 2015.] http://es.scribd.com/doc/65741986/postgreSQL.

Gonzalez Diez, Héctor Raúl, Pupo Rodriguez, Luis Angel y Medina Leyva, Yoandri. 2012. Uso del HOG para un modelo de resumen automático de videos. La Habana, Cuba : s.n., 2012. 620. UCiencia.

Hernández Heredia, Y. 2013. Modelo para la detección y reconocimiento de acciones humanas en videos a partir de descriptores espacio-temporales. Tesis doctoral, Universidad de las Ciencias Informáticas. UCI. : s.n., 2013.

Herranz Arribas, Luis. 2010. A Scalable Approach to Video Summarization and Adaptation. Madrid : s.n., 2010. Tesis doctoral, Escuela Politécnica Superior de Ingeniería Informática, Universidad Autónoma de Madrid.

Herranz Arribas, Luis y Martínez, Jose Mario. 2010. A Framework for Scalable Summrization of Video. 2010.

Igual, Raúl. 2008. Tutorial de OpenCV. [En línea] 2008. [Citado el: 29 de 1 de 2015.] http://docencia-eupt.unizar.es/ctmedra/tutorial_opencv.pdf.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2005. El Lenguaje Unificado de Modelado. Madrid : Pearson Educacion, 2005.

Kruskal, J.B. 1964. Multidimensional scaling by optimizing goodness of fit to a non metric hypothesis. 1964.

Larman, Craig. 2008. UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado. Prentice Hall : s.n., 2008.

Leterlier, P. 2013. Introducción a RUP. s.l. : Departamento de Sistemas informáticos y Computación (DSIC), Universidad Politécnica de Valencia (UPV), 2013.

Lienhart, Rainer. 1997. Comparison of Automatic Shot Boundary Detection Algorithms. 1997.

Lienhart, Rainer, Pfeiffer, Silvia y Effe, Wolfgang. 1997. Video Abstracting. Mannheim, Germany : s.n., 1997. University of Mannheim.

Llorente, Cesar de la Torre, y otros. 2010. Guía de Arquitectura N- Capas orientada al Dominio con NET 4.0. s.l. : Krasis Press, 2010.

López Meneses, Maykel. 2013. Componente para elaborar resúmenes de video escalables automáticamente. 2013.

López, Antonio R. 2011. Extracción de Características de Imagen para Navegación de Robots Móviles. 2011.

Maldonado. 2008. AplicacionesEmpresariales.com. [En línea] 2008. [Citado el: 5 de 6 de 15.] http://www.aplicacionesempresariales.com/sqlite-el-motor-de-base-de-datos-agil-y-robusto.html.

Marca Huallpara, Hugo Michael y Quisbert Limachi, Nancy Susana. 2010. Diagrama de Despliegue, Análisis y diseño de sistemas. Universidad Salesiana Bolivia : s.n., 2010.

Mora, S. L. 2006. C++ paso a paso. [En línea] 2006. [Citado el: 28 de 1 de 2015.] http://gplsi.dlsi.ua.es/~slujan/materiales/cpp-muestra.pdf.

Nagasaka, A y Tanaka, Y. 1992. Automatic Video Indexing and Full-Video Search for Object Appearances, in Visual Database Systems II. 1992.

Nokia. 2011. Qt Creator IDE and tools. [En línea] 2011. [Citado el: 29 de 1 de 2015.] http://qt.nokia.com/products/developer-tools.

OPENCV.ORG. 2015. OpenCV. [En línea] 2015. docs.opencv.org/doc/tutorials/imgproc/histograms/histogram_comparison/histogram_comparison.html.

Paradigm, Visual. 2015. Visual paradigm. [En línea] 2015. [Citado el: 30 de 1 de 2015.] http://www.visual-paradigm.com/.

Pressman, Roger S. 2012. Software Engineering. A partitioner´s Approach. 2012.

Project, Qt. 2015. Qt Project. [En línea] 2015. [Citado el: 28 de 1 de 2015.] http://qt-project.org/.

Pupo Rodríguez, Luis Angel. 2011. Implementación de un componente para la separación automática de segmentos de videos integrado al Sistema de Captura y Catalogación de Medias. Universidad de las Ciencias Informáticas : s.n., 2011.

Quintana Rondón, Yoandri, Hernández Heredia , Yanio y Díaz Berenguer, Abel. 2010. Descriptores de video, sus aplicaciones en materiales audiovisuales. s.l. : Universidad de las Ciencias Informáticas, 2010.

RAE. Real Academia Española. [En línea] www.rae.es.

Richards, Adrion W., Branstad, Martha A. y Cherniavsky, John C. 2006. Validation, Verification and Testing of Computer Software. 2006.

Rojas, M.C. y Olivares, Juan Carlos. 2010. Patrones de Diseño. 2010.

Sáez Peña, Edmundo. 2006. Segmentación automática de video. Málaga : s.n., 2006. Tesis doctoral, Universidad de Málaga.

Salvat Salvat, Isabel, y otros. 2010. Utilización del vídeo para presentar los casos en el aprendizaje basado en problemas. 2010.

Somasegar, S., Guthrie, Scott y Hill, David. 2009. Microsoft Application Archiitecture Guide, patterns & practices. s.l. : Microsoft, 2009.

Swanberg, D, Shu, C. F y Jain, R. 1993. Knowledge Guided Parsing and Retrieval in Video Databases, in Storage and Retrieval for Image and Video Databases. 1993.

Techopedia. 2015. Techopedia. [En línea] 2015. [Citado el: 30 de 1 de 2015.] http://www.techopedia.com/definition/26860/integrated-development-environment-ide.

Thakre, Kalpana S. 2010. Video Match Analysis: A Comprehensive Content based Video Retrieval System Shimna Balakrishnan. 2010.

Torres, F. 2009. Integración del PMBOK al RUP para proyectos de Desarrollo de Software. 2009.

Truong, Ba Tu y Venkatesh, Svetha. 2007. Video Abstraction: A Systematic Review and Classification. Curtin : s.n., 2007. 3. University of Technology.

Tuytelaars, T y Mikolajczyk, K. 2008. Local invariant feature detectors: a survey. 2008.

Ueda, H, Miyatake, T y Yoshizawa, S. 1991. Impact: An Interactive Natural motion-picture Dedicated Multimedia Authoring System. 1991.

Valdés, Víctor y Martínez, José M. 2007. On-line Video Skimming Based on Histogram Similarity. 2007.

Valdés, Víctor y Martínez, José Mario. 2008. Binary tree based on-line video summarization. Madrid, España : s.n., 2008. Universidad Autónoma de Madrid.

Zabih, R, Miller, J y Mai, K. 1993. A Feature-Based Algorithm for Detecting and Classifying Scene Breaks. s.l. : Proc. ACM Multimedia 95, 1993.

Anexos

I. Efectos de transición

Ejemplos de algunos efectos de transición usados en la edición de video:

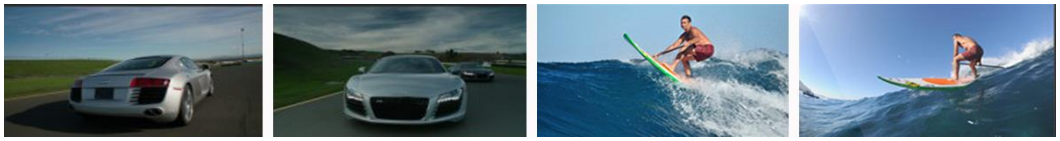

Fig. 13 Transición de corte

Fig. 14 Transición disolver

Fig. 15 Transición desvanecimiento de salida

Fig. 16 Transición desvanecimiento de entrada

Fig. 17 Transición de barrido

II. Estándar de codificación usado

Los estándares de codificación están enfocados a la estructura y apariencia física del código para facilitar la lectura, comprensión y mantenimiento del mismo. A continuación se presentan algunos de dichos estándares usados en la implementación del componente.

Se considera como identificador a los nombres de variables (arreglos, matrices, apuntadores), funciones, así como cualquier tipo de dato definido por el usuario (estructura, clase). Dichos identificadores deberán seguir las siguientes normas, además de las definidas por el propio lenguaje:

- Deberán tener un nombre significativo para que por su simple lectura pueda conocerse su función, sin tener que consultar manuales o hacer demasiados comentarios.
- Para nombres que se usen con frecuencia o para términos largos, se recomienda usar abreviaturas estándar para que éstos tengan una longitud razonable. Si usa abreviaturas deben manejar la misma lógica en todo el programa.

Especificaciones

Para distinguir palabras dentro del nombre deberá emplearse una letra mayúscula o un guion bajo (_), sin mezclar ambas formas en un mismo identificador. Ej.: tomasIncluidas, id_media.

Las funciones comenzarán con la primera letra en mayúscula. Ej.: void ObtenerDato().

Las variables serán declaradas con la primera letra en minúscula. Ej.: QString rutaEntrada.

Por su parte las clases serán identificadas por una C mayúscula al inicio. Ej.: CFotograma.

III. Diagramas de clases del diseño

CU “Resumir video”

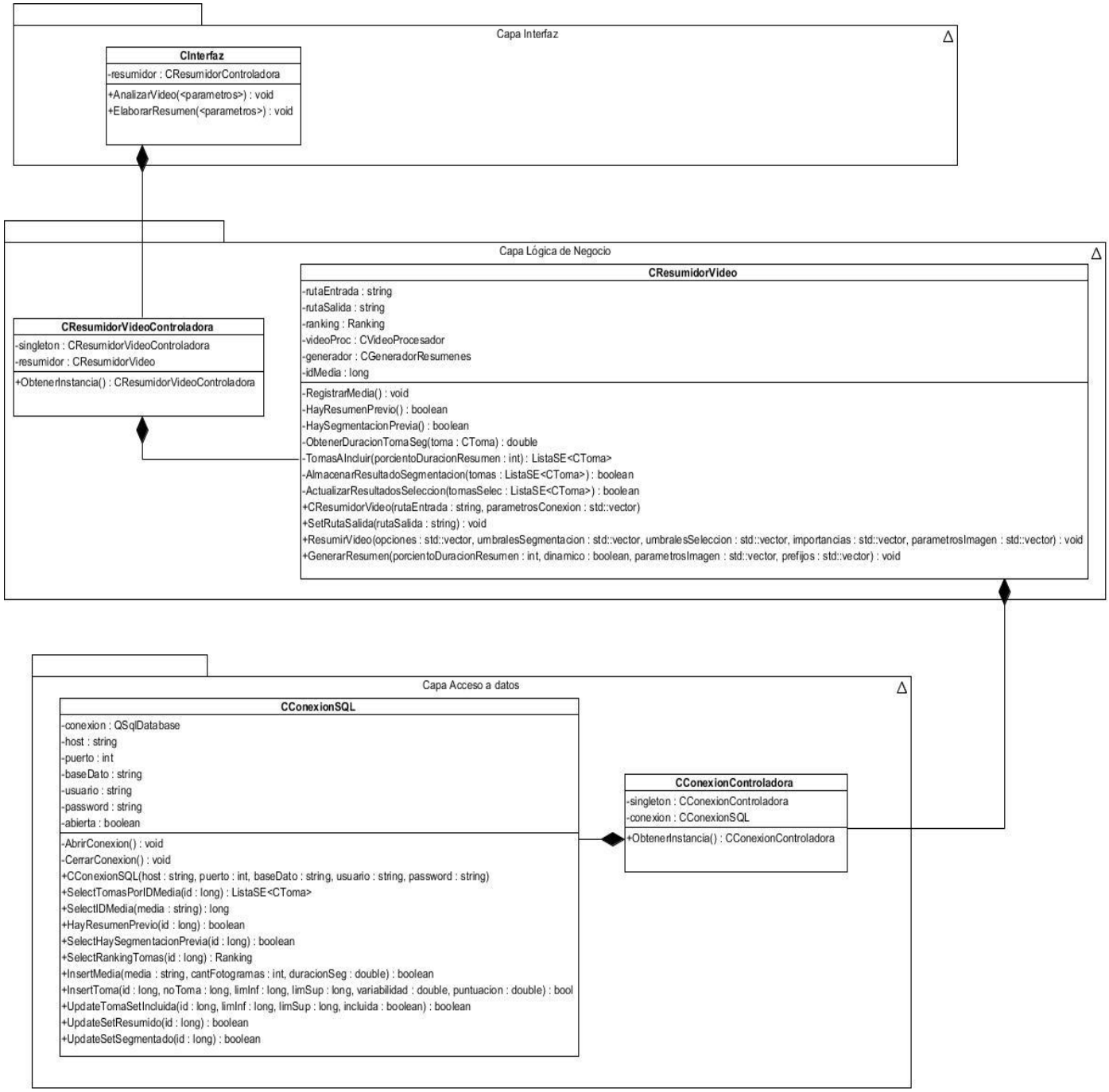

Fig. 18 Diagrama de clases del diseño completo del CU ¨Resumir video¨

CU “Análisis del video”

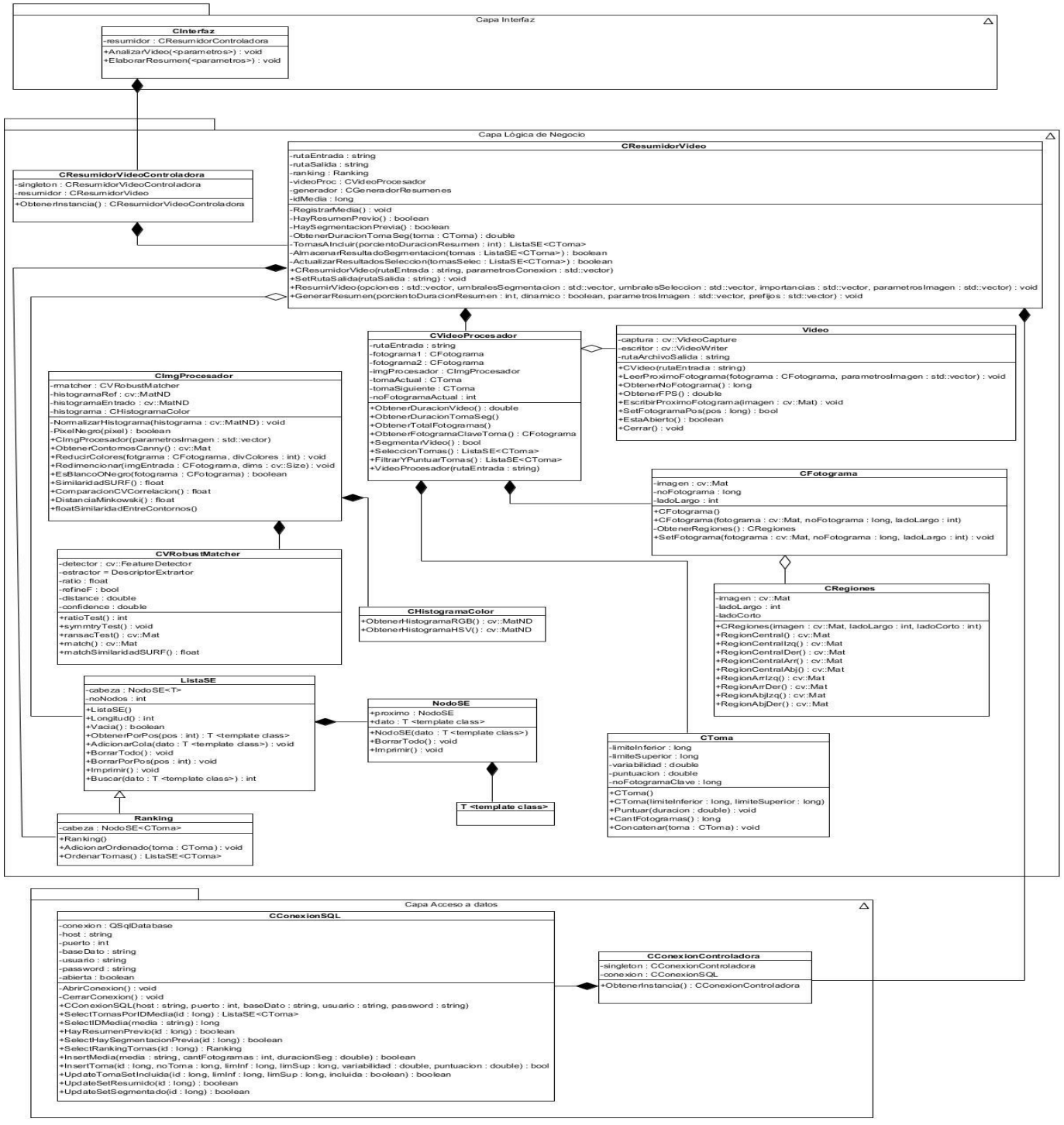

Fig. 19 Diagrama de clases del diseño completo del CU ¨Análisis del video¨

CU “Elaborar resumen”

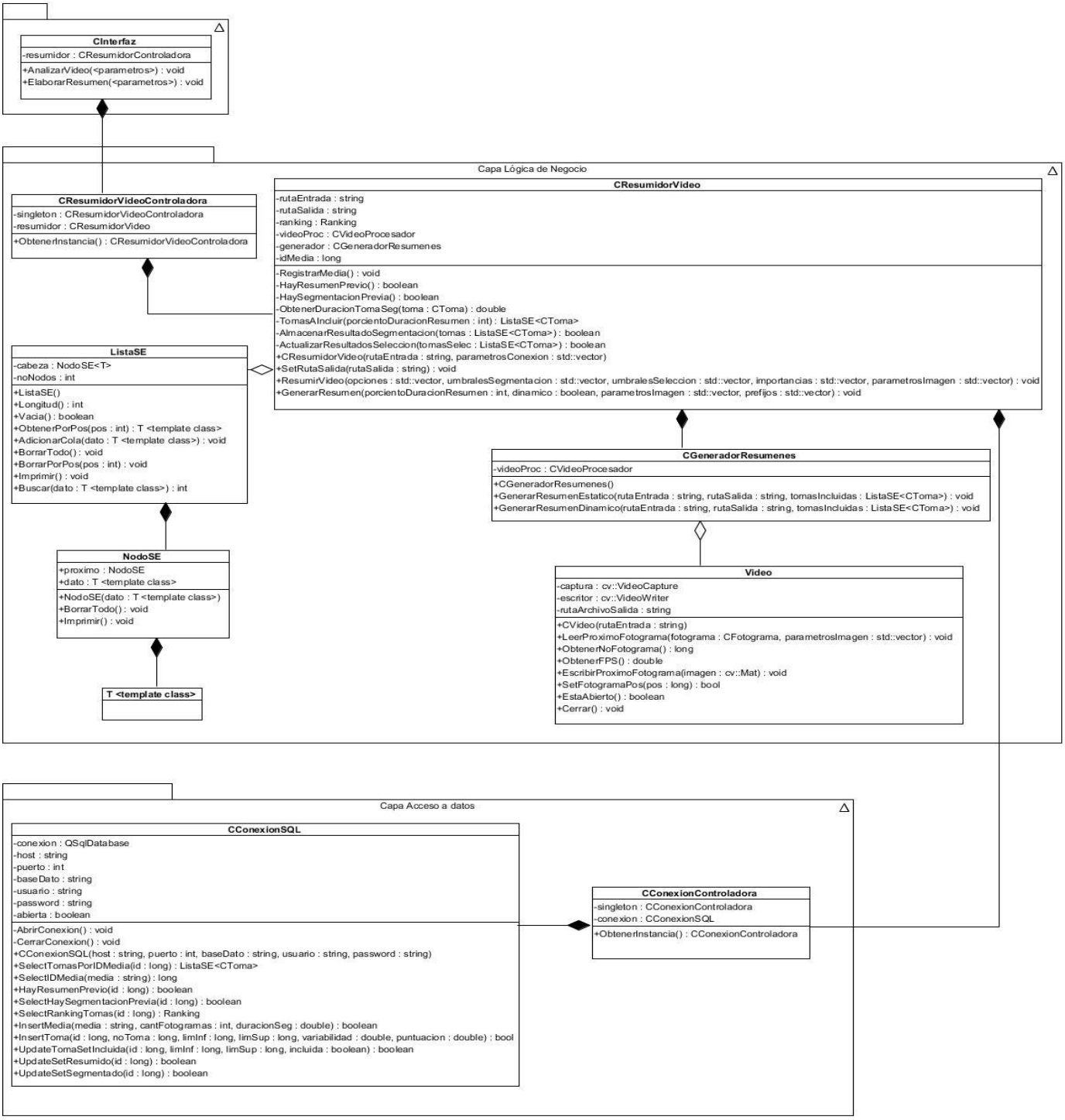

Fig. 20 Diagrama de clases del diseño completo del CU ¨Elaborar resumen¨

IV. Diagramas de secuencia

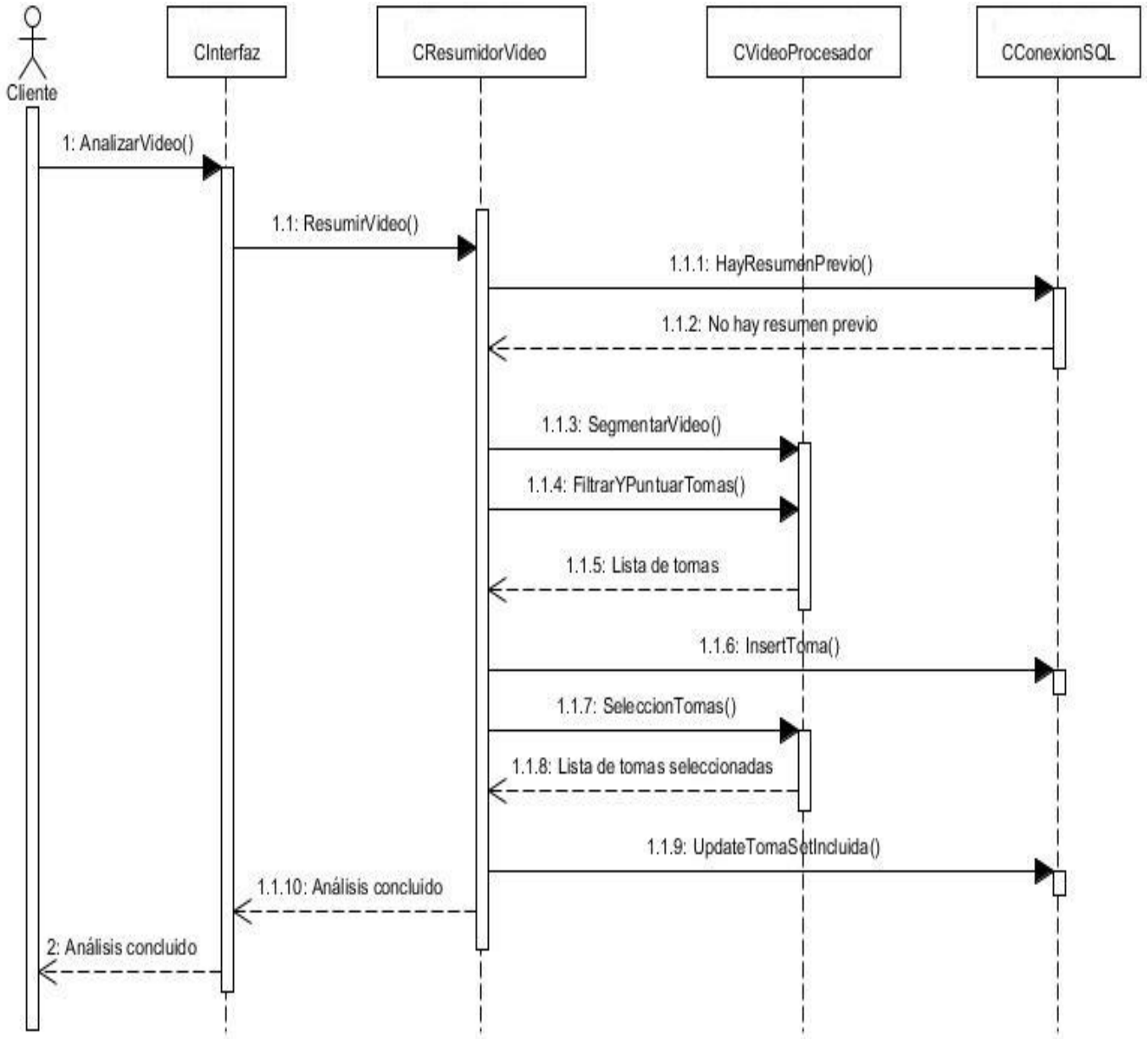

Fig. 21 Diagrama de secuencia del CU "Análisis del video"

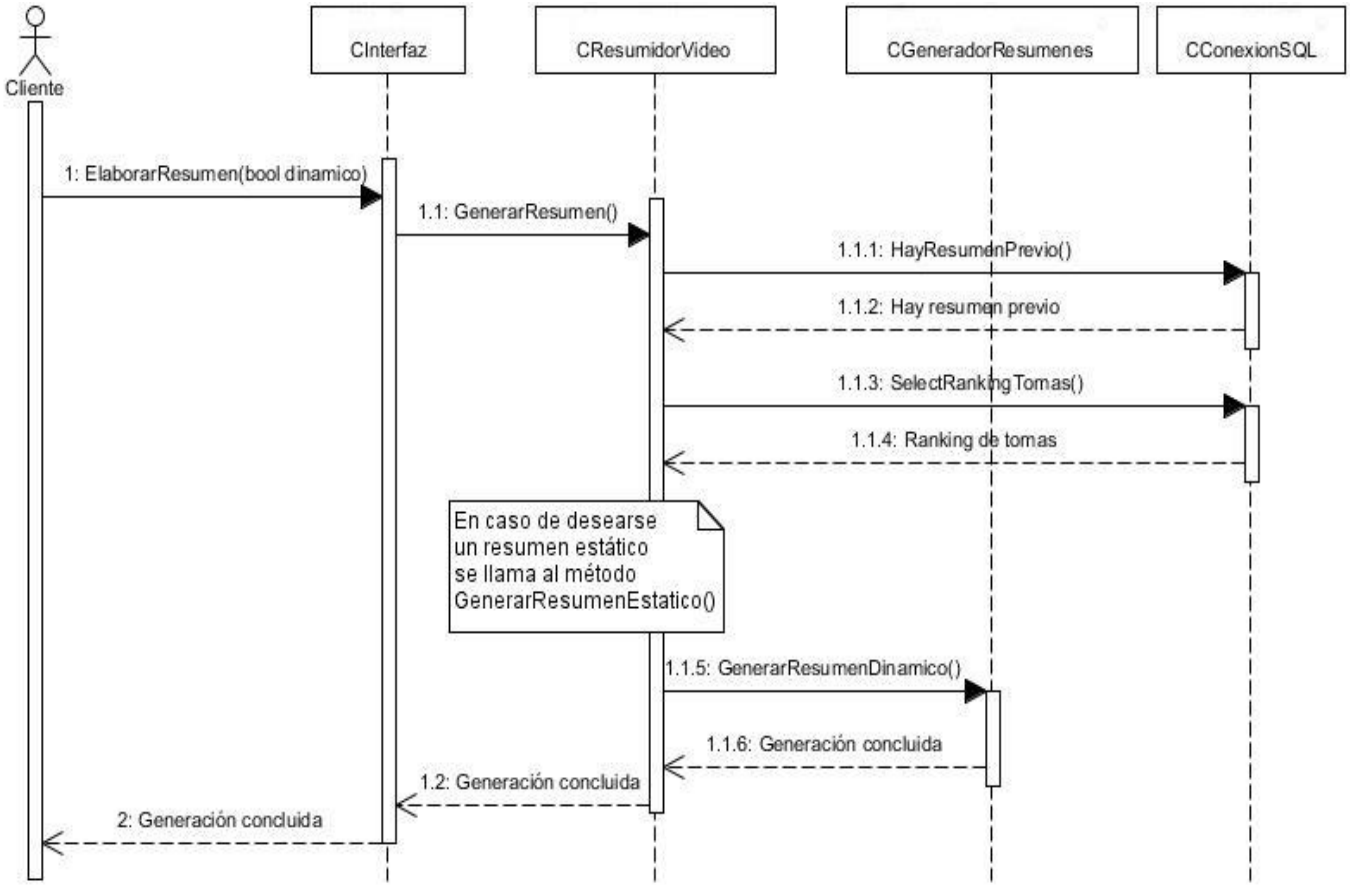

Fig. 22 Diagrama de secuencia del CU "Elaborar resumen"

Glosario de términos

- **Abstracto:** en filosofía, operación intelectual que consiste en separar mentalmente lo que es inseparable en la realidad. En programación, que genera una ilusión de simplicidad dado a que minimiza la cantidad de características que definen a un objeto.
- **API:** la interfaz de programación de aplicaciones, abreviada como API (del inglés: *Application Programming Interface*), es el conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas.
- **Artefacto:** es un producto del proceso de desarrollo de software, que puede incluir los modelos del proceso (Ej.: modelos de casos de uso, modelos de diseño, diagramas de secuencia), archivos fuente, ejecutables, documentos de diseño, reportes de prueba, prototipos, manuales de usuario y más.
- **GPL:** La Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License es la licencia más ampliamente usada en el mundo del software y garantiza a los usuarios finales la libertad de usar, estudiar, compartir y modificar el software.
- **LGPL:** Ídem GPL sin libertad de modificar el software.
- **Plugins:** es una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica. Esta aplicación adicional es ejecutada por la aplicación principal e interactúan por medio de la API.
- **SQL:** el lenguaje de consulta estructurada o SQL es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.
- **Taxonomía:** clasificación u ordenación en grupos de cosas que tienen unas características comunes.
- **XML:** proviene de eXtensible Markup Language (“Lenguaje de Marcas Extensible”). Permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.