Facultad de Ciencias y Tecnologías Computacionales

Aplicación web para la publicación de Contenidos Multimedia

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

Autor: Yobal Luis Pérez Herrera

Tutor: Ángel Ernesto Hernández Bandera

La Habana, junio de 2022

Año 64 de la Revolución de Cuba

DECLARACIÓN DE AUTORÍA

El autor del trabajo de diploma con título ‘**’Diseño de una página Web para para la visualización de contenido Multimedia**” concede a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la investigación, con carácter exclusivo. De forma similar se declara como único autor de su contenido. Para que así conste firma la presente a los <día> días del mes de <mes> del año <año>.

Yobal Luis Pérez Herrera

_____

Firma del Autor

Angel Ernesto Hernández Bandera

_____

Firma del Tutor

DATOS DE CONTACTO

Generales del diplomante

Nombre y Apellidos: Yobal Luis Pérez Herrera

Sexo: Masculino

Grupo: 601

Correo electrónico: ylherrera@estudiantes.uci.cu

Generales del tutor

Nombre y apellidos: Ángel Ernesto Hernández Bandera

Centro de trabajo: Universidad de Ciencias Informáticas

Título de la especialidad de graduado: Ingeniero en Ciencias Informáticas.

Año de graduación: 2021

Institución donde se graduó: Universidad de las Ciencias Informáticas.

Correo electrónico: aehernandez@uci.cu

AGRADECIMIENTOS

A mi mamá Iramis y a mi papá Luis Alberto que fueron quienes me dieron la vida por apoyarme en todos los momentos de mi vida. Los amo mucho.

A mi hermana Indiana, que la quiero con la vida, eres mi guía y ejemplo. Te quiero.

Primero que todo a mi familia por estar siempre preocupada y atenta por mí en todos estos años, a mi mamá y abuela que siempre han esperado este momento. A mi hermana que siempre ha sido como una segunda madre para mí. También quiero agradecer a todos mis compañeros de aula incluso a los que se fueron quedando por el camino a lo largo de estos años ya que convirtieron en mi segunda familia y me apoyaron en todo lo que pudieron.

A mi tutor José Ramón que me ha apoyado en todo momento y ha estado al tanto de mi desarrollo.

Yobal Luis Pérez Herrera

RESUMEN

El auge de los contenidos multimedia en Internet ha provocado que la sociedad contemporánea tenga nuevas motivaciones para utilizar la red de redes. Nuevos problemas asociados a esta tendencia surgieron para mostrarse como reto ante los desarrolladores de software. El objetivo de esta investigación es desarrollar una aplicación web distribuida que facilite la publicación de los contenidos multimedia en la red Tinored, de manera que las publicaciones se realicen desde servidores locales en el país. El proceso de desarrollo de software fue guiado completamente por la metodología de desarrollo OpenUP, apoyado en tecnologías como los framework Symfony2 v6, jQueryUI v1.9.2 y Bootstrap v5 y twig como motor de plantillas. El sistema gestor de base de datos PostgreSQL v14 y como herramienta para el modelado Visual Paradigm v8.0. La aplicación desarrollada: Sistema para la publicación y distribución de contenidos multimedia, reúne un conjunto de características que la convierten en una solución apta para resolver problemas de colaboración entre usuarios, gestión, reproducción y publicación de contenidos multimedia.

PALABRAS CLAVE: contenidos multimedia, colaboración, publicación.

ABSTRACT

The rise of multimedia content on the Internet has caused contemporary society to have new motivations for using the network of networks. New problems associated with this trend arose to present themselves as a

challenge to software developers. The objective of this research is to develop a distributed web application that facilitates the publication of multimedia content on the Tinored network, so that the publications are made from local servers in the country. The software development process was completely guided by the OpenUP development methodology, supported by technologies such as Symfony2 v6, jQueryUI v1.9.2 and Bootstrap v5 frameworks and twig as template engine. PostgreSQL v14 database management system and Visual Paradigm v8.0 as a modeling tool. The developed application: System for the publication and distribution of multimedia content, brings together a set of characteristics that make it a suitable solution to solve problems of collaboration between users, management, reproduction and publication of multimedia content.

KEYWORDS: multimedia content, collaboration, publication.

TABLA DE CONTENIDOS

INTRODUCCIÓN

Hasta los primeros años de la década del 90 no se podía integrar contenidos multimedia en la pantalla, debido a que no se habían desarrollado interfaces graficas para las páginas web que permitieran publicar contenido multimedia. En menos de 20 años, las páginas web sufrieron una evolución tecnológica no experimentada por otro medio de la historia. Este avance ha generado nuevos cambios en el quehacer diario y con ello la WWW(1) se ha convertido rápidamente en uno de los servicios más utilizado de Internet, lo que ha causado la proliferación de nuevos productos y servicios informativos digitales.

Los servicios de información y documentación accesibles desde Internet, más concretamente mediante servidores web, aumentan de una forma exponencial (Jódar, 2009). La lógica evolución de la web, ha generado la sustitución de páginas y documentos estáticos por documentos generados de forma dinámica. Paralelamente se transita de un simple concepto de publicación de páginas web, a esquemas más complejos que se fundamentan en procedimientos y técnicas basados en la gestión de contenidos.

La gestión de contenidos proviene del término en inglés Content Management (CM), que se asocia como un nuevo método para el diseño y desarrollo de aplicaciones web, el cual conlleva la inclusión de elementos digitales de diferentes tipos (texto y multimedia), el desarrollo de forma cooperativa y descentralizada, el paso de un modelo estático a otro mucho más dinámico y la reutilización de los contenidos (Pastor, 2006). Este concepto se asocia también con la identificación de recursos digitales, su valoración, gestión y tratamiento eficiente. A ello se le une la necesidad de utilizar tecnologías y sistemas informáticos para el almacenamiento y la distribución de información multimedia.

Las aplicaciones web implementan mecanismos de navegación sobre la información y los contenidos. De forma conjunta integran mecanismos de colaboración para el conjunto de usuarios a los que sirve como herramienta de trabajo. También gestionan información y contenidos de manera centralizada los cuales provienen de diversas fuentes. Por tanto, las aplicaciones o sitios web son básicamente una manera de facilitar el logro de una tarea específica en un entorno web.

En Cuba desde hace algunos años la entidad estatal Joven Club creo una red nacional llamada Tinored la cual brinda múltiples servicios como son: varios servidores de diferentes videojuegos, el acceso a ecurred, páginas web para chatear entre los usuarios y muchos otros. La red se compone de distintos Nodos centrales (que no pertenecen a Joven Club) que están distribuidos en algunos municipios de la Habana los cuales se conectan mediante una conexión WIFI a la sucursal de Joven Club más cercana y a su vez los usuarios se conectan al Nodo más cercano mediante una conexión WIFI (wireless network). El Nodo del Cerro o también conocido como CCN quiere poner a disposición de los usuarios un servicio de publicación de contenido multimedia a través de la red al cual va a poder acceder cualquier usuario que tenga acceso a Tinored desde cualquier parte de Cuba.

Sobre la base de los elementos expuestos anteriormente se plantea lo siguiente:

Problema de investigación:

¿Cómo contribuir a la publicación de contenidos multimedia para en Nodo del Cerro en la red Tinored?

Objeto de estudio:

Se enfoca en la gestión de contenidos multimedia en Internet.

Objetivo general:

Para darle cumplimiento al problema planteado se propone desarrollar una aplicación web distribuida, que contribuya a la publicación de contenidos multimedia para la red Tinored.

Campo de acción:

La gestión de contenidos multimedia para la red WIFI Tinored.

Objetivos específicos:

- Caracterizar las tendencias actuales de los Sistemas para la publicación de contenidos multimedia en Internet en el ámbito internacional, nacional y en la UCI.
- Identificar los requerimientos funcionales y no funcionales de la aplicación web, Sistema para la publicación de contenidos multimedia.
- Diseñar las funcionalidades de la aplicación web, Sistema para la de contenidos multimedia.
- Implementar las funcionalidades de la aplicación web, Sistema para la publicación de contenidos multimedia.
- Validar mediante pruebas la Aplicación Web para la publicación de contenidos multimedia.

Tareas de investigación:

- Caracterización de las tendencias actuales de sistemas informáticos que gestionan contenidos multimedia en el ámbito internacional, nacional y en la UCI.
- Selección de las tecnologías, herramientas, estándares, patrones y metodologías necesarias para el desarrollo de la aplicación web, Sistema para la publicación de contenidos multimedia.
- Realización del análisis y diseño de la aplicación web, Sistema para la publicación de contenidos multimedia.

- Validación mediante las pruebas funcionales, de seguridad y rendimiento la aplicación web, Sistema para la publicación de contenidos multimedia.

Para la realización de la presente investigación se hizo necesaria la utilización de los siguientes métodos de investigación científica:

Métodos Teóricos:

Analítico-Sintético: Fue empleado para realizar el análisis y la síntesis de los conceptos, estándares y herramientas en la investigación sobre los gestores de contenido. También se utilizó durante la revisión de documentos y artículos, donde se extrajeron las ideas centrales relacionadas con el funcionamiento de las aplicaciones que permiten publicar contenidos multimedia, mediante servicios web, en aplicaciones externas.

Histórico-Lógico: Empleado con el objetivo no sólo de describir cómo se han comportado las nuevas tecnologías para la reproducción de contenidos multimedia en Internet, sino también para conocer la lógica de su desarrollo y los elementos que incurrieron en los cambios operados en cada etapa por las que atravesaron dichas tecnologías.

Estudio de campos: Usado para la revisión y estudio de diferentes aplicaciones que intentan solucionar problemas similares. Permitió evaluar las tendencias evolutivas de los sistemas estudiados para realizar la caracterización de los mismos.

Métodos empíricos:

La entrevista: Se empleó como medio para la búsqueda de información y la extracción de los requisitos funcionales del sistema.

Observación: permitió un control sistemático durante todo el proceso de la investigación, para corregir los errores que se presenten.

Los **resultados esperados** al terminar la investigación son:

- Una Aplicación web para la publicación de contenido multimedia.
- La documentación generada durante la investigación.

- Los resultados de las pruebas funcionales, de seguridad y rendimiento aplicados a la Aplicación web de publicación de contenido multimedia.

1. CAPÍTULO I: Marco teórico referencial

1.1 Contenidos multimedia

Para lograr un acercamiento a la definición exacta del término contenido multimedia es necesario construirla a partir de dos conceptos fundamentales:

Contenido: Hace referencia a cualquier recurso en formato digital que se componga de dos partes fundamentales: los datos y los metadatos (Ribelles, 2013).

Multimedia: Se utiliza para referirse a cualquier objeto o sistema que utiliza múltiples medios de expresión físicos o digitales para presentar o comunicar información (Wordreference, 2014).

Tipos de información multimedia:

- **Texto**: sin formatear, formateado, lineal e hipertexto
- **Gráficos**: utilizados para representar esquemas, planos y dibujos lineales.
- **Imágenes**: son documentos formados por píxeles. Pueden generarse por copia del entorno (escaneado, fotografía digital) y tienden a ser ficheros muy voluminosos.
- **Animación**: presentación de un número de gráficos por segundo que genera en el observador la sensación de movimiento.
- **Video**: Presentación de un número de imágenes por segundo, que crean en el observador la sensación de movimiento. Pueden ser sintetizadas o captadas.
- **Sonido**: puede ser habla, música u otros sonidos.

1.1.1 Imagen digital

Las imágenes digitales son fotos electrónicas tomadas de una escena o escaneadas de documentos, fotografías, manuscritos, textos impresos e ilustraciones. Se realiza una muestra de la imagen digital y se confecciona un mapa de ella en forma de cuadrícula de puntos o elementos de la figura (píxeles). A cada píxel se le asigna un valor tonal (negro, blanco, matices de gris o color), el cual está representado en un código binario (ceros y unos) (Cabrera, 2006).

1.1.2 Video digital

Proveniente del Inglés video, y del Latín viděo, yo veo. Un archivo de video se refiere al sistema de grabación y reproducción de imágenes, acompañadas o no de sonidos. "Un video consiste en una secuencia de imágenes que son visualizadas a una frecuencia preestablecida (play rate), que normalmente suele ser alrededor de 30 imágenes por segundo" (Cores, 2003). Actualmente, el término hace referencia a distintos formatos. Además de las cintas de video analógico, como VHS y Betamax, también se incluyen los formatos digitales DVD, MPEG y OGV, Webm, entre otros.

Características de algunos formatos de video

MPEG-1

Conjunto de estándares y formatos de compresión de video y audio diseñados por MPEG (Moving Picture Experts Group). MPEG-1 se utiliza frecuentemente en el video CD (VCD) (Cores, 2003).

MPEG-2

Versión de MPEG, de mayor calidad. Se definen dos tipos de formato de contenedor: flujo de transporte y flujo de programa. El primero está enfocado a aplicaciones que transmiten el video y audio por canales poco confiables, es el indicado para el uso en aplicaciones de transmisión, el segundo está considerado para uso en discos. El códec de video es semejante a la primera versión de este formato, pero se introduce la posibilidad de codificar con interlineado. En el códec de audio, se introduce la posibilidad de codificar más de 2 canales de audio, mediante AAC codificación de 5.1 canales. Este formato es utilizado tanto en SVCD, DVD, y en la transmisión de videos digitales (Cores, 2003).

MPEG-4

Igual a sus predecesores, más varios complementos adicionales; la gestión de derechos digitales (DRM, por sus siglas en inglés), soporte de lenguaje para modelado de realidad virtual (VRML, por sus siglas en inglés) para renderizar escenas en tercera dimensión. Este formato contenedor soporta un conjunto mucho más amplio

de códec de audio y video que sus predecesores (Montañola, 2007).

Theora

Códec de video basado en VP3, desarrollado por la fundación Xiph.org como parte del proyecto Ogg. La característica principal que lo diferencia de sus competidores es la posibilidad de poderlo usar de forma libre y gratuita, sin tener que pagar ningún tipo de cuota de uso a nadie. El códec está basado en el códec VP3 que fue cedido a la fundación. Este códec, puede ser utilizado dentro de cualquier contenedor, por ejemplo, el contenedor de MPEG-4. Theora se suele usar en conjunción con el códec de audio Vorbis dentro de un contendedor Ogg. Su empaquetado está orientado a streaming lo que constituye la mayor diferencia en diseño sobre otros formatos contenedores basados en archivo (Theora.org, 2014).

Webm

Webm es un formato multimedia abierto y libre desarrollado por Google y orientado a usarse con HTML5. Presenta una licencia permisiva similar a la licencia BSD. Está compuesto por el códec de video VP8 (desarrollado originalmente por On2 Technologies) y el códec de audio Vorbis dentro de un contenedor multimedia Matroska. Cuenta con el apoyo y contribución oficial de empresas como Mozilla, Opera, Google, Adobe y otros fabricantes de software y hardware en un esfuerzo combinado para lograr que este formato multimedia sea estándar para el lenguaje web HTML5 (developer.mozilla.org, 2013).

1.1.3 Audio digital

Proveniente del Inglés audio, y del Latín audĭo, yo oigo. Un archivo de audio se refiere a la técnica relacionada con la reproducción, grabación y transmisión del sonido.

En la rama de la electrónica el audio digital se considera como la codificación digital de una señal eléctrica que representa una onda sonora. Consiste en una secuencia de valores enteros y se obtiene de dos procesos: el muestreo y la cuantificación digital de la señal eléctrica. El tamaño del archivo no varía, así contenga silencio o sonidos muy complejos. Existen diferentes formatos de archivo de audio digital, los cuales pueden clasificarse en dos categorías PCM6 (incluye formatos como WAV y AIFF) y comprimidos (incluye formatos como MP3 y Ogg) (cenart, 2013).

WAV

Es un formato de audio digital normalmente sin compresión de datos desarrollado y propiedad de Microsoft y de IBM que se utiliza para almacenar sonidos en la computadora, admite archivos mono y estéreo a diversas resoluciones y velocidades de muestreo, su extensión es WAV. En Internet no es un formato popular, fundamentalmente porque los archivos sin compresión son muy grandes (Instituto Superior de Formación y Recursos en Red para el Profesorado, 2008).

AIFF

Fue desarrollado por Apple Inc. en 1988 basado en el IFF (Interchange File Format) de Electronic Arts.

Actualmente es muy utilizado en las computadoras Apple Macintosh y por Silicon Graphics Incorporated. Presenta las características que se describen a continuación (Apple Computer Inc, 1985):

- Los datos de audio en el estándar AIFF no están comprimidos.
- Los datos son almacenados en big-endian.
- Emplea modulación por impulsos codificados (PCM).

Formatos comprimidos o con pérdidas:

MP3

MP3 o MPEG-1 Audio Layer 3, es un formato de audio digital comprimido con pérdida desarrollado por el Moving Picture Experts Group (MPEGH). El mp3 estándar es de 144 kHz y un bit rate de 317 kbps por la relación de calidad-tamaño. Este formato se convirtió en el estándar utilizado para streaming de audio y compresión de audio de alta calidad (con pérdida en equipos de alta fidelidad) gracias a la posibilidad de ajustar la calidad de la compresión, proporcional al tamaño por segundo, por tanto, el tamaño final del archivo, que podía llegar a ocupar 12 e incluso 15 veces menos que el archivo original sin comprimir (Giacoia, 2009).

Vorbis

Formato contenedor de multimedia, desarrollado por la Fundación Xiph.org, es el formato nativo para los códecs multimedia que también desarrolla Xiph.org. Dicho formato se utiliza preferentemente con los códecs de audio Vorbis y de video Theora desarrollados por dicha fundación y Webm desarrollado por Google. El formato es libre de patentes y de código abierto al igual que toda la tecnología de Xiph.org, diseñado para dar un alto grado de eficiencia en el streaming y la compresión de archivos (Xiph.org, 2013).

1.2 Las Tecnologías de publicación de contenidos multimedia (archivos de audio y video)

Se pueden definir tres tecnologías de publicación audiovisual en Internet, que se presentan en orden cronológico (coincidente con una mayor complejidad):

Video incrustado

Este modelo de plataforma de publicación básica fue la primera aproximación a la integración de audio y video en la web. Requería de un servidor web, por lo que hacía uso del protocolo HTTP, sencillo y con mecanismo de confirmación por el cliente de los datos enviados, pensado para la transmisión de pequeños ficheros a un conjunto simultáneo de clientes. El fichero de video se incrustaba o integraba en una página web en HTML, como cualquier otro fichero, mediante las instrucciones HREF, EMBED, BGSOUND, entre otros. En los tiempos de su surgimiento, los formatos contenedores de video no estaban preparados para poder ser reproducidos de

forma progresiva, por lo que se generaba una espera directamente proporcional a la duración del material. Además de este retardo, el equipo cliente debía poseer suficientes recursos para poder cargar el fichero en memoria y así abrirlo (Ribelles, 2013).

Descarga progresiva

Las limitaciones de la tecnología anteriormente descrita y la creciente demanda de contenidos multimedia en la web llevaron al desarrollo de un mecanismo de distribución de descarga progresiva, mediante el cual la reproducción es posible al recibir una parte suficiente de datos. Sin embargo, no todos los formatos son compatibles con este sistema de descarga; el fichero de video debe tener un formato contenedor que sea capaz de soportar este tipo de descarga. Su estructura debe poseer, al menos, una cabecera que prepare al programa reproductor indicándole los datos básicos del tipo de codificación y duración para prever las necesidades de memoria que requerirá. Además, los datos de video y audio tienen que estar debidamente
combinados, pues si el audio se encuentra después de todos los datos de video (o viceversa) no será posible reproducir nada en condiciones hasta no recibir todo el fichero. Esta nueva forma de reproducción, mejora considerablemente a la anterior debido a que, en condiciones normales, un servidor web envía numerosos ficheros pequeños a un número limitado de usuarios simultáneos, lo que divide el ancho de banda entre los envíos, aunque, al ser de pequeño tamaño, debería ser suficiente y posiblemente no se retarde el envío de datos desde el servidor. No obstante, en el momento que se envían ficheros de audio y video a un gran número de usuarios el ancho de banda se reduce notablemente y la capacidad de respuesta del servidor se verá disminuida. Por otro lado, el tradicional protocolo web HTTP opera utilizando protocolo de transporte TCP/IP, el cual asegura la recepción de los datos (tras los envíos, debe recibir las confirmaciones de recepción del cliente; si no las recibe, reenvía los datos). Esta seguridad en la transmisión no es eficiente para audio y video, no por el volumen de datos, sino porque la pérdida de alguno no afecta sensiblemente la recepción del cliente, y porque si reenvía datos que se han perdido previamente no puede asegurarse que lleguen a tiempo para ser reproducidos (Ribelles, 2013).

Streaming de video

Las tecnologías antes mencionadas no superaban todas las expectativas por lo que surge un tercer mecanismo de reproducción, el streaming, mediante la adición de un servidor adicional al servidor web (o servicio adicional en el mismo servidor web) cuyas características superan las limitaciones antes mencionadas. La tecnología streaming se utiliza para aligerar la descarga y ejecución de audio y video en la web, ya que, permite escuchar y visualizar los archivos mientras se están descargando. Si no se utiliza la tecnología streaming, para mostrar un contenido multimedia en la red, se descarga primero el archivo entero en el ordenador y más tarde se ejecuta, para finalmente ver y escuchar lo que el archivo contiene. Sin embargo, el streaming permite que esta tarea se realice de una manera más rápida y sea posible ver y escuchar su contenido durante la descarga (Álvarez, 2003).

Para un mejor entendimiento de cómo funciona esta novedosa tecnología se tuvieron en cuenta los conceptos fundamentales que la componen y la función que realiza cada uno de ellos. A continuación, se describe cada uno de ellos:

- **Conexión con el servidor**. El reproductor cliente se conecta al servidor y este comienza a enviarle el archivo solicitado.
- **Buffer**. El cliente recibe el archivo solicitado y comienza a construir un buffer o almacén donde guarda el archivo que está siendo enviado por el servidor.
- **Inicio de la reproducción**. Cuando el buffer está lleno con una pequeña fracción del archivo enviado por el servidor, el reproductor del cliente comienza a mostrarlo mientras continúa en segundo plano con el resto de la descarga.
- **Caídas de la velocidad de conexión**. Si la conexión experimenta descensos en la velocidad durante la reproducción, el reproductor del cliente podría seguir mostrando el contenido, consumiendo la información almacenada en el buffer. Si llega a consumir todo el buffer, el reproductor se detendría hasta que el buffer se volviese a llenar.

¿Por qué utilizar la tecnología streaming?

Luego de haber analizado algunas de las etapas y tecnologías por las que ha pasado la reproducción de contenido audiovisual en Internet el equipo de trabajo arribó a la conclusión de que la tecnología streaming es la más apropiada para la reproducción de contenidos multimedia en el Sistema para la publicación y distribución de contenidos multimedia debido a que:

- Brinda mayor rapidez en la visualización de contenidos multimedia.
- La comunicación cliente-servidor se puede realizar por protocolos alternativos al HTTP.
- A pesar de tener la desventaja del bloqueo impuesto por Firewalls, posee la ventaja de una mayor rapidez.
- Permite mejor gestión del procesador y ancho de banda de la máquina del servidor ante peticiones simultáneas de varios clientes de los mismos contenidos multimedia debido a que conoce la velocidad de codificación necesaria para su correcta reproducción, reservando un ancho de banda idéntico para su streaming.
- Garantiza la reproducción ininterrumpida gracias al establecimiento de una conexión de control inteligente entre servidor y cliente.
- Posibilita la distribución de transmisiones de audio y video en directo.

En principio no es necesario contar con un servidor especial para colocar archivos de audio o video con descarga streaming en la web. Cualquier servidor web puede mandar la información y es el cliente el que se encarga de procesarla para poder mostrarla a medida que la recibe. Sin embargo, existen servidores especiales preparados para transmitir streaming. Aunque en muchas ocasiones no es necesario utilizarlos, pueden ofrecer

importantes prestaciones como mandar un archivo de mayor o menor calidad dependiendo de la velocidad de la red de transmisión de datos. A continuación, se realiza una breve descripción de algunos de estos servidores (Álvarez, 2003).

Servidores streaming

El streaming se ha impuesto a la tradicional descarga, gracias al avance en la oferta de ancho de banda, que permite al consumidor acceder a los contenidos multimedia sin necesidad de proceder a la descarga en su unidad de almacenamiento. Esta modalidad está estrechamente relacionada a ciertos contenidos multimedia (audio y especialmente video) sobre todo, por la ventaja que supone frente a la descarga, al no necesitar almacenamiento.

Como consecuencia de que no es posible acceder a un software de pago se decide que los servidores que serán analizados deben ser libres o de código abierto. También se tuvo en cuenta que dicho software fuera capaz de soportar diferentes tipos de medias (audio y video).

Luego de realizadas estas acotaciones, se propone de manera resumida una descripción de los posibles servidores streaming a utilizar (Ver Tabla 1. Descripción de los servidores streaming) teniendo en cuenta algunos indicadores definidos por el equipo de trabajo según las necesidades imperantes.

Nombre	Patrocinador	Ultima versión estable	Sistema operativo	Licencia	Tipo de media	Tipos de formatos que soporta
VLC media player	VideoLAN	3.0.17.4	Linux/Windows/MacOS	GPL v2	Audio/Video	AVI, MP4, Ogg, Web, entre otros
Icecast	Xiph.Org Foundation	2.4.6	Linux	GPL	Audio/Video	Ogg, Theora, ACC

Tabla 1. Descripción de los servidores strreaming.

1.3 Plataformas de publicación y distribución de contenidos multimedia

Para conocer las tendencias en Internet sobre el manejo de contenidos multimedia se analizarán diferentes aplicaciones web tanto en el ámbito internacional, nacional como en el marco de la UCI. Para realizar el estudio de estas plataformas, el equipo de desarrollo basó la recopilación de información sobre diferentes métricas, las cuales se explican a continuación:

- Interoperabilidad entre sistemas: hace referencia a la habilidad de dos o más sistemas para intercambiar información y utilizar la información intercambiada.
- Compartir contenidos: es la posibilidad que tienen los sistemas de socializar los contenidos con otros usuarios del sistema.

- Votar contenidos: funcionalidad que permite realizar votaciones por los contenidos.
- Uso de la tecnología Streaming: hace referencia a la tecnología de reproducción de contenido de audio y video basado en el flujo continuo de estos recursos.
- Recuperación simple de contenidos: posibilidad que tienen los sistemas para realizar búsquedas de contenidos de menor complejidad.
- Recuperación avanzada de contenidos: posibilidad que tienen los sistemas para realizar búsquedas especificando propiedades conocidas de los contenidos.
- Reproducción bajo demanda de contenidos: hace referencia a la posibilidad de reproducir cualquier archivo de audio o video en cualquier instante de tiempo deseado por el usuario.
- Gestión de contenidos: posibilidad que tienen los sistemas de adicionar, modificar, eliminar, publicar contenidos.
- Contenidos sugeridos: funcionalidad que permite sugerir contenidos a varios usuarios del sistema.
- Adicionar comentarios al contenido: funcionalidad que permite realizar comentarios a los contenidos.
- Contenidos recientes: muestra los contenidos subidos recientemente al sistema.
- Contenidos populares: muestra los contenidos que más se han visitado o votado por los usuarios.
- Tipos de contenidos multimedia: hace referencia a los contenidos con los cuales funcionan las plataformas analizadas: imagen, audio y video.
- Creación de cuentas de usuario: posibilidad que tienen los usuarios de crear cuentas de usuario para una mejor interacción con el sistema.
- Acceso libre a todos los contenidos: posibilidad que permite a los usuarios acceder de forma libre y gratis a todos los contenidos disponibles en el sistema.
- API´s de servicio: conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizada por otro software como una capa de abstracción.

Funciones / Sistemas	YouTube	Flickr	Instagram	Vimeo	TeVeo	Internos
Interoperabilidad con otros sistemas	X	X	X	X	X	
Compartir contenidos	X	X	X	X		
Votar contenidos	X	X	X	X		X
Uso de la tecnología *Streaming*		X	X	X	X	X
Recuperación simple de contenidos	X	X	X	X	X	X
Recuperación avanzada de contenidos	X	X	X	X		
Reproducción bajo demanda de contenidos	X	X	X	X	X	X
Gestión de contenidos	X	X	X	X	X	
Contenido sugeridos	X	X	X	X		
Adicionar comentario a los contenidos	X	X	X	X		X
Contenidos recientes	X	X	X	X	X	X
Contenidos populares	X	X	X	X	X	X
Tipos de contenidos multimedia	X	X	X	X		
Creación de cuentas de usuario	X	X	X	X		
Acceso libre a todos los contenidos					X	X
API's de servicio	X	X	X	X		

Tabla 2. Análisis de las plataformas para la publicación y distribución de contenidos multimedia

1.4 Tecnologías, herramientas y metodologías

Para la creación de aplicaciones informáticas es necesario usar tecnologías y herramientas que faciliten definir cuál va a ser su estructura, además de metodologías que permitan regir la forma de trabajo del equipo de desarrollo. A continuación, se exponen la metodología, tecnologías y herramientas a emplear basadas en el marco de trabajo para el desarrollo de la propuesta de solución.

1.4.1 Metodología de desarrollo de software

Es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software. A continuación, se realiza una descripción de la metodología de desarrollo seleccionada por el equipo de trabajo para guiar el proceso de implementación de la propuesta de solución.

Open Unified Process (OpenUP o Proceso Unificado Abierto)

Es un proceso modelo y extensible, dirigido a la gestión y desarrollo de proyectos de software basados en

desarrollo iterativo, es ágil e incremental, apropiado para pequeños proyectos; y es aplicable a un amplio conjunto de plataformas de desarrollo (Balduino, 2007).

Principios básicos que caracterizan a la metodología OpenUP:

- Colaborar para concertar intereses y conocimiento compartidos.
- Equilibrar las prioridades para extender los beneficios obtenidos por los interesados en el proyecto.
- Enfocarse en la arquitectura para minimizar los riesgos y organizar el desarrollo.
- Desarrollo progresivo para obtener retroalimentación temprana y mejoramiento continuo.

Ciclo de vida de OpenUp (Balduino, 2007):

- **Concepción**: Primera de las cuatro fases en el ciclo de vida del proyecto, acerca del entendimiento del propósito y los objetivos, permite obtener suficiente información para confirmar qué debe hacer el proyecto. El objetivo de esta fase es capturar las necesidades de los stakeholder en los objetivos del ciclo de vida del proyecto.
- **Elaboración**: Es la segunda de las cuatro fases del ciclo de vida de OpenUP donde se tratan los riesgos significativos para la arquitectura. El propósito de esta fase es establecer la base de la elaboración de la arquitectura del sistema.
- **Construcción**: Esta fase está enfocada al diseño, implementación y prueba de las funcionalidades para desarrollar un sistema completo. El objetivo de esta fase es completar el desarrollo del sistema basado en la arquitectura definida.
- **Transición**: Es la última fase, cuyo propósito es asegurar que el sistema es entregado a los usuarios, y evalúa la funcionalidad y rendimiento del último entregable de la fase de construcción.

Se propone el uso de la metodología OpenUP debido a que:

- Es apropiado para proyectos pequeños y de bajos recursos, permite disminuir las probabilidades de fracaso e incrementar las de éxito.
- Es un proceso iterativo mínimo, completo y extensible.
- Motiva un enfoque centrado en el cliente y un ambiente colaborativo entre los miembros del equipo de desarrollo.
- Permite detectar errores tempranos a través de un ciclo iterativo.
- Puede ser utilizado como base para agregar o adaptar más procesos.

1.4.2 Ingeniería de Software asistida por computadoras (CASE)

Nombre que se le da al software que se utiliza para ayudar a las actividades del proceso de desarrollo de software como la ingeniería de requerimientos, el diseño, el desarrollo de programas y las pruebas. Por tanto, las herramientas CASE incluyen editores de diseño, compiladores, depuradores, herramientas de construcción de sistemas, entre otros (Sommerville, 2005).

Entre las facilidades que proporciona la tecnología CASE se pueden encontrar:

- El desarrollo de modelos gráficos del sistema como parte de la especificación de requerimientos o del diseño de software.
- La comprensión del diseño utilizando un diccionario de datos que tiene información sobre las entidades y relaciones del diseño.
- La generación de interfaces de usuario a partir de la descripción gráfica de la interfaz que es elaborada de forma interactiva por el usuario.

Luego de la puntualización de algunas de las facilidades que brinda la tecnología CASE el equipo de trabajo decidió hacer uso de la herramienta Visual Paradigm la cual se describe a continuación.

Visual Paradigm v8.0

Visual Paradigm es una herramienta CASE (Ingeniería de Software Asistida por Computación). La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, a través del análisis y diseño, hasta la generación del código fuente de los programas y la documentación. Es capaz de soportar el ciclo de vida completo del proceso de desarrollo de software a través de la representación de todo tipo de diagramas (Visual Paradigm International, 2013).

Visual Paradigm presenta las siguientes características:

- Disponibilidad en múltiples plataformas (Windows, Linux, Mac).
- Está diseñado para centrarse en casos de uso y enfocado al negocio que genera un software de alta calidad.
- Uso de un lenguaje estándar y común, lo que facilita la comunicación entre los integrantes del equipo de desarrollo.

- Tiene la capacidad de realizar ingeniería directa e inversa.
- El modelo y el código permanecen sincronizados durante todo el ciclo de desarrollo de software.
- Se encuentra disponible con licencia gratuita y comercial.
- Está diseñado para soportar aplicaciones web y de escritorio.

Para desarrollar los artefactos generados por esta herramienta CASE, se hará uso de herramientas y lenguajes que serán de vital importancia para el desarrollo de la solución, las cuales se describen a continuación.

1.4.3 Entorno de Desarrollo Integrado (IDE)

Un entorno de desarrollo integrado, llamado también (IDE, por sus siglas en inglés), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios.

Visual Studio: Es un entorno de desarrollo dedicado a diversos lenguajes entre los que se encuentran VisualBasic.Net, C#, C++, F#, Java, Python, entre otros. Este permite a los desarrolladores crear aplicaciones sobre la plataforma .NET además de incluir un diverso conjunto de herramientas para asistir al desarrollo de software. Para el presente trabajo se utilizará la versión la versión 17.2.

1.4.4 Sistema Gestor de Bases de Datos (SGBD)

Un Sistema de Gestión de Base de Datos (SGBD) consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a ellos. El objetivo fundamental de un SGBD es proporcionar un entorno que sea conveniente y eficiente para ser utilizado al extraer y almacenar información de la base de datos.

PostgreSQL v14

PostgreSQL es un sistema de gestión de base de datos relacional, distribuido bajo licencia BSD y de código abierto. PostgreSQL utiliza un modelo cliente-servidor y multiprocesos en lugar de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará al resto y el sistema continuará funcionando. La versión propuesta ofrece varias características importantes, entre ellas la eliminación de obstáculos para el despliegue de nuevas aplicaciones. Incluye replicación sincrónica que permite alta disponibilidad con consistencia a través de varios servidores, admite ordenación lingüística por base de datos,

tabla o columna y mejora en gran medida el rendimiento de datos efímeros (Martínez, 2010).

Algunas de las características por las que se eligió PostgreSQL como SGBD se mencionan a continuación:

- Es una base de datos 100% ACID.
- Integridad referencial.
- Copias de seguridad en caliente.
- Juegos de caracteres internacionales.
- Múltiples métodos de autentificación.
- Acceso encriptado vía SSL.
- Abundante documentación disponible para Linux y UNIX en todas sus variantes y Windows 32/64bit.

1.4.5 Lenguaje de modelado UML

UML es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Se usa para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre estos sistemas.

1.4.6 Lenguajes de desarrollo para la web y marcos de trabajo

Son lenguajes reconocidos directamente por el navegador. Actualmente existe gran diversidad de ellos, que han surgido como consecuencia de las tendencias y necesidades de las distintas plataformas de desarrollo. El uso de estos lenguajes garantiza un diseño óptimo, eficiente, seguro y dinámico en las aplicaciones web. El equipo de desarrollo propone el uso de PHP, HTML 5, CCS 3 y JavaScript como lenguajes para el desarrollo web.

Lenguajes de desarrollo

PHP v8.0.20

PHP es un lenguaje interpretado de propósito general ampliamente usado, diseñado especialmente para desarrollo web y que puede ser incrustado dentro de código HTML. Generalmente se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno.

HTML 5

HTML es el lenguaje con el que se escriben las páginas web. Es un lenguaje de hipertexto, es decir, un lenguaje que permite escribir texto de forma estructurada, y que está compuesto por etiquetas, que marcan el inicio y el fin de cada elemento del documento. Un documento hipertexto no sólo se compone de texto, puede contener imagen, sonido, video, entre otros., por lo que el resultado puede considerarse como un documento multimedia. HTML5 no es una nueva versión del lenguaje de marcación HTML, sino una agrupación de diversas especificaciones concernientes al desarrollo web (Álvarez, 2009). HTML5 no se limita sólo a crear nuevas

etiquetas, atributos y eliminar aquellas marcas que están en desuso o se utilizan inadecuadamente, sino que es una nueva versión de diversas especificaciones, entre las que se encuentran: HTML 4, XHTML 1, CSS Nivel 2 y DOM Nivel 2.

CSS 3

CSS es un lenguaje de hojas de estilo en cascada creado para controlar la presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para la creación de páginas web complejas. CSS3, de cara a los desarrolladores de páginas web, consiste en la incorporación de nuevos mecanismos para mantener un mayor control sobre el estilo con el que se muestran los elementos de las páginas, sin tener que recurrir a trucos, que a menudo complicaban el código de la web (Álvarez, 2011).

JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como aparición y desaparición de texto, animaciones, acciones que se activan al pulsar botones u otros elementos y ventanas con mensajes de aviso al usuario. Además, es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios (Eguíluz, 2007).

Twig

Twig es un motor de plantillas para el lenguaje de programación PHP. Es un producto de código abierto y se distribuye bajo licencia BSD. Symfony2 soporta de manera nativa el motor de plantillas Twig, por tanto, lo incluye por defecto en sus proyectos. Esto no quiere decir que el equipo de trabajo tenga que usarlo obligatoriamente, pero se decidió su utilización debido a que es (Eguí luz, 2013):

- Rápido: Compila las plantillas a código PHP de modo sencillo y optimizado. La sobrecarga en comparación con el ordinario de código PHP se ha reducido a la mínima expresión.
- Seguro: Tiene un modo de recinto para evaluar código de la plantilla que no es de confianza. Esto le permite a Twig ser utilizado como un lenguaje de plantillas para aplicaciones donde los usuarios pueden modificar el diseño de la plantilla.
- Flexible: Es impulsado por un léxico flexible y analizador. Esto permite al desarrollador definir sus propias etiquetas y filtros personalizados, y crear su propio Domain-Specific Language (DSL, por sus siglas en inglés).

Marcos de Trabajo

En el desarrollo de software, un marco de trabajo es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un marco de trabajo puede incluir soporte de programas, librerías y un lenguaje de scripting entre otros softwares para ayudar a desarrollar y unir los

diferentes componentes de un proyecto. Un marco de trabajo agrega funcionalidad extendida a un lenguaje de programación, este automatiza muchos de los patrones de programación para orientarlos a un determinado propósito. Un marco de trabajo proporciona estructura al código y hace que los desarrolladores escriban código más entendible y mantenible. Además, hace la programación más fácil, convirtiendo complejas funciones en sencillas instrucciones. Como marcos de trabajo seleccionados por el equipo de desarrollo se encuentran Symfony2, jQueryUI y Bootstrap.

Symfony2 v6

Symfony2, marco de trabajo implementado completamente con PHP 5, diseñado para optimizar gracias a sus características, el desarrollo de las aplicaciones web. Symfony2, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Suministra varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, que permiten al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea
una nueva aplicación web, haciendo posible la reutilización de código. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony2 es compatible con la mayoría de los gestores de bases de datos relacionales y no relacionales, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft y MongoDB. Se puede ejecutar tanto en plataformas Unix (Linux, entre otros) como en plataformas Windows (Eguí luz, 2013).

jQueryUI versión v1.9

JQuery, es una biblioteca o marco de trabajo de JavaScript, es software libre y de código abierto. Posee un doble licenciamiento bajo las licencias MIT y GPL de GNU v2. Permite ser usada en proyectos libres y privativos. JQuery consiste en un único fichero JavaScript que contiene las funcionalidades comunes de DOM, eventos, efectos y AJAX.

Permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones.

jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo (Álvarez, 2009).

Bootstrap v5

Bootstrap es un marco de trabajo desarrollado para facilitar el proceso de diseño de páginas web. Para ello ofrece una serie de plantillas CSS y de ficheros JavaScript que permiten conseguir interfaces web que funcionen en los navegadores actuales; un diseño que pueda ser visualizado de forma correcta en distintos dispositivos y a distintas escalas y resoluciones; una mejor integración con las librerías usadas habitualmente, como por ejemplo jQuery; un diseño sólido basado en herramientas actuales y potentes, o estándares como

CSS3/HTML5 (Otto, 2012).

Conclusiones del capítulo

El análisis evolutivo sobre la reproducción de contenidos multimedia en Internet ofreció un mejor acercamiento a las tecnologías dedicadas a esta función, por lo que se logró determinar que la tecnología streaming es la adecuada a utilizar en la solución propuesta. La comparación entre los diferentes formatos de audio y video dio como resultado los formatos contenedores que serán soportados por el sistema. El análisis de los servidores streaming arrojó como resultado que VLC media player será el utilizado para la reproducción de contenidos multimedia en la red. En la selección de las herramientas y tecnologías para el desarrollo de la solución fue tomada en cuenta la política de uso de herramientas con soporte multiplataforma y basadas en software libre.

CAPÍTULO II: CARACTERÍSTICAS Y DISEÑO DEL SISTEMA

2.1 PROPUESTA DE SOLUCIÓN

La solución propuesta es una aplicación web distribuida que constituye un Sistema para la publicación y distribución de contenidos multimedia. Este sistema debe permitir gestionar, compartir, votar e incluso sugerir contenidos multimedia. Contribuir con la publicación de contenidos multimedia en la red nacional y que esta se realice con la rapidez requerida es el objetivo principal de la presente investigación.
La arquitectura de información del Sistema para la publicación y distribución de contenidos multimedia debe ser amigable y fácil de entender para usuarios que tengan conocimientos informáticos básicos. Se implementará una capa de servicios web la cual hará posible que el proceso de publicación de contenidos multimedia en aplicaciones web se realice de forma transparente al usuario encargado de publicar.

2.2 Diagrama de clases del modelo de dominio

El análisis del dominio del software es la identificación, el análisis y la especificación de los requisitos comunes entre aplicaciones que pertenezcan al mismo dominio (Pressman, 2002). De esta manera se logra reutilizar el mismo modelo en diversos proyectos que pertenezcan al mismo dominio de aplicación. A continuación, se muestra de manera general cómo funcionan las aplicaciones web.

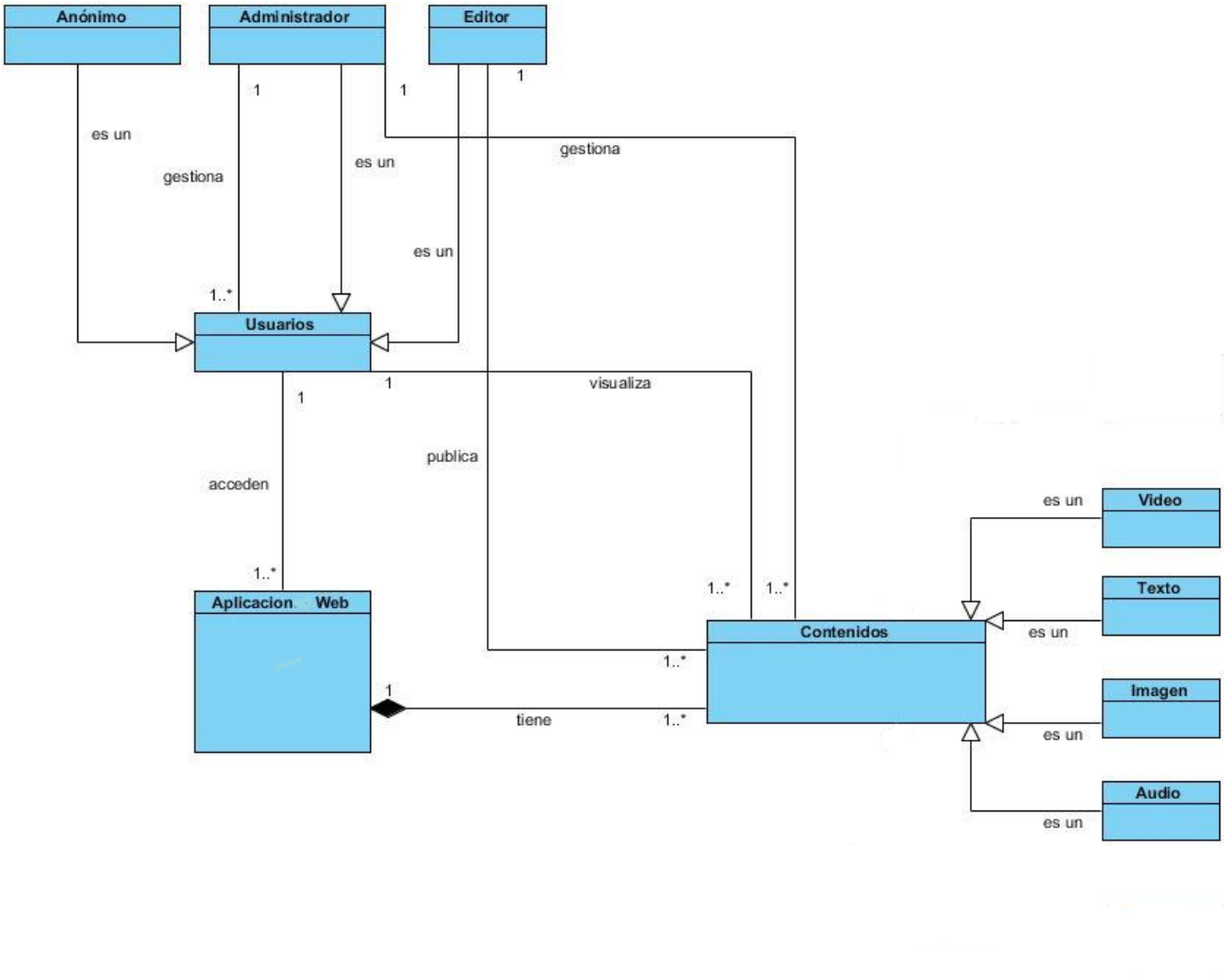


Figura 1. *Modelo de Dominio*

Descripción de las clases del modelo de dominio

Aplicación Web: es el sitio web que muestra los diferentes tipos de contenidos.

Usuarios: son las diferentes personas que interactúan con la aplicación web.

Anónimo: es el tipo de usuario que navega por la aplicación web sin autorización para realizar ningún cambio o gestión sobre contenidos o usuarios.

Editor: es el usuario que además de navegar por las aplicación web tiene permisos para realizar cambios en ellas, por ejemplo publicar o gestionar contenidos.

Administrador: usuario que navega por las aplicación web y tiene todos los permisos para realizar cambios en ellas como gestionar usuarios.

Contenidos: hace referencia a los tipos de contenidos que pueden estar presentes en las aplicación web como

por ejemplo: texto, video, gráficos, imágenes, audio y animaciones.

Texto: tipo de contenido que muestra la información que se encuentra en el cuerpo de las aplicación web.

Video: hace referencia a los contenidos audio-visuales que tienen las aplicación web.

Imagen: hace referencia a las imágenes contenidas en las aplicación web.

Audio: hace referencia a los contenidos de sonido que tienen las aplicación web.

2.3 Especificación de requisitos de software

El flujo de trabajo de los requisitos surge con el objetivo de guiar el desarrollo hacia el sistema correcto. La especificación de requisitos de software es, por tanto, una descripción completa del comportamiento del sistema que se va a desarrollar (Pressman, 2002).

El objetivo principal en esta etapa es identificar lo que realmente se necesita y de este modo transmitirlo al cliente y al equipo de desarrollo de la manera más fácil y entendible posible. Para el desarrollo del Sistema para la publicación y distribución de contenidos multimedia se realizaron entrevistas a los clientes, en las cuales se obtuvieron las características funcionales y no funcionales con las que debe cumplir el sistema.

Requisitos funcionales

Un requisito funcional define las funciones de un sistema de software o sus componentes. Los requerimientos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que un sistema debe cumplir (Pressman, 2002). Los requerimientos de comportamiento para cada requisito funcional se muestran en cómo los casos de uso pueden ser llevados a la práctica. Los requisitos funcionales del Sistema para la publicación y distribución de contenidos multimedia son descritos a continuación.

RF_1. Autenticar usuario
RF_2. Gestionar usuario
RF_3. Recuperar contraseña
RF_4. Visualizar perfil del usuario
RF_5. Visualizar usuarios seguidores y seguidos
RF_6. Gestionar preferencias
RF_7. Gestionar recursos compartidos
RF_8. Gestionar contenidos multimedia
RF_9. Descargar contenidos multimedia
RF_10. Dar Me gusta a los contenidos multimedia
RF_11. Visualizar cantidad de seguidores
RF_12. Compartir contenido multimedia
RF_13. Realizar búsqueda simple

RF_14. Realizar búsqueda avanzada
RF_15. Visualizar términos de servicio
RF_16. Denunciar contenido
RF_17. Gestionar etiquetas
RF_18. Gestionar sugerencia
RF_19. Configuración avanzada del sistema
RF_20. Publicar contenido
RF_21. Capa de servicios web
RF_22. Visualizar gráfica

Requisitos no funcionales

Los requisitos no funcionales complementan los requisitos funcionales, es decir, se enfocan en la especificación de criterios que pueden utilizarse para juzgar la operación de un sistema. Por tanto, se refieren a todos los requisitos que no describen información a guardar, ni funciones a realizar (Pressman, 2002). Los requisitos no funcionales del Sistema para la publicación y distribución de contenidos multimedia son descritos a continuación.

Seguridad

- Controlar los permisos de acceso, escritura, lectura en dependencia del rol que desempeñe cada usuario del sistema.
- El sistema permitirá que un usuario solamente esté autenticado una vez, es decir, que posea sesión única.
- El sistema cerrará la sesión de un usuario inactivo después de un determinado tiempo.

Rendimiento

- El sistema requiere, para mejorar los problemas planteados, un rendimiento eficiente apoyado en la mínima transferencia de datos entre cliente-servidor.

Apariencia o interfaz de usuario

- El sistema debe poseer una interfaz de usuario amigable, brindando facilidades que permitan interactuar y navegar con el sistema de forma fácil y rápida, sin necesidad de conocimientos avanzados.

Software

- Usar preferentemente para los servidores el sistema operativo Ubuntu Server 12.04 LTS.
- El sistema que se usará para la reproducción de los archivos de audio y video será el servidor

Flumotion Streaming Server.

Restricciones del diseño y la implementación

- Las interfaces destinadas al usuario, deben programarse usando HTML5 y como generador de plantillas Twig y ser compatible con las versiones de los navegadores Microsoft Internet Explorer 11, Mozilla Firefox 80, y Chromium 78.
- Las computadoras clientes deben tener un navegador web (recomendado Mozilla Firefox 80).
- La implementación del producto se hará usando herramientas Open Source, sobre la filosofía de Software Libre.
- Solo se procesarán imágenes con los siguientes formatos: JPG, PNG y GIF.
- Solo se procesará audio y videos con los siguientes formatos: webm, ogg, ogx y ogv.

Hardware

- Se necesita un servidor con al menos 16GB de memoria RAM, microprocesador de cuarta generación Intel Inside CORE i7, tarjeta NVidea con al menos 2GB de procesamiento gráfico. Se necesita un servidor que contenga al menos 8TB de espacio de almacenamiento.

Modelo de caso de uso del sistema

El modelo de casos de uso contiene actores, casos de uso y las relaciones que existen entre ellos. Además, describe el comportamiento del sistema en diferentes condiciones, mientras responde a las peticiones realizadas por los usuarios. Un actor es una entidad externa del sistema que participa en cómo se desarrolla el caso de uso en la aplicación (Pressman, 2002).

Diagrama de caso de uso del sistema

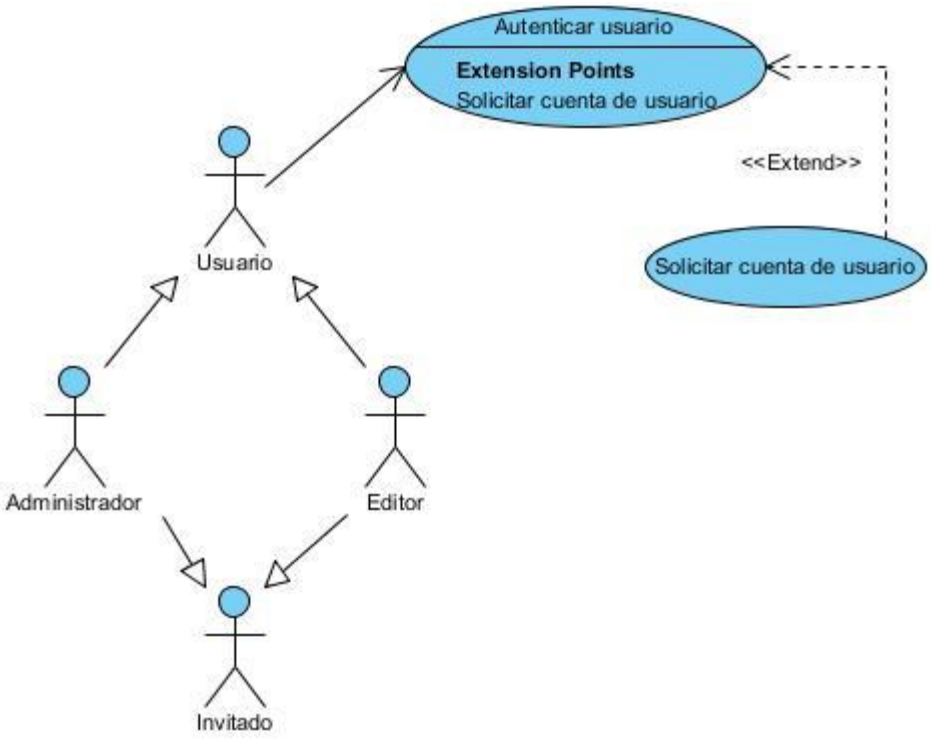


Figura 2. *Diagrama de caso de uso del sistema, por usuario*

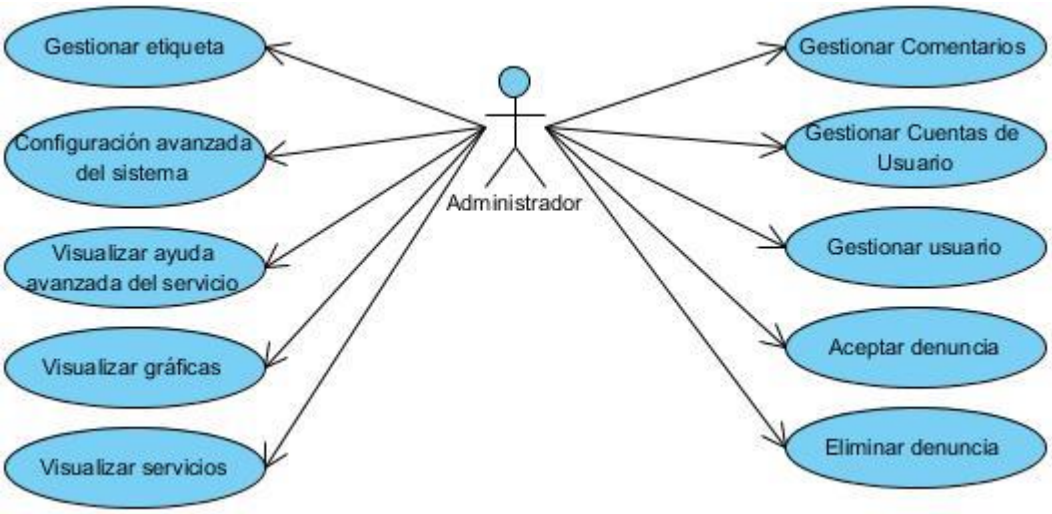


Figura 3. *Diagrama de caso de uso del sistema, por el rol "Administrador"*

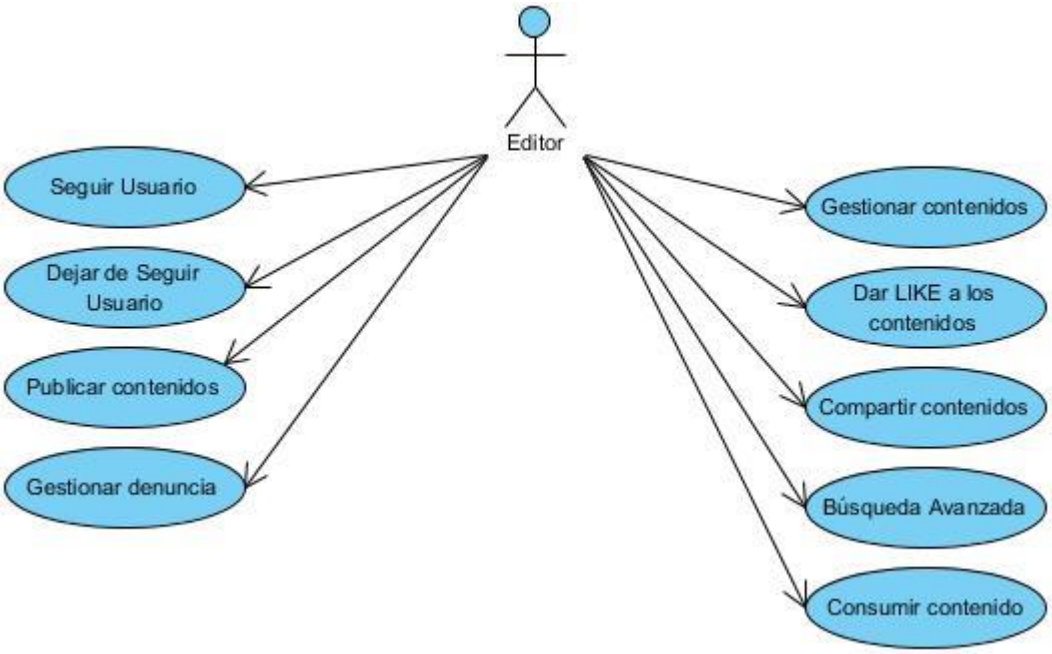


Figura 4. *Diagrama de caso de uso del sistema, por rol "Editor"*

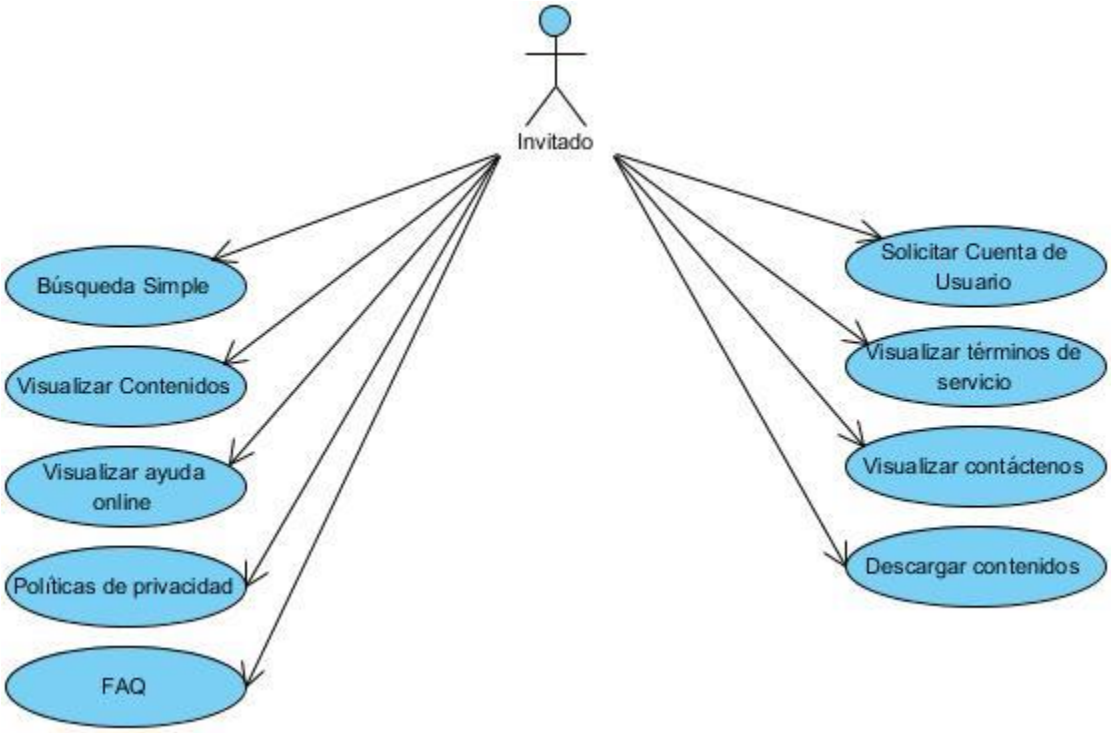


Figura 5. *Diagrama de caso de uso del sistema, por rol "Invitado"*

2.4 Descripción de estilos arquitectónicos y patrones de diseño

Patrones arquitectónicos

Los patrones arquitectónicos ofrecen soluciones a problemas de arquitectura de software dentro de la ingeniería de software. Un patrón arquitectónico expresa un esquema de organización estructural esencial para un sistema de software que consta de subsistemas, sus responsabilidades e interrelaciones. En comparación con los patrones de diseño, los patrones arquitectónicos tienen un mayor nivel de abstracción (Pressman, 2002).
Symfony2 es el marco de trabajo que ha sido seleccionado para solucionar el problema planteado. Para comenzar, separa la lógica del negocio, la lógica del servidor y la presentación de la aplicación web, por lo que Symfony2 en sí mismo usa el patrón arquitectónico Modelo Vista Controlador (MVC) el cual se describe a continuación.
Cuando el usuario solicita ver la portada del sitio, internamente sucede lo siguiente:

- El sistema de enrutamiento determina qué Controlador está asociado con la página de la portada.
- Symfony2 ejecuta el Controlador que renderiza la portada.
- El Controlador solicita al Modelo los datos de la petición.
- Con los datos devueltos por el Modelo, el Controlador solicita a la Vista que cree una página mediante una plantilla y que inserte los datos del Modelo.
- El Controlador entrega al servidor la página creada por la Vista.

El funcionamiento interno de Symfony2 siempre es el mismo, pero se pueden implementar funcionalidades complejas y con mayor organización del código fuente:

- El Controlador manda y ordena.
- El Modelo busca la información que se le pide.
- La Vista crea páginas con plantillas y datos.

Patrón arquitectónico MVC

Es un patrón de arquitectura de software que separa los datos y la lógica del negocio de una aplicación web, de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Para lograr dicho objetivo este patrón propone la construcción de tres componentes distintos: el modelo, la vista y el controlador. Por un lado, define componentes para la representación de la información, y por otro lado para la interacción del usuario. Es un patrón de diseño que se basa en la reutilización de código y la separación de conceptos y características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento (Pressman, 2002).

- El **Modelo:** es la representación de la información con la cual el sistema opera, es decir, la lógica del negocio del sistema.
- El **Controlador:** gestiona las solicitudes del usuario, selecciona el comportamiento del modelo y

responde al usuario mostrando una vista.

- La **Vista:** transforma el modelo en una representación visual con la cual el usuario puede interactuar, actualiza las solicitudes del modelo y presenta la vista seleccionada por el controlador.

Patrones de diseño

Son aquellos que expresan esquemas para definir estructuras de diseño o las relaciones que existen entre ellos a partir de las cuales se puede construir sistemas de software. Proponen una forma de reutilizar experiencias de los desarrolladores, para ello clasifican y describen formas de solucionar problemas frecuentes durante el desarrollo de software. Estos se aplican a un elemento específico como un agregado de componentes para resolver un determinado problema de diseño, relaciones entre los componentes o los mecanismos para efectuar la comunicación componente a componente (Pressman, 2002).

En el diseño de la propuesta de solución se tuvo en cuenta la utilización de los Patrones Generales de Software para Asignación de Responsabilidades (GRASP, por sus siglas en inglés), los cuales se enfocan en los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones y los patrones GoF, utilizados para ocultar la complejidad del sistema. Los patrones de diseño tienen como características principales:

- Son soluciones concretas. Proponen soluciones a problemas concretos, no son teorías genéricas.
- Son soluciones técnicas. Indican resoluciones técnicas basadas en Programación Orientada a Objetos (POO). En ocasiones tienen más utilidad con algunos lenguajes de programación y en otras son aplicables a cualquier lenguaje.
- Se utilizan en situaciones frecuentes. Debido a que se basan en la experiencia acumulada al resolver problemas reiterativos.

Favorecen la reutilización de código. Ayudan a construir software basado en la reutilización, a construir clases reutilizables. Los propios patrones se reutilizan cada vez que se vuelven a aplicar.

Patrones GRASP

De los patrones GRASP en la propuesta de solución se utilizaron los siguientes:

Experto

Permite asignar una responsabilidad al experto en información, o sea, a la clase que cuenta con la información necesaria para cumplir una determinada responsabilidad.

Symfony2 utiliza este patrón con la inclusión de Doctrine para el mapeo de bases de datos. Se utiliza específicamente para crear una capa de abstracción en el modelo, encapsular toda la lógica de los datos y generar las clases entidades con todas las funcionalidades comunes (GET, SET y el constructor de la entidad); las clases de abstracción de datos poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan y contienen la información necesaria de la tabla asociada.

Beneficios del patrón Experto:

- Se conserva el encapsulamiento, ya que, los objetos se valen de su propia información para realizar todas las peticiones. Esto posibilita tener sistemas de fácil mantenimiento.
- El comportamiento se distribuye entre las clases que cuentan con la información requerida para cumplir con la responsabilidad asignada.

Creador

Su implementación permite identificar quién debe ser el responsable de la creación o instanciación de nuevos objetos o clases. Dicho de otra manera, este patrón plantea que se debe asignar a una clase A la responsabilidad de crear una instancia de una clase B. Como la creación de objetos es una de las actividades más frecuentes en un sistema orientado a objetos, es importante el uso de este para guiar la asignación de responsabilidades relacionadas con la creación de objetos.

Este patrón es utilizado en la implementación de las clases controladoras donde se encuentran las acciones definidas para el sistema. En dichas acciones se crean los objetos de las clases que representan las entidades.

Beneficios del patrón Creador:

- Se crean menos dependencias y existe mayor posibilidad de reutilización de código.

Controlador

El patrón controlador sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es el que recibe los datos del usuario y los envía a las distintas clases según el método encargado de la ejecución.

Se manifiesta en todo el sistema debido a que cada uno de los eventos generados por el usuario son atendidos por el archivo *routing.yml,* el encargado de redirigir la petición al método de una clase controladora que realice las operaciones solicitadas, pero siempre manteniendo las clases controladoras sin sobrecarga, es decir, manteniendo siempre la alta cohesión.

Alta cohesión

La alta cohesión hace referencia a cuanta responsabilidad tiene una determinada clase controladora, o sea, una clase debe tener responsabilidades moderadas. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.

Symfony2 permite la organización del trabajo en cuanto a la estructura del proyecto y la asignación de responsabilidades con alta cohesión. Se puede observar en el sistema, ya que, cada clase controladora maneja solamente las responsabilidades correspondientes a las entidades con las que se relaciona, además para cada vista existe una página controladora encargada de manejar sus solicitudes.

Beneficios del patrón Alta cohesión:

- Mejora la claridad y la facilidad con que se entiende el diseño.

- Se simplifica el mantenimiento y las mejoras en funcionalidad.

Bajo acoplamiento

Este patrón plantea que se deben asignar las responsabilidades de forma tal que las clases dependan del menor número de clases que sea posible. Resuelve el problema de cómo dar soporte a una dependencia escasa y a un aumento de la reutilización. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras, con que las conoce y con que recurre a ellas.

En Symfony2 las clases controladoras heredan de la clase *Controller* alcanzando de esta manera un bajo acoplamiento. Las clases que implementan la lógica del negocio y de acceso a datos se encuentran en el modelo, y no tienen asociaciones con las de la vista o el controlador, lo que proporciona bajas dependencias.

Beneficios del patrón Bajo acoplamiento:

- Fáciles de entender por separado.
- Fáciles de reutilizar.

Patrones GoF

Los patrones GoF (del inglés *Gang-of-Four*) más conocidos como "*Pandilla de los Cuatro*" son utilizados básicamente para ocultarle a los usuarios la complejidad de un sistema, mostrándole solamente lo que necesitan ver (Guerrero, 2013). A continuación se describen los patrones *GoF* utilizados y cómo *Symfony2* los implementa.

Decorador

Añade funcionalidad a una clase de manera dinámica, proporcionando una alternativa flexible a la especialización mediante herencia, cuando se trata de añadir funcionalidades.

Tiene como ventaja aportar una mayor flexibilidad que la herencia estática, permitiendo, entre otras cosas, añadir una funcionalidad dos o más veces, evita concentrar en lo alto de la jerarquía clases "guiadas por las responsabilidades", es decir, pretenden, en vano, satisfacer todas las posibilidades y de esta forma las nuevas funcionalidades se componen de piezas simples que se crean y se combinan con facilidad.

En Symfony2 este patrón es fácilmente visible, ya que, la vista se separa por niveles, en hasta 3 niveles, una plantilla base y varias plantillas que heredan de esta. Normalmente, la plantilla base es global en toda la aplicación y contiene el código HTML que es común a la mayoría de las páginas.

Comando

Encapsula una operación en un objeto, permitiendo ejecutar dicha operación sin necesidad de conocer el contenido de la misma. Este patrón, al encapsular en un objeto la acción, promueve una separación entre dicha acción y la petición que resuelve, lo cual redunda en una mayor flexibilidad en todo lo relativo a la acción. Esto trae como consecuencia que diferentes objetos puedan ejecutar la misma acción sin necesidad de repetir su declaración e implementación, se pueden añadir nuevas acciones sin tocar las clases ya existentes, entre otras.

2.5 Modelo de Diseño

El modelo de diseño es una abstracción del Modelo de Implementación y su código fuente, el cual fundamentalmente se emplea para representar y documentar su diseño. Es usado como entrada esencial en las actividades relacionadas con la implementación. Representa a los casos de uso en el dominio de la solución. El Modelo de Diseño puede contener: diagramas, clases, paquetes, subsistemas, cápsulas, protocolos, interfaces, relaciones, colaboraciones, atributos, realizaciones de los casos de uso, entre otros que se puedan considerar para el sistema en desarrollo (Pressman, 2002).

Diagramas de clase del diseño

A continuación se muestra el diagrama de clase del diseño del principal caso de uso del sistema en el cual se expresan las principales relaciones entre las clases. La única diferencia notable con los diagramas tradicionales es que cuentan con diversos estereotipos de UML adicionales que permiten modelar aplicaciones para la web.

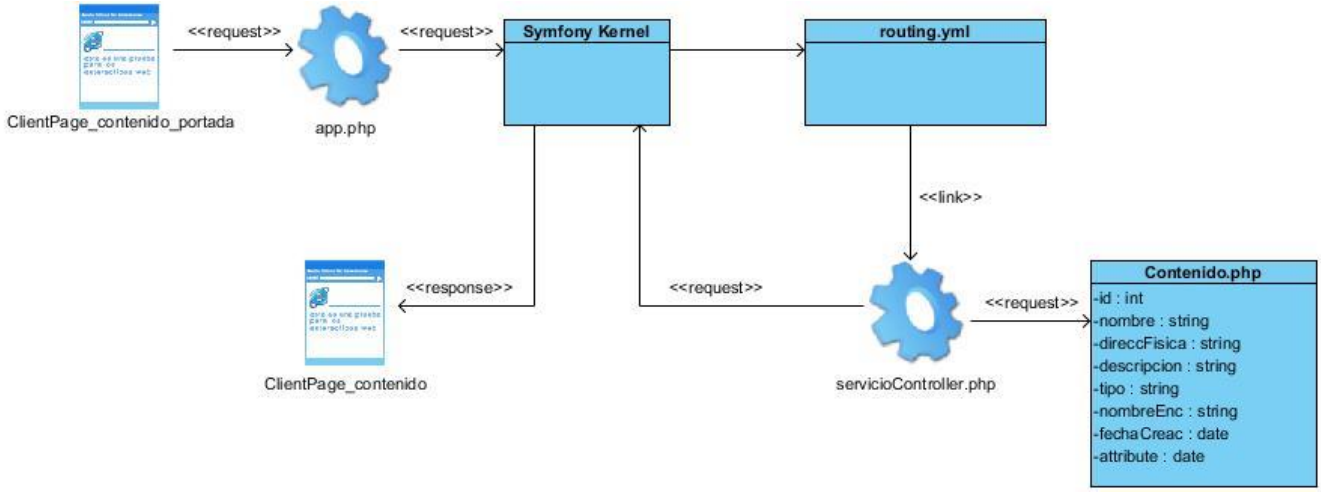


Figura 7. *Diagrama de Clase del Diseño CU "Consumir contenido multimedia"*

2.6 Diagrama de interacción

El diagrama de interacción, representa la forma de comunicación en petición a un evento entre un actor y las clases con las que interactúa. Dicho de otra manera, muestra una cierta vista sobre los aspectos dinámicos de los sistemas modelados. Esto implica recorrer toda la secuencia de llamadas, de donde se obtienen las responsabilidades. Los diagramas de interacción pueden ser de secuencia o de colaboración.

Diagrama de secuencia

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso.

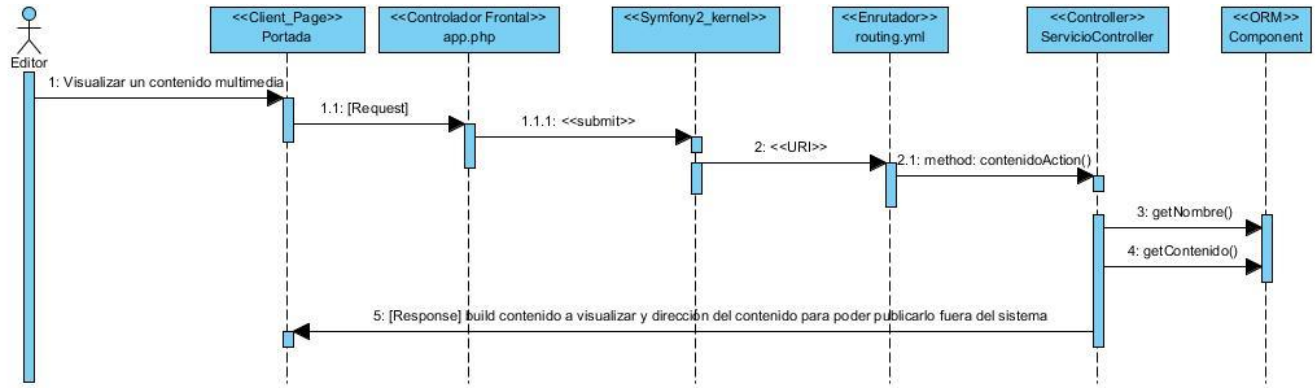


Figura 8. *Diagrama de secuencia del CU "Publicar Contenido"*

Conclusiones del capítulo

- El levantamiento de los requisitos funcionales permitió tener una mejor visión sobre los requerimientos que debía tener el sistema para su posterior modelación.
- El diseño de las funcionalidades solicitadas permitió profundizar en el problema a resolver, así como representar una vista interna del sistema en la que se refinaron los requisitos y se estructuraron en base a clases y paquetes.
- El reconocimiento y definición de patrones arquitectónicos y de diseño establecieron las bases para fomentar la reutilización y las buenas prácticas de programación durante la implementación del sistema.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

La etapa de implementación es decisiva en el desarrollo de software, es en este momento donde se define y organiza el código de la propuesta de solución. Entre los principales objetivos cabe destacar la implementación de los diferentes elementos del diseño y los artefactos obtenidos del capítulo anterior. Antes de que un sistema de software pueda ser usado primero debe ser probado. Durante este proceso se deben poner en práctica todas las estrategias posibles para garantizar que el usuario final del sistema se encuentre libre de problemas.

3.1. Modelo de componentes

El modelo de componentes ilustra los componentes de software que se usarán para construir un sistema informático. En la ingeniería de software, es una rama que enfatiza en la separación de asuntos por lo que se refiere a la funcionalidad de amplio rango disponible a través de un sistema de software dado. El modelo de componentes también es un acercamiento basado en la reutilización para definir, implementar y agrupar componentes que se encuentran débilmente acoplados en el sistema.

Diagrama de componentes

El diagrama de componentes muestra la relación entre componentes de software, sus dependencias, su comunicación, su ubicación y otras condiciones. En la Figura 9 se ilustra el diagrama de componentes del Sistema para la publicación y distribución de contenidos multimedia.

Descripción de los componentes generales de la propuesta de solución

- **BackEndBundle:** es el *bundle* que se encarga de la administración del sistema, gestión de usuario, gestión de etiquetas, entre otras funcionalidades administrativas.
- **Command:** contiene la clase que se encarga de borrar automáticamente las solicitudes de contraseñas caducadas.
- **Controller:** contiene las clases controladoras del sistema que se encargan de la administración.
- **DataFixtures:** contiene las clases que se encargan de generar los datos iniciales del sistema.
- **DependencyInjection:** se encarga de la comunicación de Symfony2 con los servicios implementados en el BackEndBundle.

- **Entity:** contiene las entidades relacionadas con la administración y los usuarios del sistema.
- **Form:** contiene los formularios que se generan para las operaciones de gestión.
- **Resources:** contiene las vistas, estilos y archivos javascript que se les muestran al usuario como respuesta del sistema
- **Subscriber:** contiene la clase que se encarga de crear las configuraciones iniciales del usuario.
- **Tests:** contiene las pruebas que se ejecutan a cada clase controladora del sistema.
- **Útil:** contiene las clases que representan los servicios locales del propio sistema como la obtención de los parámetros de configuración del servidor *streaming*.

Figura 9. Diagrama del BackEndBundle

3.2. Modelo de despliegue

El modelo de despliegue muestra las relaciones físicas de los distintos nodos que componen un sistema y la distribución de los componentes sobre dichos nodos. Se utiliza como entrada principal en las actividades de diseño e implementación, debido a que la distribución del sistema tiene una influencia fundamental en él. El modelo de despliegue para el Sistema para la publicación y distribución de contenidos multimedia se ilustra en la Figura 10.

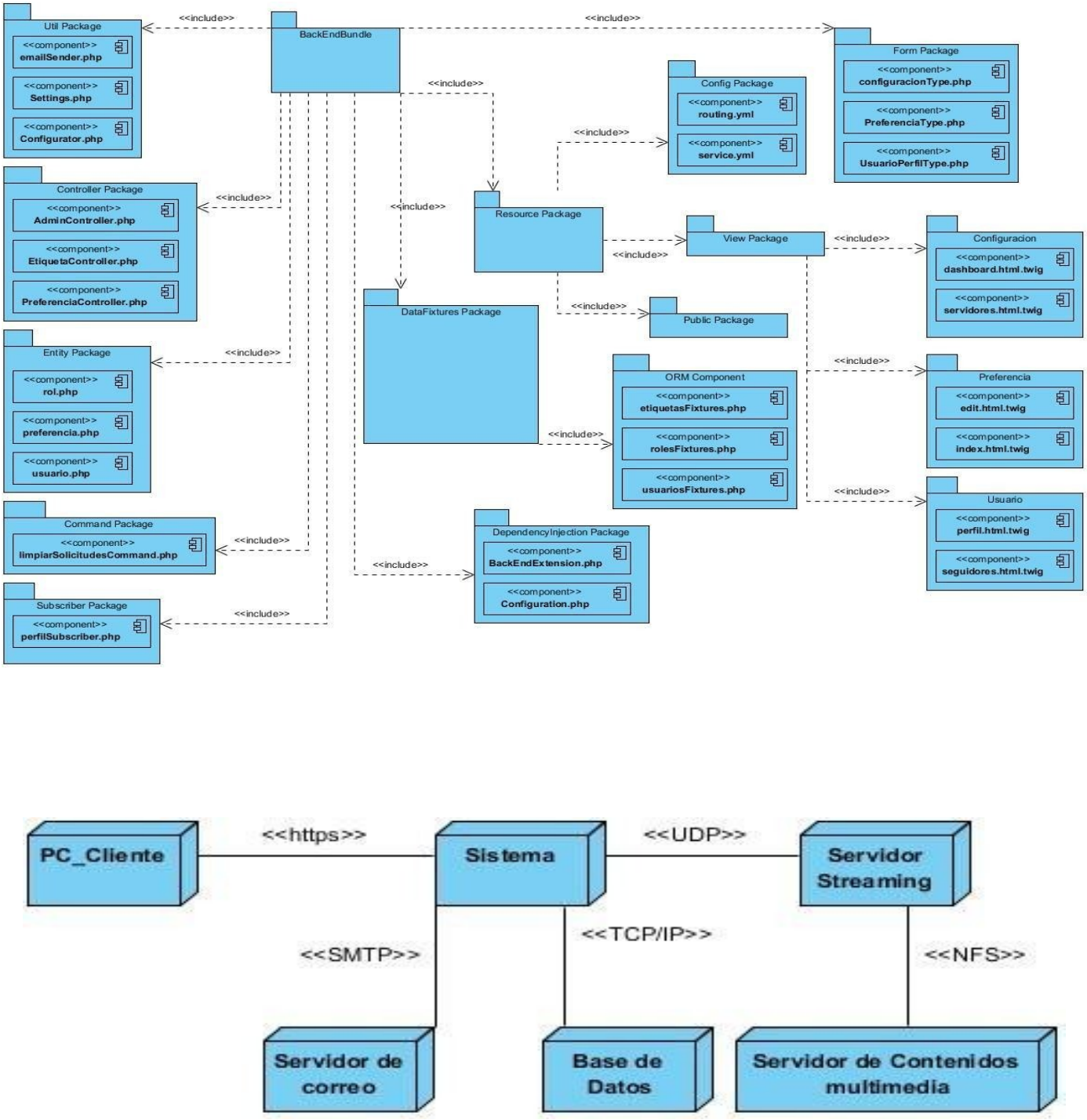


Figura 10. *Diagrama de despliegue*

El despliegue del Sistema para la publicación y distribución de contenidos multimedia está compuesto

principalmente por 6 nodos:

- **PC_Cliente:** funciona como cliente del sistema, en este caso puede ser una aplicación web o un usuario que desee interactuar directamente con la solución, y accederá al mismo mediante el protocolo *HTTPS* a través de un navegador web.
- **Sistema:** hace referencia al Sistema para la publicación y distribución de contenidos multimedia y se comunica mediante el protocolo TCP/IP con el Sistema de Base de Datos, para tener acceso a todo lo referente con datos de usuarios y contenidos multimedia. Además, se comunicará mediante el protocolo UDP con el Servidor *Streaming* para consumir los contenidos multimedia.
- **Servidor Streaming:** posibilita la reproducción de los archivos de audio y video en el Sistema. Cuenta con unidades de almacenamiento del Servidor de Contenidos Multimedia compartidas mediante NFS.
- **Servidor de Contenidos Multimedia:** funciona como unidad de almacenamiento de los contenidos multimedia. Sus unidades de almacenamiento son compartidas a través de NFS en el Servidor *Streaming*.
- **Base de Datos:** representa el Sistema Gestor de Base de Datos que utiliza la aplicación y se comunican mediante el protocolo TCP/IP con el Sistema.
- **Servidor de correo:** es empleado por el Sistema para enviar notificaciones a los usuarios.

3.3. Diseño de base de datos

El diseño de la base de datos es la herramienta conceptual para la descripción de los datos, relaciones entre ellos, semántica y restricciones de consistencia. Tal representación permite contar con una vista del modelo a implementar a través de la perspectiva de entidades.

Modelo de Datos

El modelo entidad-relación (E-R) es un modelo de datos de alto nivel. Está basado en una percepción del mundo real que consiste en una colección de objetos básicos, denominados entidades, y de relaciones entre estos objetos. En la Figura 11 se ilustra el modelo de datos de la aplicación web Sistema para la publicación y distribución de contenidos multimedia.

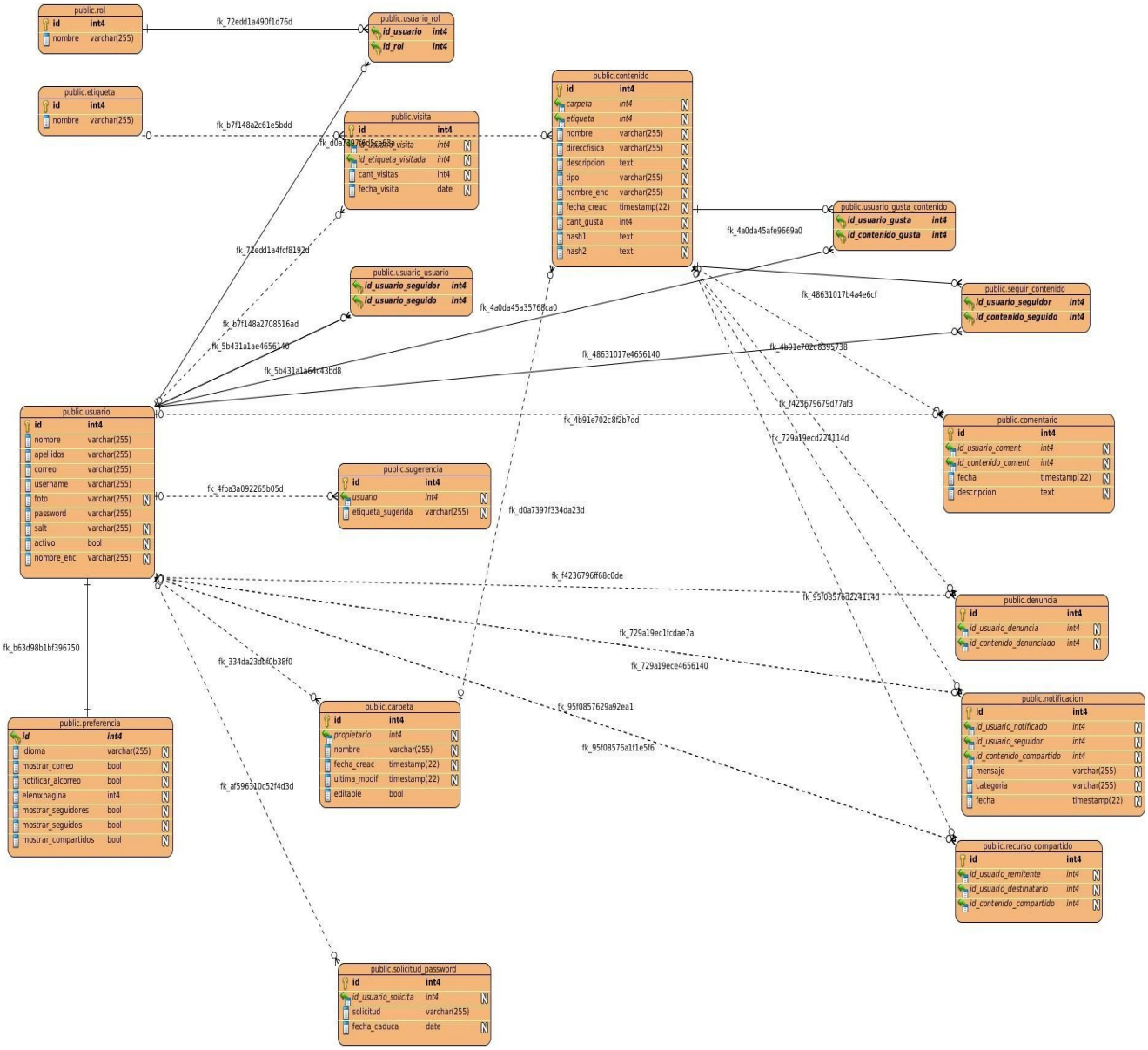


Figura 11. *Modelo físico de la base de datos del sistema*

3.4. Código fuente del Sistema para la publicación y distribución de contenidos multimedia

El código fuente de un programa informático es un conjunto de líneas de texto que describen las instrucciones que debe seguir la computadora para ejecutarlo. Está escrito por uno o varios programadores en algún lenguaje de programación, Java, PHP, Perl, entre otros, pero en este primer estado no es directamente ejecutable por la computadora, sino que debe ser traducido a lenguaje de máquina o código objeto para que pueda ser ejecutado por el hardware de la computadora. Para realizar la traducción se utilizan compiladores, ensambladores, intérpretes y otros sistemas de traducción.

Gracias al uso del marco de trabajo Symfony2 y la arquitectura MVC, el código fuente del Sistema para la publicación y distribución de contenidos multimedia está escrito de manera entendible, organizado y fácil de mantener.

Estándar de codificación

Los estándares de código resultan importantes en cualquier proyecto de desarrollo cuando muchos desarrolladores trabajan en el mismo proyecto. El propósito fundamental de los estándares de codificación es que el sistema en cuestión tenga una arquitectura y un estilo consistente, con lo cual resulte fácil de entender y mantener.

Los estándares de codificación son un complemento a la programación por pares, o sea, en equipo, y no sólo es importante usar un estándar, sino usar un buen estándar de codificación, de esta forma se deberá:

- Clarificar más que confundir.
- Promover la intención del código.
- Permitir que los programas se acerquen lo mejor posible al lenguaje natural.
- Incorporar las mejores prácticas de la codificación.

Entre otros aspectos se apunta a las variaciones en las convenciones o adiciones necesarias en el código del lenguaje PHP especialmente dirigido al trabajo con el *framework* Symfony2. Los principales aspectos incluidos en dicho estándar se describen a continuación:

- Nunca utilizar las etiquetas cortas (<?).
- No finalizar las clases con la etiqueta de cierre (?>).
- La identación se realiza utilizando cuatro espacios (las tabulaciones no están per-

mitidas).

- Agregar un único espacio después de cada delimitador coma.
- No poner espacios después de la apertura de un paréntesis y antes del cierre del mismo.
- Agregar un único espacio alrededor de operadores (==, *&&*).
- Agregar un único espacio antes de los paréntesis de apertura de una palabra clave de control (*if*, *else*, *for*, *while*).
- Agregar una línea en blanco antes de la sentencia *return*.
- No agregar espacios al final de las líneas.
- Utilizar llaves para indicar el cuerpo de las estructuras de control sin importar el número de sentencias que estas contengan.
- Colocar las llaves en sus propias líneas para clases, métodos y declaración de funciones.
- Separar las sentencias condicionales y las llaves de apertura con un único espacio sin dejar una línea en blanco.
- Declarar explícitamente la visibilidad de las clases, métodos y propiedades (el uso de *var* está prohibido).
- Utiliza constantes de tipo PHP nativas en minúsculas: *false*, *true* y *null*. Lo mismo aplica para los *array()*.

- Utilizar letras mayúsculas para constantes, con palabras separadas por un guión bajo.
- Definir una clase por archivo.
- Declarar las propiedades de las clases antes de los métodos.
- Declarar los métodos públicos primero, luego los protegidos y finalmente los privados.

Convención de nombres

- Utilizar camelCase y no guiones bajos, para variables, funciones y nombres de métodos.
- Utilizar guiones bajos para definir opciones, argumentos y nombres de parámetros.
- Utilizar los *namespace* para todas las clases.
- Utilizar *Symfony* como el *namespace* de primer nivel.
- Añadir como sufijo *Interface* a las interfaces.
- Utilizar caracteres alfanuméricos y guiones bajos para nombres de archivos.

3.4. Principal pantalla del Sistema para la publicación y distribución de contenidos multimedia

La portada inicial de cualquier sistema es el elemento más importante, a partir de la impresión que cause al usuario desde el primer momento tendrá la capacidad de definir cuán agradable será la experiencia con la aplicación. En la Figura 12 se muestra la página de inicio de la aplicación para todos los usuarios que accedan al sistema.

Esta interfaz está compuesta por varios botones que indican funcionalidades específicas como por ejemplo "*mediaShare*: Muestra la Portada del sistema" seguido de página de contenidos "Reciente" y a continuación la página de contenidos "Popular"; un buscador en la parte superior, seguido de un botón indicando la búsqueda avanzada, otro que indica la opción para subir un archivo y posteriormente un menú desplegable en el cual se encuentran operaciones que los usuarios autenticados pueden realizar. El área de trabajo de esta vista está dividida en cuatro bloques:

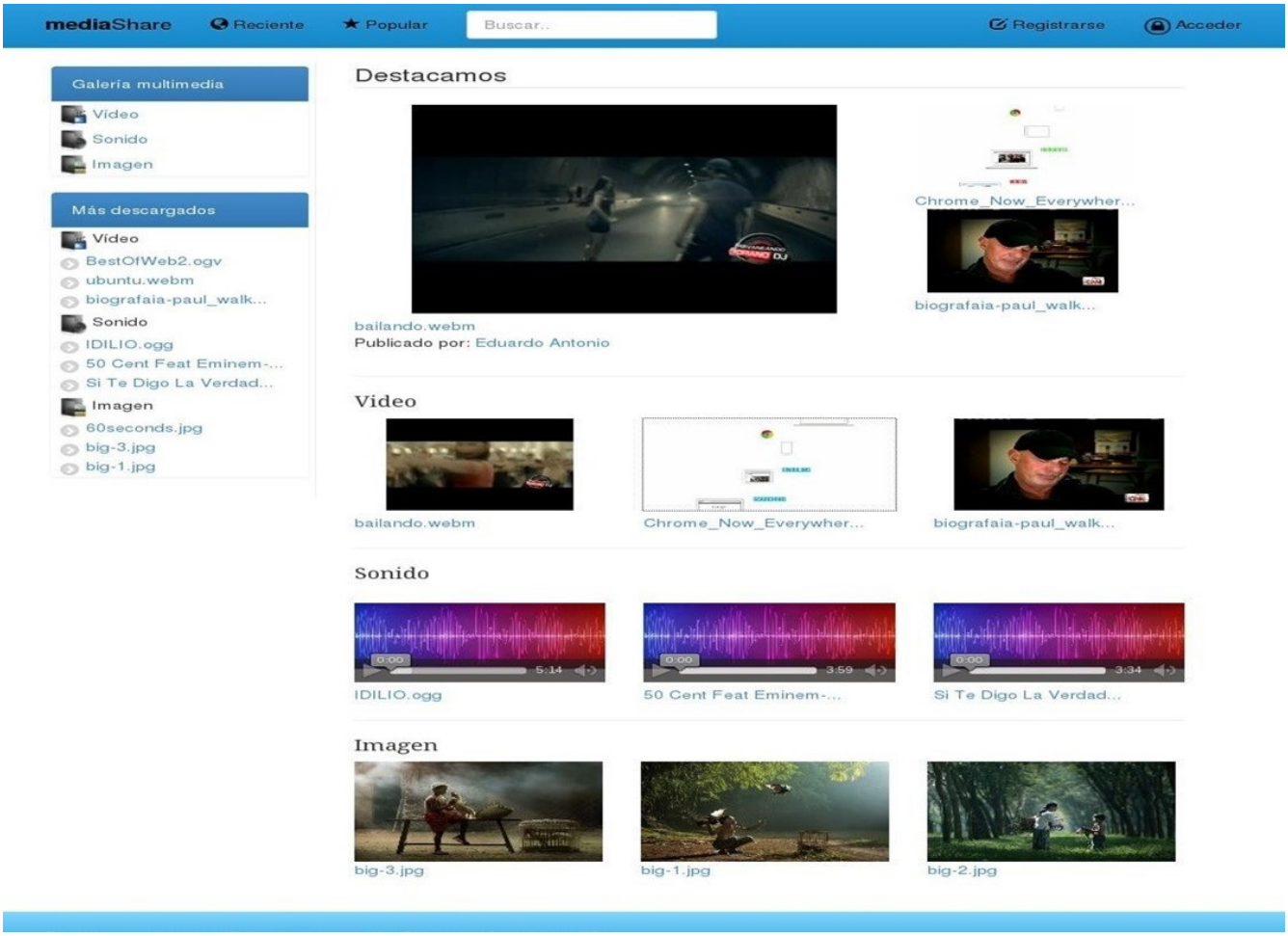


Figura 12. *Pantalla principal del sistema*

Galería multimedia: Contiene las galerías de los tipos de contenidos con los que trabaja el sistema: video, sonido e imágenes.

Más descargados: Contiene los tres contenidos de tipo video, sonido e imagen que más se han descargado.

Usuarios registrados: Contiene los usuarios que se han registrado en el sistema.

Destacamos: Muestra los contenidos más interesantes para el usuario autenticado.

Video: Muestra los videos más populares.

Sonido: Muestra los sonidos más populares.

Imágenes: Muestra las imágenes más populares.

3.5. Validación del sistema

Las pruebas de validación en la ingeniería de software son el proceso de revisión que verifica que el sistema desarrollado cumple con las especificaciones y logra su objetivo (Pressman, 2002). Es una parte del proceso de pruebas de software de un proyecto. La validación es el proceso de comprobar que lo que se ha especificado es lo que el usuario realmente solicitó. Se evalúa el sistema o parte de este, durante o al final del desarrollo, para determinar si satisface los requisitos iniciales. Según Pressman, el proceso de pruebas consta de siete etapas: contenido, interfaz, navegación, componente, configuración, desempeño o rendimiento y seguridad. El equipo de desarrollo junto al cliente realizó un análisis sobre los puntos críticos que no deben fallar en el sistema, por lo que solamente se realizaron pruebas de desempeño o rendimiento, funcionales y de seguridad.

Estrategias de pruebas

Una estrategia de prueba integra los métodos de diseño de caso de pruebas del software en una serie bien planeada de pasos que desembocará en la eficaz construcción del software (Pressman, 2002). La estrategia de prueba proporciona un plano que describe los pasos que se darán, cuándo se darán y cuánto tiempo y esfuerzo se necesitará dedicar a la prueba. Para llevar a cabo una correcta ejecución de las pruebas se tuvieron en cuenta los siguientes pasos:

- Se revisa, de manera constante, el modelo de contenido del sistema para descubrir errores de forma anticipada y continua.
- Se revisa, sistemáticamente, la interfaz del usuario para descubrir errores en la presentación del sistema.
- El sistema se prueba en diferentes ambientes y se toman las experiencias de compatibilidad con sus configuraciones.

- Se realizan pruebas de seguridad con el objetivo de detectar vulnerabilidades que permitan el acceso al sistema con fines maliciosos.
- Se llevan a cabo pruebas de desempeño con el objetivo de medir el rendimiento del sistema.

Pruebas funcionales

Constituyen pruebas de software que tienen como objetivo probar que los sistemas desarrollados cumplan con las funciones específicas para las cuales han sido creados (Pressman, 2002). A este tipo de pruebas se les denomina también pruebas de comportamiento y para realizarlas se emplea el método de caja negra, donde los probadores o analistas de prueba no enfocan su atención en cómo se generan las respuestas del sistema, sino en el funcionamiento de la interfaz. Básicamente el enfoque de este tipo de prueba se basa en el análisis de los datos de entrada y salida, esto generalmente se define en los casos de prueba preparados antes del inicio de las pruebas. Este tipo de pruebas hace referencia a dos dimensiones de la calidad: contenido y función, y evalúa los aspectos de sus dimensiones (Pressman, 2002).

El objetivo final es garantizar que los requerimientos hayan sido cumplidos y que el sistema sea aceptable. A continuación, se muestra una propuesta de diseño de caso de prueba basado en caso de uso a realizar al Sistema para la publicación y distribución de contenidos multimedia.

Diseño de caso de prueba basado en caso de uso

Los casos de prueba son un conjunto de condiciones o variables bajo las cuales los analistas determinan si el requisito de una aplicación es parcial o completamente satisfactorio. A continuación, se describe el diseño de caso de prueba basado en el caso de uso Publicar Contenido.

Caso de uso: Publicar Contenido.

Condiciones de ejecución:

- El usuario debe acceder al navegador web y conectarse al sistema mediante la dirección https://CCNtube.ccnnet.ml/.
- El usuario debe autenticarse en el sistema.
- Solo los usuarios autenticados tienen acceso a esta funcionalidad.

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**
EC 1.1 El Contenido se encuentra en el servidor.	El usuario desea publicar un contenido en una aplicación web externa.	El sistema genera una dirección para el contenido especificado y la devuelve al usuario. 2. El contenido se reproduce desde el servidor *streaming* que implementa la solución.	1. El usuario selecciona el contenido a publicar en una aplicación web externa. 2. El usuario copia la dirección que brinda el sistema y la pega donde desea publicar dicho contenido.

EC 1.2 El contenido se encuentra en el servidor de almacenamiento, pero el servidor *streaming* se encuentra deshabilitado.		El sistema genera un mensaje de error "El servidor *streaming* se encuentra deshabilitado".	1. El usuario selecciona el contenido a publicar en una aplicación web externa.

Tabla 7. *Caso de prueba basado en Caso de uso "Publicar Contenido"*

Resultados de las pruebas funcionales

Para la validación de los requisitos funcionales se realizaron cuatro iteraciones de pruebas donde se ejecutaron 34 diseños de casos de pruebas. En la Figura 13 se muestran los resultados obtenidos en cada una de las iteraciones de pruebas realizadas al Sistema para la publicación y distribución de contenidos multimedia.

No conformidades

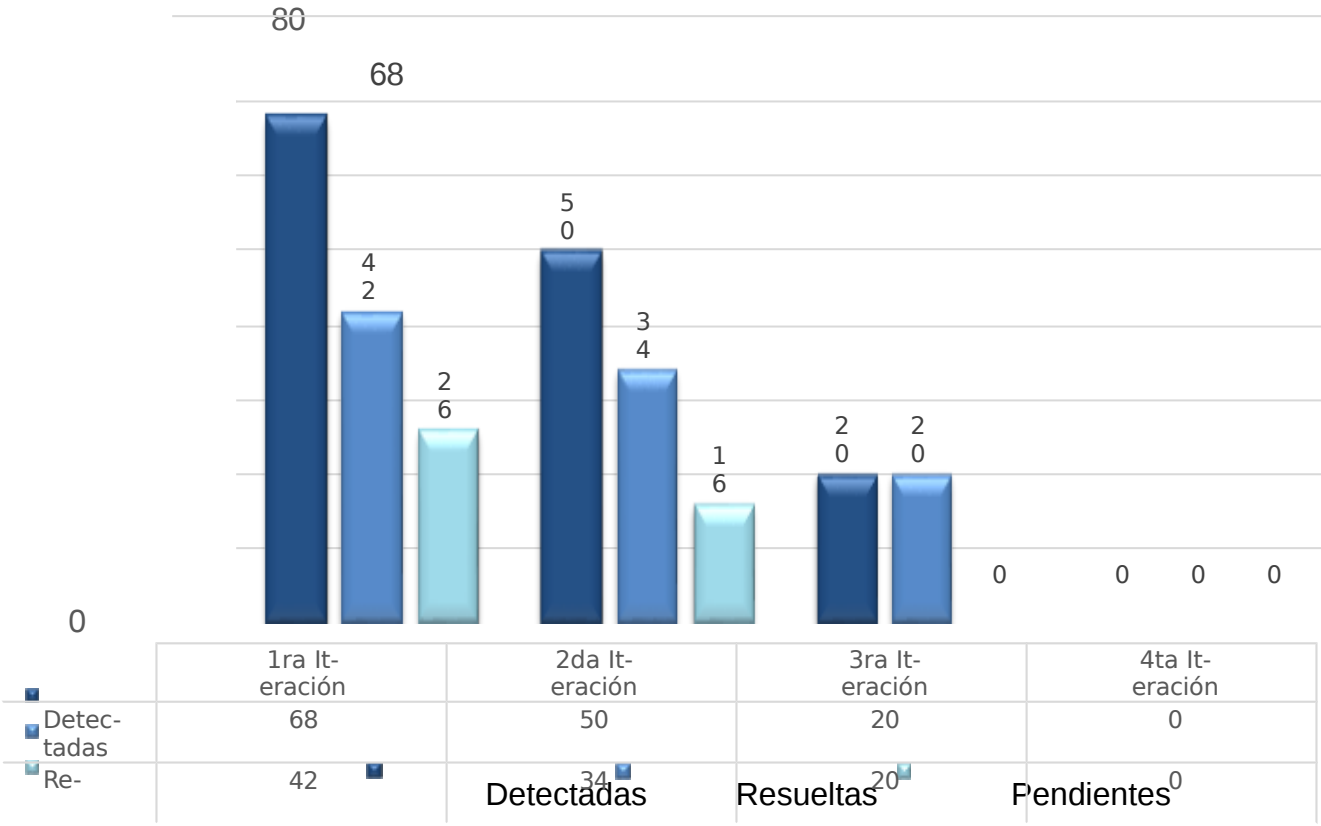


Figura 13. *Comportamiento de las no conformidades por iteraciones*

Entre las no conformidades detectadas durante el proceso de pruebas se destacan las siguientes:

- No se muestran mensajes de alerta cuando el usuario deja campos obligatorios sin completar.
- El resultado de las búsquedas se muestra de forma incorrecta.
- El sistema no comprueba que los servidores asociados a cada uno de los servicios con los que cuenta se encuentran en funcionamiento.
- No se verifican correctamente los formatos asociados a los contenidos multimedia para los que tiene soporte el sistema.

Pruebas de seguridad

Las pruebas de seguridad garantizan que los usuarios estén restringidos a funciones específicas o que su acceso esté limitado únicamente a los datos que están autorizados a acceder. Sólo aquellos usuarios autorizados a acceder al sistema son capaces de ejecutar las funcionalidades disponibles. El objetivo fundamental de este tipo de pruebas es comprobar los niveles de seguridad lógica del sistema.
Se han establecido tres niveles para realizar las pruebas de seguridad. Para evaluar la seguridad de las aplicaciones en un primer nivel (nivel 1) se definió una lista de chequeo que cuenta con 15 indicadores agrupados en cuatro tipos de pruebas accesibles a los tres niveles.
El resultado de la evaluación de dicha lista arrojó, para los cuatro tipos de pruebas, los siguientes resultados:

1. **Pruebas de Autorización:** Ningún usuario estándar puede modificar sus privilegios ni los de otro usuario en la aplicación.

2. **Pruebas de Gestión de Sesiones:** No se puede acceder a la aplicación copiando la URL después de estar autenticado, cerrar el navegador y volver a abrirlo. Tampoco se puede acceder al cerrar la sesión de un usuario y dar *clic* en el botón “Atrás” del navegador.

3. **Validación de Datos:** Se enmascaran datos confiables cuando se visualizan en la aplicación. Solamente se permiten contraseñas alfanuméricas, que incluyan caracteres especiales y que tengan seis caracteres, como mínimo de longitud. La funcionalidad de cambio de contraseña únicamente se permite a usuarios autenticados validando la antigua contraseña y la nueva contraseña. El sistema no muestra mensajes indebidos al colocar en la barra de dirección o en campos de entrada los caracteres: comillas simples (´´), signos de *ampersand* (&), o signos*: + - /.*

4. **Comprobación del Sistema de Autenticación:** Los mensajes de error para distintas combinaciones de autenticación muestran la misma información. Los tiempos de respuestas usuario correcto - contraseña incorrecta y usuario incorrecto - contraseña incorrecta son los mismos. El sistema protege el envío de los datos mediante protocolo seguro (*HTTPS*). Pero aún no se bloquea la sesión del usuario después de una hora de inactividad, ni se bloquea la cuenta del usuario después

de tres intentos de autenticación fallidos.

Esta lista de chequeo permitió recoger los puntos eficientes e ineficientes que tienen los elementos comprobados, así como también la verificación de que el grado de seguridad de la aplicación es adecuado para la protección de la información.

Para evaluar la seguridad de la aplicación en un segundo nivel (nivel 2) se hizo uso de la herramienta Acunetix WVS v8. Durante el escaneo realizado por la herramienta fueron detectadas 15 vulnerabilidades, de ellas ocho de prioridad baja, una de prioridad media y seis informativas. Entre las no conformidades de prioridad baja se encontraron: ficheros del servidor Apache vulnerables a ataques por fuerza bruta y *cookies* de sesión sin el indicador de seguridad, no se corrigieron estas no conformidades debido a que no se encontraban dentro del alcance del equipo de trabajo. La no conformidad de prioridad media está relacionada con el autocompletamiento de los formularios, y fue corregida. El resto deja al descubierto información acerca de una página de error que contiene la versión del servidor web y una lista de los módulos habilitados en este servidor, lo cual no constituye un problema propio del sistema desarrollado, sino del servidor donde se ejecuta.

Pruebas de rendimiento

Mediante las pruebas de rendimiento es posible hallar tendencias y comportamientos para los elementos de una aplicación, los cuales generan bajo rendimiento. Este tipo de pruebas permite identificar cuellos de botella, capacidad de concurrencia de usuarios, tiempos de respuesta de operaciones del negocio a nivel de sistema, establecer un marco de referencia para pruebas futuras, determinar el cumplimiento de los objetivos de rendimiento y requerimientos no funcionales, entre otros (V&V Quality, 2013).

Pruebas de carga: Mediante la ejecución de las pruebas de carga es posible identificar la capacidad de recuperación de un sistema cuando es sometido a cargas variables, tanto de usuarios como de procesos. Al realizar las pruebas de carga se puede determinar el tiempo de respuesta de todas las transacciones críticas del sistema y encontrar cuellos de botella de la aplicación (V&V Quality, 2013).

Prueba de estrés: Mediante las pruebas de estrés es posible identificar la capacidad de respuesta de un sistema bajo condiciones de carga extrema, representadas por una alta concurrencia de usuarios y procesos. Una vez realizadas las pruebas de estrés se podrá conocer el punto de quiebre del aplicativo en términos de capacidad de respuesta, con lo cual será posible establecer acciones de optimización en diferentes niveles para asegurar una mejor capacidad de concurrencia de usuarios y procesos, que se verá reflejada en una óptima operación de negocio (V&V Quality, 2013).

Resultados de las pruebas de rendimiento

Para las pruebas de rendimiento, específicamente pruebas de Carga y Estrés, realizadas al Sistema para la publicación y distribución de contenidos multimedia fue utilizada la herramienta *JMeter* v2.8.4 la cual está implementada en el lenguaje de programación *Java*, y permite realizar pruebas de rendimiento y funcionales sobre aplicaciones web.

JMeter es un software de código abierto diseñado para pruebas de carga de comportamiento funcional y la medición del rendimiento. Originalmente fue diseñado para probar las aplicaciones web, pero desde entonces se ha expandido a otras funciones de prueba. Es utilizado para probar el rendimiento tanto en los recursos estáticos como dinámicos. Puede ser utilizado para simular una carga pesada en un servidor de red o un objeto para poner a prueba su resistencia o para analizar el rendimiento general en diferentes tipos de carga (Martínez, 2009).

Hardware de prueba (PC servidor):

- Tipo de procesador: *Intel (R)* Core (TM) i3 4130 @ 3.4GHz (2CPUs)
- Memoria RAM: 4,00 GB
- Tipo de Red: Ethernet 10/100Mbps

Hardware de prueba (PC servidor):

- Tipo de procesador: *Intel (R)* Core (TM) i5-8400 CPU @ 2.8GHz
- Memoria RAM: 8,00 GB
- Tipo de Red: Ethernet 10/100Mbps

Software instalado en ambas PC:

- Tipo de servidor web: Apache/2.2.22
- Memoria máxima: 2048 MB
- Máximo de hilos concurrentes: 150
- Plataforma: SO Ubuntu 12.04.2 LTS (precise)
- Servidor de BD: PostgreSQL v14

La realización de las pruebas de rendimiento a pesar de que no fueron ejecutadas sobre hardware de altas prestaciones, arrojó resultados que ofrecen un acercamiento al funcionamiento del sistema bajo cierta cantidad de solicitudes. Para llevar a cabo la ejecución de dichas pruebas se tomó un ambiente de simulación con un total de 150 usuarios conectados concurrentemente y con período entre cada petición, de los usuarios, de un segundo. Bajo las citadas condiciones, el sistema retornó los siguientes resultados:

Reporte resumen

Nombre: Reporte resumen

Comentarios

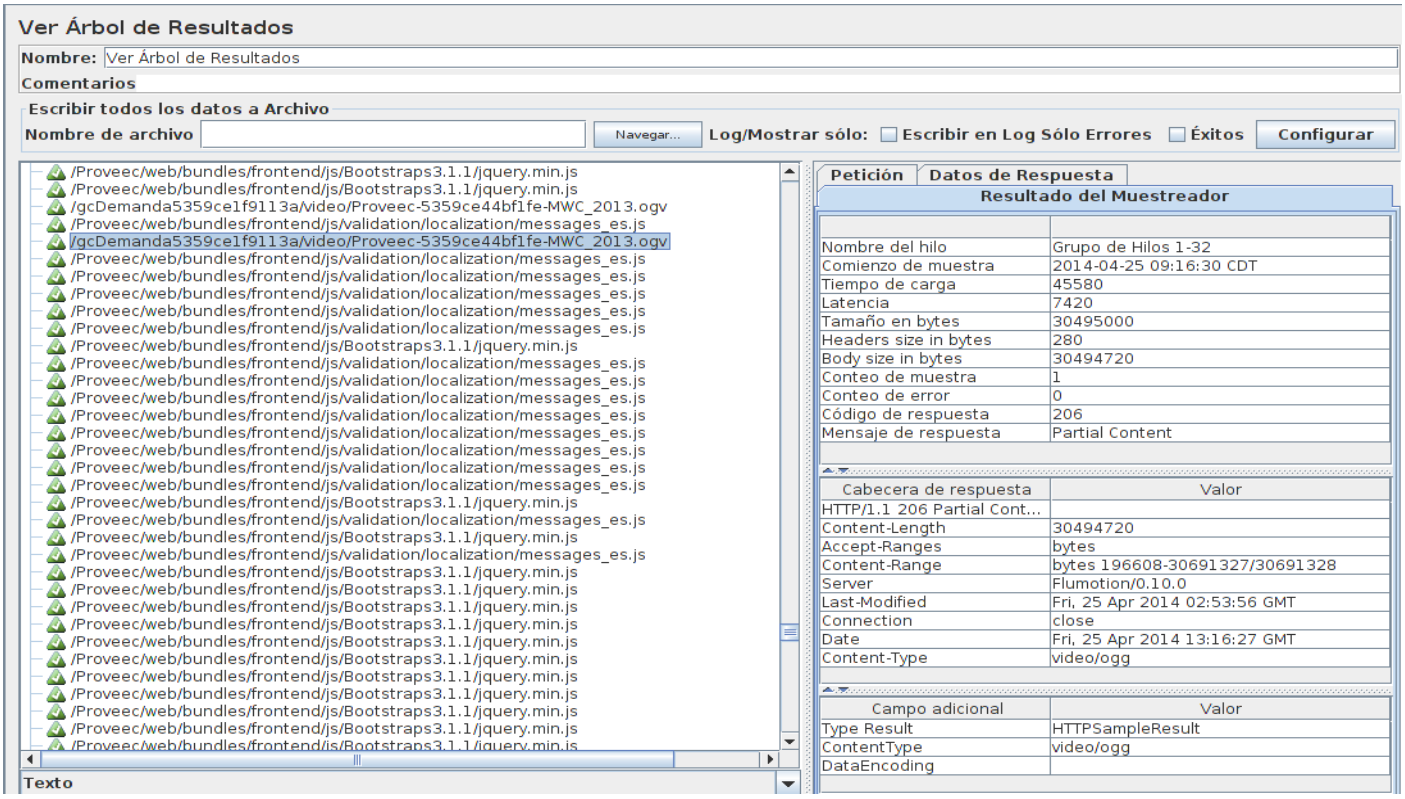
Escribir todos los datos a Archivo

Nombre de archivo Navegar... **Log/Mostrar sólo:** Escribir en Log Sólo Errores Éxitos Configura

Etiqueta	# Muestras	Media	Mín	Máx	% Error	Rendimiento	Kb/sec	Media de B
/Proveec/web/a...	150	245788	2902	550650	9,33%	16,2/min	0,89	33
/Proveec/web/b...	150	16421	6	88769	0,00%	16,3/min	4,65	175
/Proveec/web/b...	150	12275	3	58714	0,00%	16,3/min	1,31	49
/Proveec/web/b...	150	7067	2	63994	2,00%	16,4/min	0,21	7
/Proveec/web/b...	150	3823	1	34266	0,67%	18,4/min	0,26	8
/Proveec/web/b...	150	12113	2	68650	0,00%	18,4/min	0,63	20
/Proveec/web/b...	150	3695	1	39373	0,00%	18,4/min	0,26	8
/Proveec/web/b...	150	3608	1	27113	0,00%	18,5/min	0,15	4
/Proveec/web/b...	150	6770	1	44588	0,00%	19,0/min	0,19	6
/Proveec/web/b...	150	4717	2	44587	0,00%	19,0/min	0,41	13
/Proveec/web/b...	150	2512	2	22591	0,00%	20,4/min	0,32	9
/Proveec/web/b...	150	702	3	14174	0,00%	21,1/min	3,16	91
/Proveec/web/b...	150	411	1	14010	0,00%	21,5/min	0,23	6
/Proveec/web/b...	150	577	1	10395	0,00%	21,5/min	0,23	6
/Proveec/web/b...	300	464	1	12771	0,00%	28,0/min	0,40	8
/Proveec/web/b...	150	884	2	20381	0,00%	21,5/min	1,54	44
/Proveec/web/b...	150	537	3	17271	0,00%	21,5/min	2,81	80
/Proveec/web/b...	300	435	4	41687	0,00%	28,1/min	5,01	109
/Proveec/web/b...	300	1031	2	20985	0,00%	28,0/min	1,98	43
/gcDemanda53...	319	106914	49	414440	100,00%	29,8/min	0,16	3
/Proveec/web/b...	300	535	9	9610	0,00%	1,3/sec	43,66	337
/Proveec/web/b...	150	1158	2	12221	0,00%	40,1/min	0,65	10
/gcDemanda53...	19	89036	9885	116475	63,16%	5,1/min	938,22	112350
/Proveec/web/a...	19	2325	609	3582	0,00%	6,5/min	0,37	34
/Proveec/web/b...	19	68	2	512	0,00%	6,6/min	0,28	26
/Proveec/asset...	19	639	5	2775	100,00%	6,7/min	0,06	5
/gcDemanda53...	19	1052	41	7312	100,00%	35,0/min	0,19	3
/gcDemanda53...	19	4707	2100	5784	0,00%	1,2/sec	1802,27	15321
/gcDemanda53...	4	2887	8	9878	0,00%	11,5/min	1535,92	82409
Total	4187	20342	1	550650	9,24%	5,3/sec	367,02	715

Figura 14. *Resultados de la prueba de rendimiento*

Figura 15. *Resultados de las peticiones realizadas a la aplicación y sus respuestas*



Interpretación de resultados de las pruebas de rendimiento

- **Etiqueta:** El nombre de la muestra (conjunto de muestras).
- **# Muestras:** El número de muestras para cada URL.
- **Media:** El tiempo medio transcurrido para un conjunto de resultados.
- **Mín:** El mínimo tiempo transcurrido para las muestras de la URL dada.
- **Máx:** El máximo tiempo transcurrido para las muestras de la URL dada.
- **% Error:** Porcentaje de las peticiones con errores.
- **Rendimiento:** Rendimiento medido en base a peticiones por segundo/minuto/hora.
- **Kb/sec:** Rendimiento medido en Kilobytes por segundo.
- **Media de Bytes:** Tamaño medio de la respuesta de la muestra medido en bytes.

Teniendo en cuenta que se realizaron las pruebas con un total de 150 usuarios concurrentes, con un período de peticiones enviadas al servidor, por los usuarios, de un segundo, equivalente a una petición por cada 0.0066 segundos aproximadamente. Se puede observar que las pruebas se han realizado con un margen de error igual a 9.24% sobre un total de 4187 peticiones realizadas para un rendimiento aproximado de 16,2 peticiones por cada segundo. Atendiendo a la cantidad de peticiones por cada segundo que se enviaron, la velocidad de respuesta de las peticiones enviadas y las prestaciones del *hardware* donde se realizaron las pruebas, se considera que los resultados alcanzados son buenos y considerablemente superables si se mejoran las características del entorno de prueba.

3.6 Conclusiones del capitulo

- La confección del diagrama de componentes ofreció una vista arquitectónica de alto nivel que ayudó al equipo de desarrollo durante la implementación.
- Los estándares de código permitieron adoptar una estructura homogénea que facilita la comunicación y asegura la calidad, menos errores y fácil mantenimiento.
- La detección temprana de errores, mediante la aplicación de pruebas de funcionalidad, seguridad y carga y estrés permitió validar que el sistema trabaja como fue diseñado, que los requisitos no fueron violados y que la implementación fue correcta.

CONCLUSIONES

Al término de la fundamentación teórica en la que se sustentó la presente investigación y del desarrollo y validación del Sistema para la publicación y distribución de contenidos multimedia, se arriba a las siguientes conclusiones:

- Durante la evaluación de las tendencias de aplicaciones similares fueron identificados sistemas innovadores y con un alto nivel de aceptación, que, aunque no satisfacen las expectativas del proyecto ni las necesidades y restricciones del cliente en cuanto a la gestión de contenidos multimedia, permitieron adquirir nuevos elementos que contribuyen al mejoramiento del sistema.
- Durante las entrevistas con el cliente se identificaron la mayoría de los requisitos funcionales y no funcionales, con los que debía cumplir el sistema.
- Con la modelación de esquemas y generación de artefactos, se creó una propuesta de solución que permitió obtener una visión del funcionamiento del sistema de forma adecuada, lo cual guió el desarrollo del mismo.
- Fue comprobada la efectividad de la solución propuesta a partir de los resultados satisfactorios obtenidos en las evaluaciones internas realizadas por el proyecto, pues las no conformidades detectadas durante las iteraciones de pruebas fueron resueltas.

RECOMENDACIONES

Una vez concluida la investigación y el desarrollo del sistema se recomienda:

- Desarrollar nuevas funcionalidades que aporten al sistema características que contribuyan a mejorar el uso del espacio de almacenamiento, como por ejemplo detectar imágenes, y material audiovisual duplicados.
- Agregar a la aplicación soporte para mayor cantidad de contenidos, por ejemplo, documentos.
- Implementar nuevos servicios web, REST, con el objetivo de incrementar la interoperabilidad del sistema.

REFERENCIAS BIBLIOGRÁFICAS

1-Abuín Vences, Natalia, Vinader Segura, Raquel. 2011. EL DESARROLLO DE LA WORLD WIDE WEB EN ESPAÑA: UNA APROXIMACIÓN TEÓRICA DESDE SUS ORÍGENES HASTA SU TRANSFORMACIÓN EN UN MEDIO SEMÁNTICORazón y Palabra [en línea] 2011, 16 (Febrero-Abril): Disponible en: http://www.redalyc.org/articulo.oa?id=199518706065 ISSN 1605-4806

2-Albarello, Francisco, Canella, Rubén, Tsuji, Teresa. 2008. INTERNET COMO MEDIO DE COMUNICACIÓN ESTRATÉGICA EN LA FORMACIÓN DEL PROFESORADO Razón y Palabra [en línea] 2008, 13 (Julio-Agosto):Disponible en: http://www.redalyc.org/articulo.oa?id=199520798007 ISSN 1605-4806

3-Alonso, Jaime. 2005. Comunicar en Internet: el papel interactivo de los sujetos en los nuevos mediosOpción [en línea] 2005, 21 (diciembre): Disponible en: http://www.redalyc.org/articulo.oa?id=31004803 ISSN 1012-1587

4-Alvarez, Angel M., Alvarez, S. y Sánchez, B. 2011. Manual de CSS 3. [En línea] Nuevas características de las especificaciones de Hojas de Estilo en Cascada CSS 3 para diseñar páginas web, 2011. [Citado el: 04 de diciembre de 2013]. Disponible en: http://www.desarrolloweb.com/manuales/css3.html

5-Álvarez, Miguel Ángel. 2001. desarrolloweb. [En línea] Perl, 2001. [Citado el: 22 de marzo de 2014.]. Disponible en: http://www.desarrolloweb.com/articulos/541.php

6-Álvarez, Miguel Ángel. 2008. DOM_1. [En línea] ¿Qué es el DOM?, 2008. [Citado el: 23 de enero de 2014.] Disponible en: http://www.desarrolloweb.com/articulos/que-es-el-dom.html

7-Álvarez, Miguel Angel. 2003. ¿Qué es *Streaming*? [En línea] 2003. [Citado el: 23 de enero de 2014.] Disponible en: http://www.desarrolloweb.com/articulos/482.php

8-Álvarez, Sergio, Gértrudix, Manuel. 2011. Contenidos digitales abierto y participación en la sociedad digitalEnl@ce: Revista Venezolana de Información, tecnología y conocimiento [en línea] 2011, 8 (Mayo-Agosto): [Fecha de consulta: 10 de marzo de

2014] Disponible en: http://www.redalyc.org/articulo.oa?id=82319126006 ISSN 1690-7515

9-Apple Computer, Inc. 1985. multimedia. [En línea] *Audio Interchange File Format: "AIFF"*, 1985. [Citado el: 03 de marzo de 2014.] Disponible en: http://multimedia.cx/mirror/AudioIFF1_2_1.htm

10-Arellano, Ceballos, Aideé C. 2001. Reseña de "Post/Televisión: ecología de los medios en la era de Internet" de Alejandro Piscitelli. Estudios sobre las Culturas Contemporáneas [en línea] 2001, VII (diciembre): Disponible en: http://www.redalyc.org/articulo.oa?id=31601407 ISSN 1405-2210

11-Balduino, Eng. Ricardo. 2007. Introduction to OpenUP (Open Unified Process). [En línea] Agosto de 2007. [Citado el: 3 de diciembre de 2013.] Disponible en: http://www.eclipse.org/epf/general/OpenUP.pdf

12-Brooks, Federick P.1975. *The Mythical Man-Month,* ddison-Wesley, 1975.

13-Cabrera Díaz, Dayani. 2006. *Propuesta de líneamientos para el tratamiento documental de las fotografías de prensa en los medios cubanos.* Universidad de La Habana, La Habana, 2006.

14-Cenart. 2013. Teoría y técnica [En línea] Audio Digital, 2013. [Citado el: 08 de mayo de 2014.] Disponible en: http://cmm.cenart.gob.mx/tallerdeaudio/cursos/cursoardour/Teoria_y_tecnicas/Audiodigital.html

15-Cores Prado, Fernando. 2003. Arquitecturas Distribuidas para Sistemas de Videobajo-Demanda a gran escala. Doctor en ciencias. Departament d´Informática. Universitat Autónoma de Barcelona, 2003.

16-developer.mozilla.org. 2013. Mozilla Developer Network. [En línea] developer.mozilla.org, 2013. [Citado el: 28 de mayo de 2014.] Disponible en: https://developer.mozilla.org

17-Dowling Michael. 2012. Guzzle. [En línea] 2012. [Citado el: 03 de abril de 2014.] Disponible en: http://docs.guzzlephp.org

18-Eguíluz Pérez, J. 2013. *Desarrollo Ágil con Symfony2,* 2013. 618 p.

19-Eguíluz Pérez, J. 2013. Introducción a JavaScript, 2007. 185 p.

20-ETECSA. 2013. Cable submarino ALBA 1 está operativo y se comienzan pruebas para tráfico de internet. [Enlínea] 2013. [Citado el: 12 de abril de 2014.] Disponible en:

http://www.cubadebate.cu/noticias/2013/01/24/cable-submarino-alba-1-esta-operativo-y-se-comienzan pruebas-para-trafico-de-internet/

21-Fielding, Roy Thomas. 2000. *Architectural Styles and the Design of Network-based Software Architectures,* 2000.

22-Fleischman, Luciana, Ginesta, Xavier, López Calzada, Miguel. 2009. LOS MEDIOS ĀLTERNATIVOS E INTERNET: UN ANÁLISIS C̄UALITATIVO DEL SISTEMA MEDIÁTICO ESPAÑO LAndamios. Revista de Investigación Social [en línea] 2009, 6 (Agosto-Sin mes): Disponible en: http://www.redalyc.org/articulo.oa?id=62812720011 ISSN 1870-0063

23-García de Jalón, Javier. 2000. *Aprenda Java como si estuviera en primero.* NAVARRA, UNIVERSIDAD DE NAVARRA, 2000.

24-Giacoia, Andrea. 2009. Orígenes-del-formato-mp3. [En línea] 2009. [Citado el: 03 de marzo de 2014.] Disponible en: http://www.tecnopedia.net/historia-tecno/origenes-del-formato-mp3/

25-Guerrero, Carlos A; Suarez, Johanna M y Gutiérrez, Luz E. 2013. Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web. *Inf. tecnol.* [En línea]. 2013, vol.24, n.3 [citado 2014-05-28], pp. 103-114. Disponible en: http://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0718-07642013000300012&lng=es&nrm=iso ISSN 0718-0764.

26-Guzmán Luna, Jaime A., Torres Pardo, Durley, Ovalle, Demetrio A. 2007. SABIOS: una aplicación de la Web semántica para la gestión de documentos digitales Revista Interamericana de Bibliotecología [en línea] 2007, 30 (Enero-Junio): Disponible en: http://www.redalyc.org/articulo.oa?id=179014344008 ISSN 0120-0976

27-Instituto Superior de Formación y Recursos en Red para el Profesorado. 2008. Audio. [En línea] 2008. [Citado el: 03 de marzo de 2014.] Disponible en: http://www.ite.educacion.es/formacion/materiales/107/cd/audio/audio0102.html

28-Islas, Octavio. 2008. INTERNET EN 2008 Razón y Palabra [en línea] 2009, 14 (Marzo-Abril): Disponible en: http://www.redalyc.org/articulo.oa?id=199520725021 ISSN 1605-4806

29-Jódar Marín, Juan Ángel. 2010. LA ERA DIGITAL: NUEVOS MEDIOS, NUEVOS USUARIOS Y NUEVOS PROFESIONALES Razón y Palabra [en línea] 2010, 15 (Febrero-Abril): Disponible en: http://redalyc.org/articulo.oa?id=199514914045 ISSN 1605-4806

30-Larrondo Ureta, Ainara. 2005. Reseña de "Ciberperiodismo"Sphera Pública [en línea] 2005, Disponible en: http://www.redalyc.org/articulo.oa?id=29700521 ISSN 1180- 9210

31-Martínez, Ander. 2009. *Apache JMeter: Manual de usuario,* 2009.

32-Martínez, Rafael. 2010. PostgreSQL-es. [En línea] 2 de Octubre de 2010. [Citado el: 26 de noviembre de 2013.] Disponible en: http://www.postgresql.org.es/

33-Matroska.org. 2005-2013. Matroska. [En línea] 2005-2013. [Citado el: 03 de marzo de 2014.] Disponible en: http://www.matroska.org/

34-Montañola, Alberto. 2007. Gestor de contenidos de vídeo bajo demanda. Enginyeria Tècnica en Informàtica de Sistemes, Universitat de Lleida, 2007.

35-Monteiro Lazaro, Juliana. 2001. ¿Qué es CSS? [En línea] 2001. [Citado el: 23 de enero de 2014.] Disponible en: http://www.desarrolloweb.com/articulos/26.php

36-Oracle. 2013. netbeans. [En línea] 2013. [Citado el: 16 de enero de 2014.] Disponible en: https://netbeans.org/

37-Otto, Mark y Thornton, Jacob. 2012. librosweb. [En línea] 2012. [Citado el: 28 de enero de 2014.] Disponible en: http://librosweb.es/bootstrap_3/

38-Pastor Pérez, Lluís. 2006. Hacia una gestión integral de los contenidosRUSC. Universities and Knowledge Society Journal [en línea] 2006, 3 (octubre): Disponible en: http://www.redalyc.org/articulo.oa?id=78030203 ISSN 1698-580X

39-Pressman, Roger S. 2002. Como probar aplicaciones web. *INGENIERÍA DE SOFTWARE. Un enfoque práctico*, Mc Graw Hill, 2002.

40-Pressman, Roger S. 2002. Estrategias de prueba del software. *INGENIERÍA DE SOFTWARE. Un enfoque práctico.* s.l.: Mc Graw Hill, 2002.

41-Pressman, Roger S. 2002. Ingeniería de requisitos. *INGENIERÍA DE SOFTWARE. Un enfoque práctico*, Mc Graw Hill, 2002.

42-Pressman, Roger S. 2002. Modelado de diseño para aplicaciones web. *INGENIERÍA DE SOFTWARE. Un enfoque práctico*, Mc Graw Hill, 2002.

43-Pressman, Roger S. 2002. Modelado del análisis. *INGENIERÍA DE SOFTWARE. Un enfoque práctico*, Mc Graw Hill, 2002.

44-Pressman, Roger S. 2002. Software e ingeniería del software. [aut. libro] Pressman Roger S. *INGENIERÍA DE SOFTWARE. Un enfoque práctico.* s.l.: Mc Graw Hill, 2002.

45-RAE. 2014. Real Academia de la lengua española. [En línea] 2014. [Citado el: 26 de febrero de 2014.] Disponible en: http://lema.rae.es/

46-Ribelles García, A. 2013. *Plataformas de publicación y distribución de audio y vídeo.* Cataluña. España: FUOC. Fundación para la Universitat Oberta de Catalunya, 2013.

47-Rigo, Marisa. 2008. *Roxana Morduchowicz: La generación multimedia. Significados, consumos y prácticas culturales de los jóvenes.* s.l.: Paidós, 2008.

48-Sandoval, Martín, María Teresa. 2000. Medios de comunicación y publicidad en Internet. Revista Latina de Comunicación Social [en línea] 2000, 3 (diciembre): Disponible en: http://www.redalyc.org/articulo.oa?id=81933610

49-Sanguineti de, Brasesco, Susana. 2001. Estética en la comunicación audio. Revista Latina de Comunicación Social [en línea] 2001, 4 (enero): Disponible en: http://www.redalyc.org/articulo.oa?id=81943706

50-Sommerville, Ian. 2005. *Ingeniería del Software.* Madrid: Pearson Educación, 2005.

51-Soto Arango, Diana, Puig-Samper Mulero, Miguel Ángel. 2012. Documentos Revista Historia de la Educación Latinoamericana [en línea] 2012, 14 (Julio-

Diciembre): Disponible en: http://www.redalyc.org/articulo.oa?id=86926976016 ISSN 0122-7238

52-Theora.org. 2014. theora. [En línea] 2014. [Citado el: 26 de febrero de 2014.] Disponible en: http://www.theora.org/

53-V&V Quality. 2013. V&V Quality. [En línea] 2013. [Citado el: 24 de abril de 2014.] Disponible en: http://www.vyvquality.com/w1/index.php?option=com_content&view=article&id=91&Itemid=162

54-V&V Quality. 2013. V&V Quality. [En línea] 2013. [Citado el: 24 de abril de 2014.] Disponible en: http://www.vyvquality.com/w1/index.php?option=com_content&view=article&id=89&Itemid=160

55-V&V Quality. 2013. V&V Quality. [En línea] 2013. [Citado el: 24 de abril de 2014.] Disponible en: http://www.vyvquality.com/w1/index.php?option=com_content&view=article&id=78&Itemid=150

56-Visual Paradigm International. 2013. visual-paradigm. [En línea] 2013. [Citado el: 15 de diciembre de 2013.] Disponible en: http://www.visual-paradigm.-com/

57-wordpress.org. 2009. projectwp.wordpress. [En línea] 30 de junio de 2009. [Citado el: 26 de febrero de 2014.] Disponible en: http://projectwp.wordpress.com/tag/mo-zilla/

58-Wordreference. 2014. Wordreference. [En línea] 2014. [Citado el: 26 de febrero de 2014.] Disponible en: http://www.wordreference.com/

59-xiph.org. 2013. Ogg. [En línea] 2013. [Citado el: 03 de marzo de 2014.] Disponible en: http://xiph.org/ogg/

60-Zapata, Ros, Miguel. 2002. Las buenas maneras en Internet RED. Revista de Educación a Distancia [en línea] 2002, (octubre): Disponible en: http://www.redalyc.org/articulo.oa?id=54700501