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

Sistema para la Publicación y Distribución de Contenidos Multimedia

Autores:

Eduardo Antonio Roque Díaz

Edel Alejandro Sariol Soto

Tutor:

Ing. Aylín Estrada Velazco

Ing. Máxora R. Castro Pérez

Ing. Adriam Balceiro Rodríguez

Consultantes:

Msc. Leonardo Herrera Boza

Ing. Gilberto Lissabet Hernández

La Habana, 2014

Declaración de autoría

Declaramos ser los únicos autores de la presente investigación y autorizamos al Centro CIDI de la Universidad de las Ciencias Informáticas a hacer el uso que estimen pertinente con el trabajo.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año 2014.

_____ _____

Autor: Eduardo Antonio Roque Díaz

Autor: Edel Alejandro Sariol Soto

_____ _____

Tutora: Ing. Aylín Estrada Velazco

Tutor: Ing. Adriam Balceiro Rodríguez

Dedicatoria

A mi mamá, mi papá, mi hermana, a mis tíos y mi primo.

A mi abuelo y mi abuela y en especial a mi bisabuelo, mi muy querido "Sariol".

A mi abuela Hildelisa y a mi abuelo Antonio Soto por enseñarme desde pequeño.

En general a mi familia y a todo el que se ha ganado mi aprecio.

Edel Alejandro.

Dedico este trabajo a mi familia, en especial a mi mamá que ha sido mi guía siempre. A mi primo Ariel Díaz, por apoyarme, por enseñarme a no conformarme y a exigirme más cada día, por nunca rendirse conmigo, por criarme, por su sinceridad, por ser un ejemplo a seguir y sobre todas las cosas por su afecto, gracias de todo corazón.

Eduardo Antonio

Agradecimientos

Quisiera agradecerle a todos los que tuvieron no solo que ver con el desarrollo de esta tesis, sino también aquellos que me hicieron ser como soy ahora.

Para ti mamá que siempre estás sufriendo los dolores de cabeza que te he dado y nunca te has rendido, a ti papá que siempre estuviste para corregirme y mostrarme el mejor camino y a mi hermana por ser mi "pelota". A mi tío y mi tía quienes lograron que me convirtiera en Ingeniero en esta gran UCI y mi primo que siempre me apoyó en todo.

A mis abuelos que a pesar de todo siempre han estado al tanto de mí.

A mi novia Maylen "Mamol" por estar siempre aquí.

A mi amigos más cercanos Daniel (el dispa) con sus consejos sabios, a Luis (War Nasty), Eva (mi mejor amiga) por sus conversaciones y su apoyo en todo. A mi compañero de tesis no solo por ser compañero de tesis sino por ser amigo incondicional no solo conmigo sino con todos. Gracias a todos ustedes hoy soy una mejor persona y seré un mejor profesional.

Gracias.

Edel Alejandro Sariol Soto.

A mi familia, por apoyarme siempre en todas mis buenas y malas decisiones, por su afecto, por nunca rendirse conmigo y tratar siempre de convertirme en un mejor ser humano. Por otra parte, quisiera agradecerle a todos mis amigos porque sé que siempre podré contar con ellos, por ser incondicionales.

Gracias

Eduardo Antonio Roque Díaz.

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 y distribución de los contenidos multimedia en la red cubana, 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 v2.3, jQueryUI v1.9.2 y Bootstrap v3.0.2 y twig como motor de plantillas. El sistema gestor de base de datos PostgreSQL v9.1 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 en aplicaciones externas; además cuenta con una capa de servicios REST la cual permite a los desarrolladores interactuar con la aplicación, utilizando sus contenidos para el desarrollo de sus soluciones.

Palabras clave: colaboración, contenidos multimedia, distribución, publicación, servicios.

Índice

ÍNDICE

Índice de figuras

Índice de tablas

Introducción

Hasta los primeros años de la década del 90, Internet era un conjunto de ordenadores inconexos y no se podía navegar de una dirección a otra pulsando en un enlace (Abuín, 2011). Tampoco existían los buscadores, ni se podían integrar contenidos multimedia en la pantalla, debido a que no se habían desarrollado interfaces gráficas para la web. En menos de veinte años, la web sufrió una evolución tecnológica no experimentada por otro medio en 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.

El Centro de Ideoinformática (CIDI), perteneciente a la Facultad 1 de la Universidad de Ciencias Informáticas (UCI) fue creado con el objetivo de ofrecer a los usuarios soluciones informáticas enfocadas a resolver problemas actuales en Internet. Para dar cumplimiento a los objetivos por los que fue creado dicho centro, se crean diferentes líneas de producción. La línea de producción para el desarrollo de portales web perteneciente al Centro CIDI fue creada con el objetivo de lograr mayor cantidad de soluciones de calidad

1 World Wide Web por sus siglas en inglés.

en el menor tiempo posible. Para alcanzar los objetivos por los que fue creada dicha línea se utiliza el sistema de administración de contenido *Drupal (CMS*, por sus siglas en inglés*)*. Sin embargo, existen problemas que afectan tanto el desarrollo de las soluciones como las aplicaciones desplegadas. Las afectaciones, generalmente se presentan debido a que la cantidad de información y contenidos multimedia que se publican en las aplicaciones web se encuentran en diversas fuentes, lo que produce atrasos durante el desarrollo y afectaciones en las publicaciones de los sistemas desplegados. De manera específica, en el proceso de desarrollo, no se comparten los recursos ni se socializa la información entre los diferentes equipos de trabajo, lo que provoca que se duplique información y contenidos multimedia y produce aumento en el consumo del espacio de almacenamiento. Los sistemas desplegados en la red nacional se encuentran limitados con el uso de contenidos multimedia, específicamente los contenidos de audio y video. Las aplicaciones que actualmente cuentan con ellos obtienen los recursos multimedia de plataformas que se encuentran en el exterior del país, lo que produce que se afecte la cobertura en tiempo real y la inmediatez de las publicaciones debido a la deficiente velocidad de conexión con la que cuenta el país.

Sobre la base de los elementos expuestos anteriormente se ha formulado el siguiente **problema de investigación**: ¿Cómo contribuir a la publicación y distribución de contenidos multimedia para la red cubana?

El **objeto de estudio** se enfoca en la gestión de contenidos multimedia en Internet, como **objetivo general** para darle cumplimiento al problema planteado se propone desarrollar una aplicación web distribuida, que contribuya a la publicación y distribución de contenidos multimedia para la red cubana; para ello se definió como **campo de acción** la gestión de contenidos multimedia para la red cubana. Teniendo en cuenta el problema planteado se propone como **idea a defender** “la implementación de una aplicación web distribuida contribuirá a la publicación y distribución de contenidos multimedia en la red cubana”.

Para dar solución al objetivo general de la investigación se han formulado los siguientes **objetivos específicos** que facilitan el desarrollo del mismo:

- Caracterizar las tendencias actuales de los Sistemas para la publicación y distribución 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 y distribución de contenidos multimedia.
- Diseñar las funcionalidades de la aplicación web, Sistema para la publicación y distribución de contenidos multimedia.
- Implementar las funcionalidades de la aplicación web, Sistema para la publicación y distribución de contenidos multimedia.
- Validar mediante pruebas el Sistema para la publicación y distribución de contenidos multimedia.

Teniendo como premisa resolver los objetivos planteados se proponen las siguientes **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 y distribución de contenidos multimedia.
- Realización del análisis y diseño de la aplicación web, Sistema para la publicación y distribución de contenidos multimedia.
- Validación mediante las pruebas funcionales, de seguridad y rendimiento la aplicación web, Sistema para la publicación y distribución de contenidos multimedia.

En la investigación se utiliza como enfoque general el dialéctico materialista, fueron considerados tanto los aspectos objetivos como subjetivos de la interacción de los diferentes actores en los sistemas para la publicación y distribución de contenidos multimedia. A continuación se exponen los métodos empleados en la investigación clasificados en teóricos y empíricos:

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.

• **Investigación-Acción:** Incluye diagnóstico del problema, intervención de acción y aprendizaje reflexivo. El mismo fue usado durante la investigación, para conocer el modo de implementar una capa de servicios usando REST.

Métodos Empíricos:

• **La entrevista:** Se empleó como medio para la búsqueda de información y la extracción de los requisitos funcionales del sistema. Además de la bibliografía consultada, este método brindó especificaciones de los especialistas del centro sobre las tendencias del desarrollo para Internet en la actualidad.

• **La observación:** Permitió constatar la utilización que tienen algunos sistemas similares que se encuentran en funcionamiento en la UCI, así como las funcionalidades que brindan.

La tesis consta de la presente introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos que ampliarán la información que se aporta en la investigación.

Capítulo 1: **"Fundamentación teórica que sostiene la propuesta de solución":** Contiene los fundamentos teóricos que giran en torno a los sistemas para la publicación y distribución de contenidos multimedia. En el mismo se hace un estudio de sistemas homólogos en cuanto a objetivos de uso, su funcionamiento y tecnologías que utilizan. Además se describen los principales conceptos para comprender el dominio del problema, así como las tendencias, metodologías, tecnologías y herramientas que se emplean en la implementación de la propuesta de solución.

Capítulo 2: "Características y diseño del sistema": En este capítulo se explica cómo se desarrolla el flujo actual de los procesos, y se describe la propuesta de solución para resolver el problema planteado. Por otra parte, se especifican los requisitos funcionales y no funcionales, y los elementos fundamentales del diseño y de la arquitectura que se deben tener en cuenta para el desarrollo del sistema, ayudado por la realización del modelado de diagramas.

Capítulo 3: "Implementación y validación del sistema": En este capítulo se incluye la programación realizada a partir de los requerimientos, así como las pruebas realizadas para su validación. Además se establecen los estándares de codificación que se tuvieron en cuenta para el desarrollo del sistema.

Completan este trabajo un conjunto de conclusiones y recomendaciones, así como los anexos que aportan información valiosa sobre la investigación. Las fuentes bibliográficas utilizadas se encuentran asentadas al final de la investigación.

Capítulo 1: Fundamentación teórica que sostiene la propuesta de solución

Con el surgimiento de la WWW en la década del 90 y el aumento constante de las velocidades de acceso a la red de redes, nuevos métodos de implementación, lenguajes de programación y marcos de trabajo, entre otras tecnologías, han tomado auge en el desarrollo para Internet. Además, alrededor de una aplicación web giran una serie de conceptos que mejoran su comprensión, tanto para su desarrollo como para su posterior uso. Para garantizar la soberanía tecnológica de la propuesta de solución, todas las tecnologías y herramientas que se utilicen durante su desarrollo estarán bajo los preceptos del software libre.

1.1. Generación Multimedia: Tendencias

La sociedad contemporánea se autodefine por su relación con la cultura popular, que es construida por los medios de comunicación y está compuesta por la música, la televisión, el cine, los videojuegos, así como Internet y las redes sociales (Rigo, 2008). Son los consumos culturales de la sociedad, por tanto, los que ayudan a la construcción de la identidad e ideología propia de cada ente social. De aquí la importancia de masificar su distribución y diversificar sus contenidos.

Imma Tubella manifiesta que se va produciendo una transformación y cambio en el consumo, porque los llamados medios tradicionales, especialmente la televisión y la radio, ceden lugar en la vida cotidiana a los nuevos medios como Internet. Se trata de las posibilidades y libertades que ofrece Internet respecto a los medios tradicionales. Socializar[2] y compartir[3] cualquier tipo de información y contenidos multimedia, crear comunidades virtuales, en general, colaborar[4], pueden ser algunas de las principales ventajas presentadas por Internet sobre estos medios. La propuesta de solución pretende contribuir con la distribución de la información y los contenidos multimedia e intenta concretar, mediante la colaboración entre usuarios, las palabras de *Imma Tubella* cuando expresa: *"con la paulatina universalización de las herramientas que permiten la participación activa en los procesos de generación, edición y distribución de información y contenidos, el consumidor pasa a ser también, al mismo tiempo, un creador activo con capacidad para aportar y compartir de múltiples maneras su visión del mundo en el que vive"* (Tubella, 2008). Esta frase no solo motiva al consumo de los contenidos y las herramientas que se colocan a disposición de los usuarios en Internet, sino llama a que todos muestren y compartan su visión del mundo.

2 Promover las condiciones sociales que favorezcan en los seres humanos el desarrollo integral de su persona (RAE, 2014).

3 Repartir, distribuir las cosas en partes para que otro u otros puedan beneficiarse de ello (RAE, 2014).

4 Es ayudar y servir de manera espontánea a los demás (RAE, 2014).

Motivación de la investigación

Con la aplicación de las nuevas tendencias en el desarrollo para Internet, las soluciones propuestas por el Centro CIDI han tenido mejor aceptación entre los clientes, de todas ellas es preciso destacar, debido a que son las más relevantes durante el desarrollo: el uso de imágenes, lo cual ofrece a los usuarios una mejor apariencia de la aplicación; ante el aumento del ancho de banda y el surgimiento de nuevas tecnologías los contenidos audiovisuales se han ganado un puesto importante dentro de las nuevas tendencias y es la simplicidad, o sea, el diseño minimalista, lo que ha dado paso a que imágenes y contenidos audiovisuales tengan mayor presencia en Internet. Siguiendo dichas tendencias, los clientes solicitan cada vez más la presencia de contenidos multimedia en la web (Pastor, 2000). Nuevos problemas surgen a raíz de este enfoque; necesidad de poseer hardware de altas prestaciones y mayor ancho de banda, pero no todos pueden darle respuesta a dichas problemáticas. Sin embargo, servidores distribuidos de medianas prestaciones y la tecnología *streaming* han logrado minimizar las problemáticas antes mencionadas, pero para los usuarios finales sería casi imposible explotar sus ventajas para darle solución a los problemas, sin que un software las implemente en su solución. Esta deficiencia provoca que la presencia en Internet de quienes poseen dichos problemas se vea afectada. Ciertamente esto ocurre en países subdesarrollados y con pocos recursos y ante las limitaciones existentes, la innovación y creación son elementos importantes para minimizar la magnitud de la brecha y seguir adelante con los objetivos planteados. Para un mejor entendimiento de los principales conceptos que fundamentan la propuesta de solución se realizará a continuación la descripción de los mismos.

1.2. 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[5] y los metadatos[6] (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.

5 Información estática (documentos, imágenes), dinámica (un *blog* o un sitio web) o continua (*streaming* de audio y video), por lo que su origen puede ser pregrabado u originario de fuentes vivas, y su existencia puede ser perecedera o perenne dentro del sistema.

6 Descripción del contenido que facilita su identificación, búsqueda, e incluso su gestión si es multimedia, y sobre todo hace más sencilla su interpretación.

- **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.

Los autores de esta investigación señalan que el término **contenido multimedia** se aplicará a cualquier recurso en formato digital que contenga múltiples medios de expresión, físicos o digitales, para presentar o comunicar información. En el Sistema para la publicación y distribución de contenidos multimedia se hará referencia, solamente, a los tipos de contenidos multimedia: imagen, video y sonido.

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*[7]). 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).

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*[8] y Betamax[9], también se incluyen los formatos digitales *DVD*, *MPEG* y *OGV*, Webm, entre otros.

Características de algunos de algunos formatos de video

MPEG-1

7 Superficie homogénea más pequeña de las que componen una imagen, que se define por su brillo y color (RAE, 2014).

8 Siglas de Video Home System, es un sistema de grabación y reproducción analógica de audio y video.

9 Proyector de video para pantallas de gran formato que sugió en 1975.

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*[10] (*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*[11], *DVD*[12], 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*[13]. Está compuesto por el *códec* de video *VP8* (desarrollado originalmente por *On2 Technologies*[14]) y el *códec* de audio Vorbis dentro de un contenedor

10 Disco Compacto (CD por sus siglas en inglés).
11 Formato utilizado para almacenar video en discos compactos estándar.
12 Corresponde a "Digital Versatile Disc", que a su vez es el término que engloba los distintos tipos de DVD.
13 http://www.freebsd.org/doc/es/articles/explaining-bsd/article.html
14 Corporación especializada en desarrollar tecnologías de compresión de video. Desde el 19 de febrero de 2010 es una subsidiaria de Google Inc.

multimedia *Matroska*[15]. 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).

Conclusiones parciales sobre formatos de video a usar en la propuesta de solución

Después de haber realizado la revisión de los diferentes formatos de video y sus características, el equipo de desarrollo ha optado por utilizar, *Theora* y *Webm* como contenedores de video, ya que, usan los formatos libres *VP3* y *VP8* los cuales están orientados especialmente para *streaming*[16] en Internet (wordpress.org, 2009).

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

15 Formato contenedor de estándar abierto, un archivo informático que puede contener una cantidad ilimitada de video, audio, imagen o pistas de subtítulos dentro de un solo archivo (Matroska.org, 2013).

Fue desarrollado por *Apple Inc*. en 1988 basado en el IFF (Interchange File Format) de *Electronic Arts*[17]. Actualmente es muy utilizado en las computadoras *Apple Macintosh*[18] 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*[19]).

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*[20] 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 *códec*s multimedia que también desarrolla *Xiph.org*. Dicho formato se utiliza preferentemente con los *códec*s 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).

Conclusiones parciales sobre formatos de audio a usar en la propuesta de solución

Después de haber realizado la revisión de los diferentes formatos de audio y sus características, el equipo de desarrollo ha optado por utilizar el formato de audio comprimido o con pérdida *Vorbis* debido a que tiene menor *bit rate* por lo que la transferencia a través de Internet es mucho más rápida, además los formatos contenedores de video que se seleccionaron utilizan este formato de audio de forma nativa.

---

17 Empresa estadounidense que se dedica a la creación y publicación de videojuegos, conocida por sus videojuegos relacionados con deportes como el fútbol, béisbol, baloncesto entre otros. http://www.ea.com/

18 Empresa multinacional estadounidense que diseña y produce equipos electrónicos y software. Entre los productos de hardware más conocidos de la empresa se cuenta con equipos Macintosh, el IPod, el IPhone y el IPad.

19 Este tipo de modulación, es uno de los más utilizados de todas las modulaciones de pulsos es, básicamente, el método de conversión de señales analógicas a digitales.

20 Tasa de bits (del inglés ***bit rate***).

1.3. 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):

1. 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)**.**

2. 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).

3. *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 la migración de software en la que está inmerso el país 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	Última versión estable	Sistema Operativo	Licencia	Tipo de media	Tipos de formatos que soporta

VLC media player	VideoLAN[21]	2.1.2 (2013/10)	Linux/Windows/MacOS	GPL v2[22]	Audio/Video	AVI, MP4, Ogg, Webm, entre otros
Flumotion *Streaming* Server	Flumotion[23]	0.10. (2011/10)	Linux/Windows	LGPL[24]	Audio/Video	WMA, Ogg, Theora y Webm
Icecast	Xiph.Org Foundation[25]	2.3.3 (2012/06)	Linux	GPL[26]	Audio/Video	Ogg, Theora, AAC

Tabla 1. *Descripción de los servidores streaming*

Conclusiones parciales sobre servidores *streaming* analizados

Luego de haber analizado los datos recogidos en la Tabla 1, teniendo en cuenta que los servidores estudiados cumplen con los requerimientos señalados, el equipo de trabajo arriba a las siguientes conclusiones: cada uno ayuda con la solución de los problemas por los cuales surgió la investigación y todos son desarrollados bajo los preceptos del software libre. Se decide usar *Flumotion Streaming Server* debido a la compatibilidad de los formatos para los que tiene soporte el lenguaje *HTML5*; presenta grandes ventajas en cómo realizar las configuraciones para su posterior uso; permite el monitoreo y control de los recursos de red asociados al uso del servicio prestado. Además la experiencia de uso que se ha tenido con el mismo ha sido satisfactoria.

1.4. 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.

21 http://www.videolan.org/vlc/
22 http://www.gnu.org/licenses/gpl-2.0.html
23 http://www.flumotion.net/
24 https://www.gnu.org/licenses/lgpl.html
25 http://www.xiph.org/
26 http://www.gnu.org/licenses/gpl.html

- **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 contenido 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*

Conclusiones parciales sobre las aplicaciones analizadas

Las aplicaciones estudiadas anteriormente presentan funcionalidades que pueden ser desarrolladas e incluidas en la propuesta de solución. Sin embargo, dichas aplicaciones no resuelven las problemáticas por las que surgió la presente investigación debido a que:

Las aplicaciones como *YouTube, Instagram, Flickr* y *Vimeo* cuentan con el patrocinio de grandes empresas, lo cual posibilita de cierta manera que el espacio de almacenamiento y el potencial de procesamiento utilizado ascienda constantemente, o sea, no tienen limitaciones de hardware. El ancho de banda con que cuentan los países en los cuales se encuentran implantados estas aplicaciones crece constantemente y Cuba apenas se encuentra en fase de implantación y expansión de las infraestructuras de transmisión de datos de alta velocidad (ETECSA, 2013). Acceder a estas aplicaciones se convierte en una tarea compleja, debido a la deficiente velocidad de acceso existente en el país.

La aplicación del ICRT[27], teVeo, no cumple totalmente con los requerimientos necesarios para darle solución a los problemas planteados debido a que solo maneja contenidos multimedia de tipo audio y video. No presenta funcionalidades que permitan la socializacion de la información, como votar y sugerir contenidos. La interfaz de usuario no es amigable, lo que provoca poca afluencia de usuarios a la aplicación.

La aplicación desarrollada y desplegada en la UCI, Internos, cuenta con una interfaz de usuario agradable y fácil de usar. Presenta un grupo de funcionalidades y servicios que pueden ser desarrollados e incluidos

27 Instituto Cubano de Radio y Televisión.

en la propuesta de solución. Sin embargo no permite crear ni solicitar cuentas de usuario, lo cual funciona como mecanismo para impedir la duplicación de los contenidos. Tampoco permite compartir contenidos multimedia frenando la socialización de la información. Además solo maneja contenidos multimedia de tipo audio y video.

1.5. Metodología de desarrollo que guiará la propuesta de solución

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 beneficio 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*[28] 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.

28 *Stakeholder*: Persona que es significativamente afectada por el resultado del producto. Representa los intereses cuyas necesidades deben ser satisfechas por el proyecto.

- **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.

1.6. 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, el equipo de trabajo hará uso de herramientas y lenguajes que serán de vital importancia para el desarrollo de la solución, las cuales se describen a continuación.

1.7. Entorno de Desarrollo Integrado (IDE)

Un entorno de desarrollo integrado, llamado también (IDE, por sus sigla 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.

NetBeans v7.3

NetBeans es un entorno de desarrollo integrado disponible para Windows, Mac, Linux y Solaris. Consiste en un IDE de código abierto y una plataforma de aplicaciones que permiten a los desarrolladores crear aplicaciones de escritorio, web y móviles, soporta varios lenguajes, entre ellos: PHP, JavaScript, C++, entre otros. Puede ser integrado fácilmente con Symfony2. Está apoyado por una comunidad de desarrolladores y, ofrece una amplia documentación y recursos de capacitación, así como una gran cantidad de plugins de terceros (Oracle, 2013).

1.8. Sistema Gestor de Base 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 v9.1

PostgreSQL es un sistema de gestión de base de datos relacional, distribuido bajo licencia BSD[29] 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[30].
- 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.

Entre los límites de PostgreSQL se encuentran:

Límite	**Valor**
Máximo tamaño base de datos	Ilimitado (Depende del sistema de almacenamiento)
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 (dependiendo del tipo de columna)
Máximo número de índices por tabla	Ilimitado

29 En español, Distribución de Software Berkeley. Se utiliza para identificar un Sistema operativo que deriva del sistema Unix nacido a partir de los aportes realizados a ese sistema por la Universidad de California en Berkeley.

30 En bases de datos se denomina **ACID** a un conjunto de características necesarias para que una serie de instrucciones puedan ser consideradas como una transacción. En concreto **ACID** es un acrónimo de Atomicidad, Consistencia, Aislamiento y Durabilidad.

Tabla 3. *Límites de PostgreSQL*

1.9. Lenguajes de desarrollo y marcos de trabajo

Durante la implementación de la propuesta de solución se hará uso de varios lenguajes de desarrollo y marcos de trabajo (*framework*) que garantizarán el desarrollo de la propuesta de solución.

Lenguajes de desarrollo para la web

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.

PHP v5.4.0

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). *HTML*5 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 (Eguíluz, 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 software 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 v2.3

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 (Eguíluz, 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*[31], 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 v3.0.2

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).

31 Document Objet Model es la estructura de objetos que genera el navegador cuando se carga un documento (Álvarez, 2008).

Tecnologías para garantizar la interoperabilidad de la solución

REST

La Transferencia de Estado Representacional (REST por sus siglas en inglés) es una técnica de arquitectura software para sistemas hipermedia distribuidos como la *World Wide Web*. REST permite crear servicios y aplicaciones que pueden ser usadas por cualquier dispositivo o cliente que entienda HTTP. Los sistemas que siguen los principios *REST* se llaman con frecuencia *RESTFul*. *REST* afirma que la web ha disfrutado de escalabilidad como resultado de una serie de diseños clave (Fielding, 2000).

- **Un protocolo cliente-servidor sin estado:** cada mensaje *HTTP* contiene toda la información necesaria para comprender la petición. Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes. Sin embargo, en la práctica, muchas aplicaciones basadas en *HTTP* utilizan *cookies* y otros mecanismos para mantener el estado de la sesión (algunas de estas prácticas, como la reescritura de *URL*, no son permitidas por *REST*)
- **Un conjunto de operaciones bien definidas que se aplican a todos los *recursos* de información:** *HTTP* en sí define un conjunto pequeño de operaciones, las más importantes son *POST, GET, PUT y DELETE*. Con frecuencia estas operaciones se equiparan a las operaciones *CRUD* en bases de datos (ABMC en castellano: Alta, Baja, Modificación y Consulta) que se requieren para la persistencia de datos, aunque *POST* no encaja exactamente en este esquema.
- **Una sintaxis universal para identificar los recursos**. En un sistema *REST*, cada recurso es direccionable únicamente a través de su *URI*.
- **El uso de hipermedias**, tanto para la información de la aplicación como para las transiciones de estado de la aplicación: la representación de este estado en un sistema *REST* son típicamente *HTML* o *XML*. Como resultado de esto, es posible navegar de un recurso *REST* a muchos otros, simplemente siguiendo enlaces sin requerir el uso de registros u otra infraestructura adicional.

Guzzle

Guzzle es una librería que facilita el envío de peticiones *HTTP*, incluyendo las herramientas necesarias para crear un cliente de servicios web, utilizando todo el poder de *cURL*[32]. Trabajando con una aplicación Symfony2 y utilizando la inyección de dependencias es muy fácil la integración y el uso de *Guzzle*.

32 cURL es una herramienta para usar en un intérprete de comandos para transferir archivos con sintaxis URL, soporta FTP, FTPS, HTTP, HTTPS, TFTP, SCP, SFTP, Telnet, DICT, FILE y LDAP.

1.10. Conclusiones parciales

- 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 *Flumotion Streaming Server* será el utilizado para la reproducción de contenidos multimedia en Internet.
- 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, las cuales fueron escogidas por el cliente y el equipo de trabajo. Por tanto, se utilizará como *framework* de desarrollo Symfony v2.3, jQuery v1.9 y Bootstrap v3.0.2; como lenguajes para la web PHP v5.4.0, HTML 5, CSS 3 y JavaScript; como SGBD PostgreSQL v9.1; como herramienta CASE para el modelado del Software Visual Paradigm v8.0 y como metodología de desarrollo OpenUP. Para apoyar el proceso de implementación fue escogido el IDE de desarrollo NetBeans v7.3.

Capítulo 2: Características y diseño del sistema

La parte más difícil en la construcción de sistemas software es decidir precisamente qué construir. Ninguna otra parte del trabajo conceptual es tan ardua como establecer los requerimientos técnicos detallados, incluyendo todas las interfaces con humanos, máquinas y otros sistemas de software. Ninguna otra parte del trabajo puede perjudicar tanto el resultado final si se realiza de forma errónea. Ninguna otra parte es tan difícil de rectificar posteriormente (Brooks, 1995).

El desarrollo de software es un proceso iterativo incremental (Pressman, 2002), por lo que en el presente capítulo se dirigen los esfuerzos hacia la ejecución de tareas de modelado que guían al equipo de desarrollo hacia una completa especificación de los requisitos y hacia una representación más concreta de la solución a desarrollar.

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.

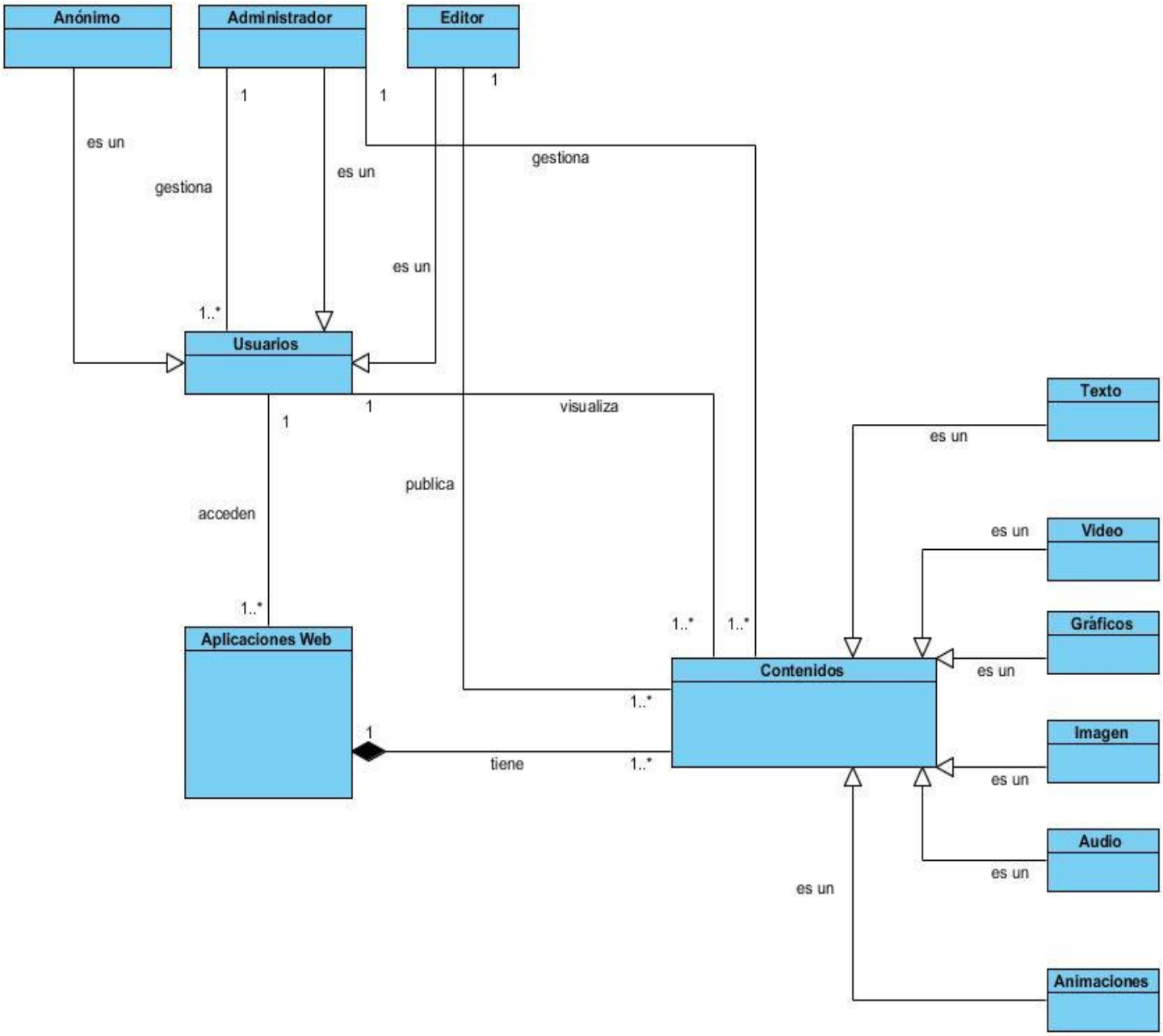


Figura 1. *Modelo de Dominio*

Descripción de las clases del modelo de dominio

Aplicaciones Web: son los sitios web que muestran los diferentes tipos de contenidos.

Usuarios: son las diferentes personas que interactúan con las aplicaciones web.

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

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

Administrador: usuario que navega por las aplicaciones 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 aplicaciones 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 aplicaciones web.

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

Gráficos: tipo de contenido que muestra una representación de datos, generalmente numéricos, mediante recursos gráficos, para que se manifieste visualmente la relación matemática o correlación estadística que guardan entre sí.

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

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

Animaciones: tipo de contenido que da la sensación de movimiento a imágenes, dibujos u otro tipo de objetos inanimados; las animaciones se pueden realizar mediante la utilización de CSS 3 o Flash.

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.

Requisitos funcionales	
Usuarios	
RF_1.	Autenticar usuario
RF_2.	Gestionar usuario
RF_3.	Solicitar cuenta de usuario
RF_4.	Deshabilitar cuenta de usuario
RF_5.	Recuperar contraseña
RF_6.	Eliminar solicitud de contraseña caducada
Perfil de usuario	
RF_7.	Visualizar perfil del usuario
RF_8.	Seguir\ Dejar de seguir usuario
RF_9.	Visualizar usuarios seguidores y seguidos
RF_10.	Visualizar espacio de trabajo
RF_11.	Visualizar preferencias
RF_12.	Editar preferencias
RF_13.	Visualizar recursos compartidos
RF_14.	Eliminar recursos compartidos
Contenidos multimedia	
RF_15.	Gestionar contenidos multimedia
RF_16.	Descargar contenidos multimedia
RF_17.	Dar Me gusta a los contenidos multimedia
RF_18.	Seguir\ Dejar de seguir contenidos multimedia
RF_19.	Visualizar cantidad de seguidores
RF_20.	Compartir contenido multimedia
Comentarios	
RF_21.	Gestionar comentarios
Búsqueda	

RF_22. Realizar búsqueda simple
RF_23. Realizar búsqueda avanzada
Comunes
RF_24. Visualizar ayuda online
RF_25. Visualizar contáctenos
RF_26. Visualizar términos de servicio
Denuncias
RF_27. Denunciar contenido
RF_28. Aceptar denuncia
RF_29. Eliminar denuncia
Etiquetas y Sugerencias
RF_30. Gestionar etiquetas
RF_31. Gestionar sugerencia
Servidores
RF_32. Configuración avanzada del sistema
Interoperabilidad
RF_33. Publicar contenido
RF_34. Capa de servicios web
RF_35. Visualizar gráfica

Tabla 4. *Requisitos funcionales del sistema*

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.

Requisitos no funcionales

Usabilidad	
RNF_US1.	El sistema podrá ser usado por cualquier usuario poseedor de conocimientos básicos en el manejo de la computadora y de aplicaciones web.
Seguridad	
RNF_SE1.	Controlar los permisos de acceso, escritura, lectura en dependencia del rol que desempeñe cada usuario del sistema.
RNF_SE2.	El sistema permitirá que un usuario solamente esté autenticado una vez, es decir, que posea sesión única.
RNF_SE3.	El sistema cerrará la sesión de un usuario inactivo después de un determinado tiempo.
Rendimiento	
RNF_1.	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	
RNF_2.	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 o medios en computación.
Software	
RNF_3.	Usar preferentemente para los servidores el sistema operativo Ubuntu Server 12.04 LTS[33].
RNF_4.	El sistema debe desarrollarse usando la tecnología Symfony2 V2.3 LTS, utilizando la arquitectura Modelo Vista Controlador.
RNF_5.	El lenguaje de programación a utilizar será PHP V5.4.
RNF_6.	Como gestor de base de datos se utilizará PostgreSQL V9.1
RNF_7.	El servidor de aplicaciones será Apache 2.
RNF_8.	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	
RNF_9.	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 29, y Chromium 32.
RNF_10.	Las computadoras clientes deben tener un navegador web (recomendado Mozilla Firefox 29).
RNF_11.	La implementación del producto se hará usando herramientas *Open Source*, sobre la filosofía de Software Libre.
RNF_12.	Solo se procesarán imágenes con los siguientes formatos: JPG, PNG y GIF.
RNF_13.	Solo se procesará audio y videos con los siguientes formatos: webm, ogg, ogx y ogv.

33 Long Time Support (LTS, por sus siglas en inglés).

Hardware	
RNF_14.	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 3TB de espacio de almacenamiento.
Ayuda y documentación online	
RNF_15.	Debe documentarse cómo utilizar el sistema a través de un manual de usuario del producto final.

Tabla 5. *Requisitos no funcionales del sistema*

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

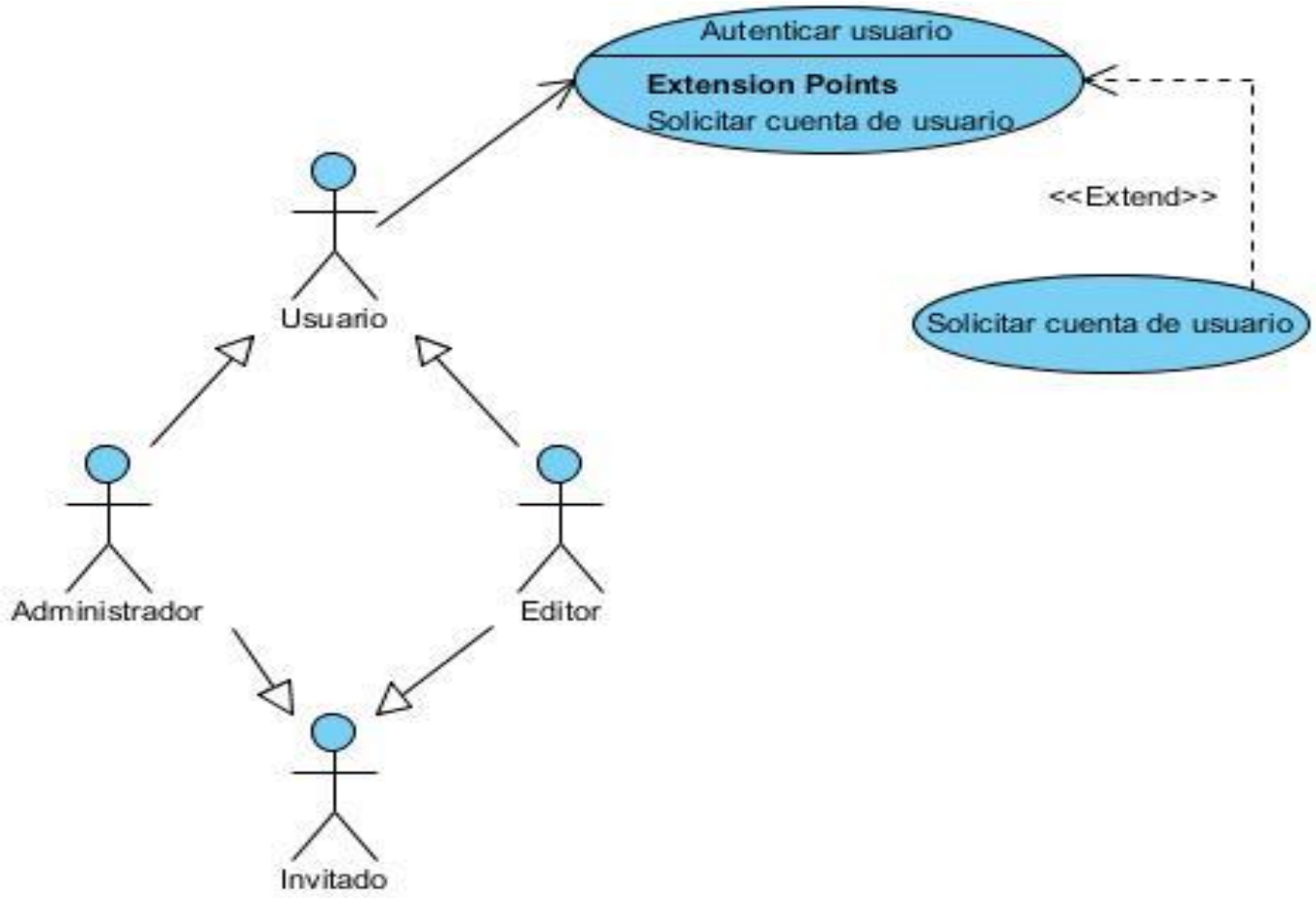


Figura 2. *Diagrama de caso de uso del sistema, por usuario*

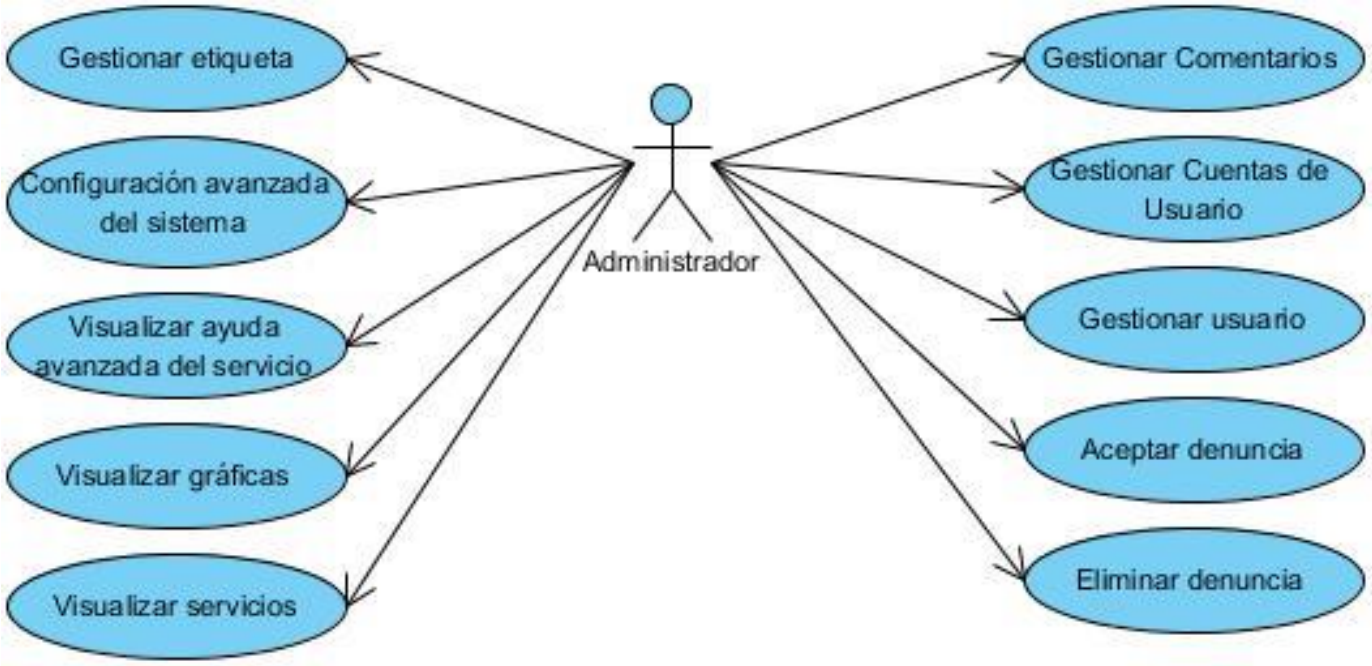


Figura 3. *Diagrama de caso de uso del sistema, por el rol "Administrador"*

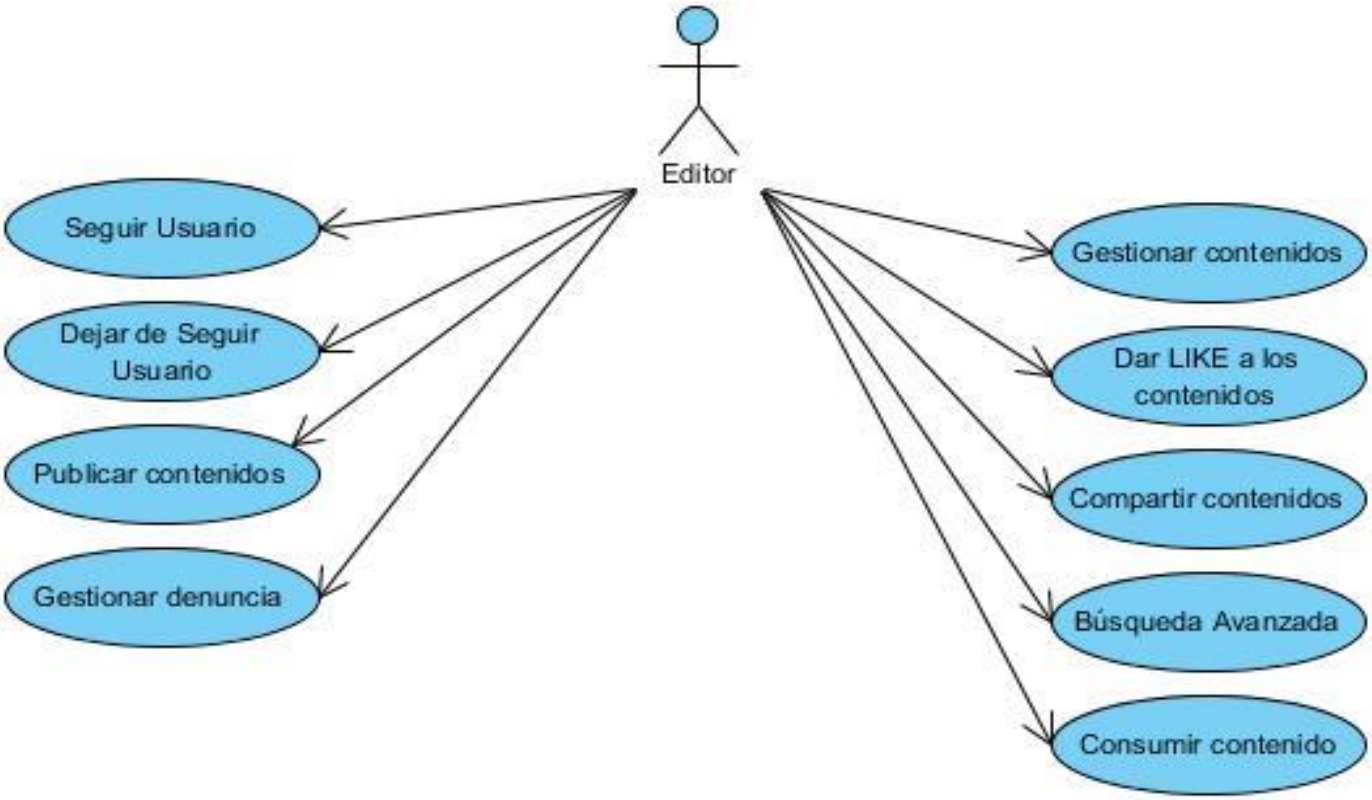


Figura 4. *Diagrama de caso de uso del sistema, por rol "Editor"*

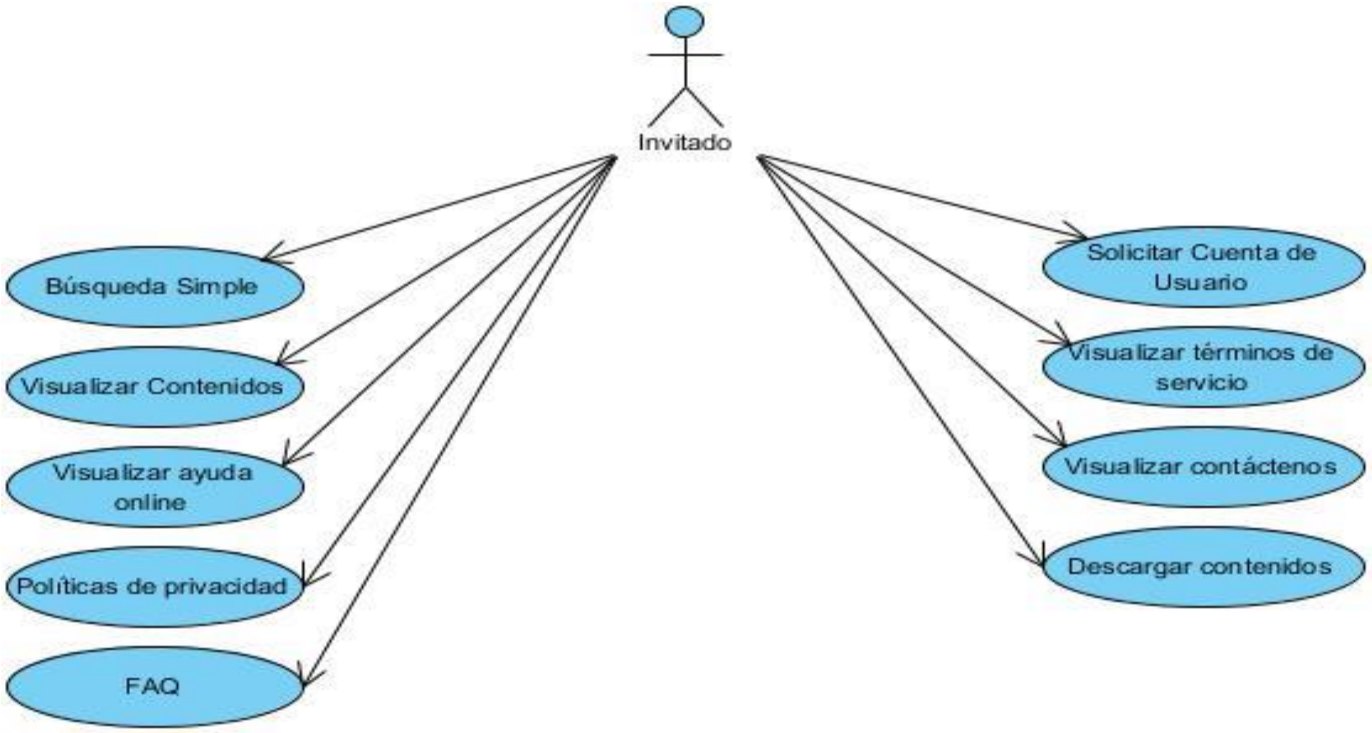
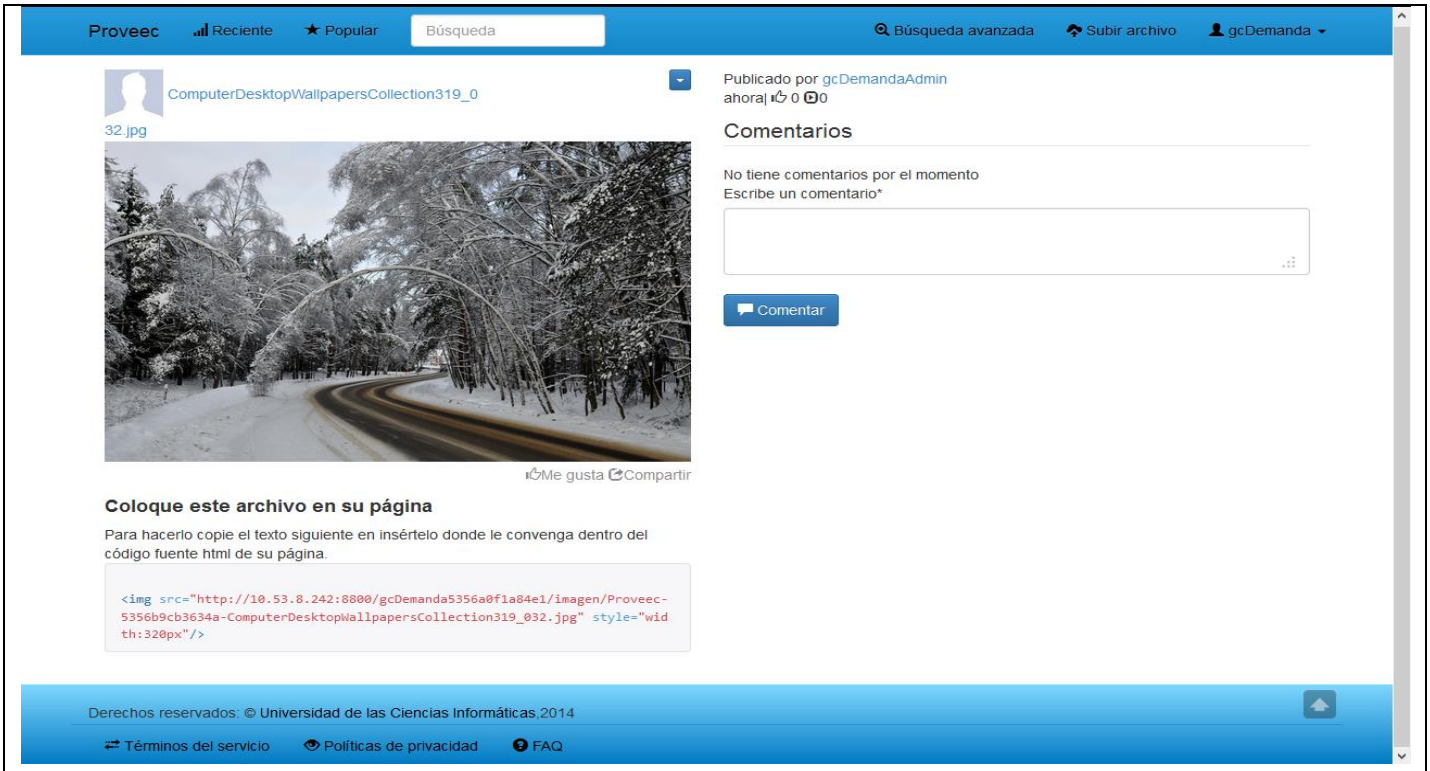


Figura 5. *Diagrama de caso de uso del sistema, por rol "Invitado"*

Con el objetivo de comprender mejor la relación actor-caso de uso y cómo, de manera conjunta estos componen el modelo de caso de uso. Se describe a continuación uno de los principales caso de uso del sistema.

Objetivo	Publicar contenido multimedia en una aplicación web externa.
Actores	Editor: (Inicia) Autenticándose en el Sistema para la publicación y distribución de contenidos multimedia.
Resumen	El CU permite a los usuarios con el rol autorizado a, publicar un contenido multimedia en una aplicación web externa al sistema.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	El usuario debe estar autenticado en el sistema.
Postcondiciones	El contenido multimedia debe estar disponible en el servidor. El servidor *streaming* debe estar disponible.
Prototipo	



Flujo de eventos		
Flujo básico “Publicar contenido”		
	Actor	Sistema
1.	Selecciona el contenido que desea publicar en una aplicación web externa.	Genera un código html y se lo muestra al usuario en una nueva vista donde también se muestra el contenido seleccionado.
2.	Copia la dirección brindada por el sistema y la copia en el lugar donde desee publicar dicho contenido.	El servidor reproducirá el contenido en la aplicación web donde se haya referenciado.
Flujos alternos		
1. El servidor *streaming* no se encuentra disponible.		
	Actor	Sistema
1.		Muestra un mensaje “El servidor *streaming* no se encuentra disponible”.
2.		Termina el caso de uso.

Tabla 6. *Descripción del caso de uso "Publicar contenido"*

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 por el equipo de desarrollo 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.

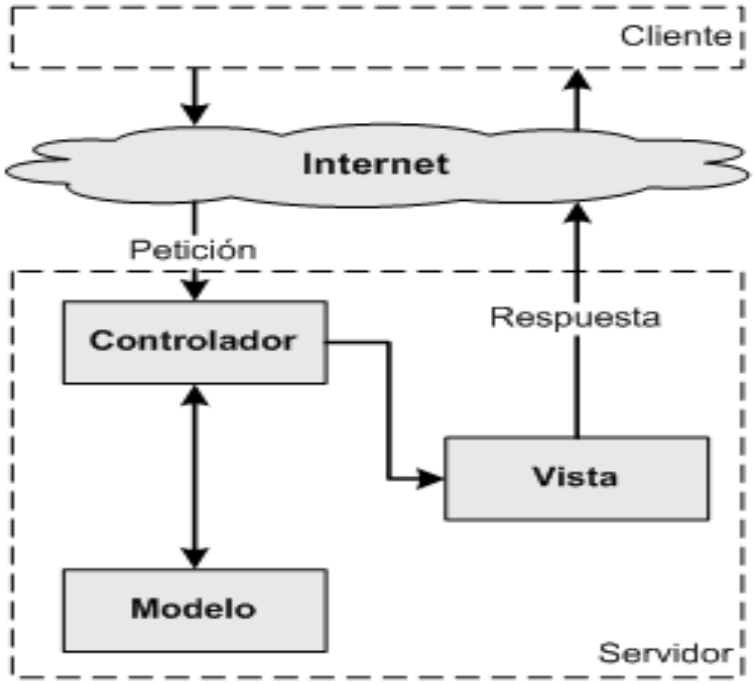


Figura 6. *Representación del patrón MVC*

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 utlización 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 deterrminada 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 concecuencia 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.

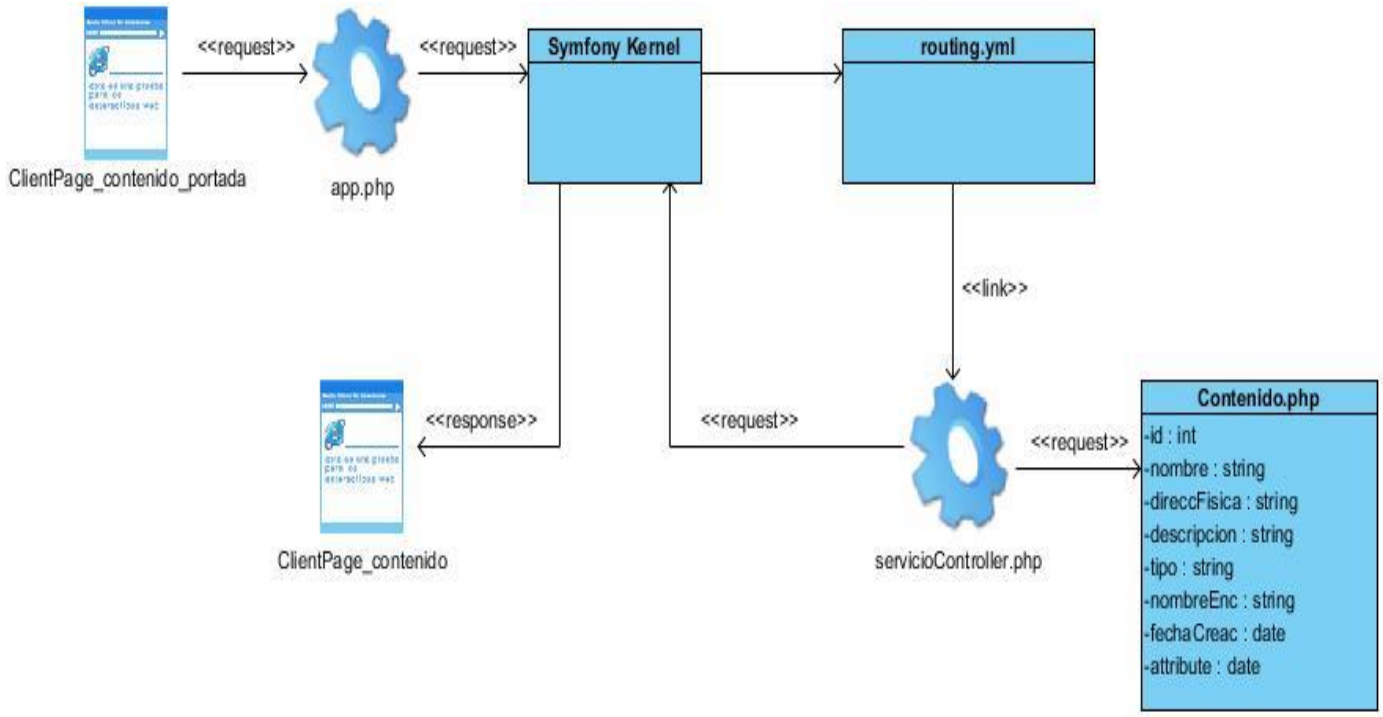


Figura 7. *Diagrama de Clase del Diseño CU "Consumir contenido multimedia"*

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.

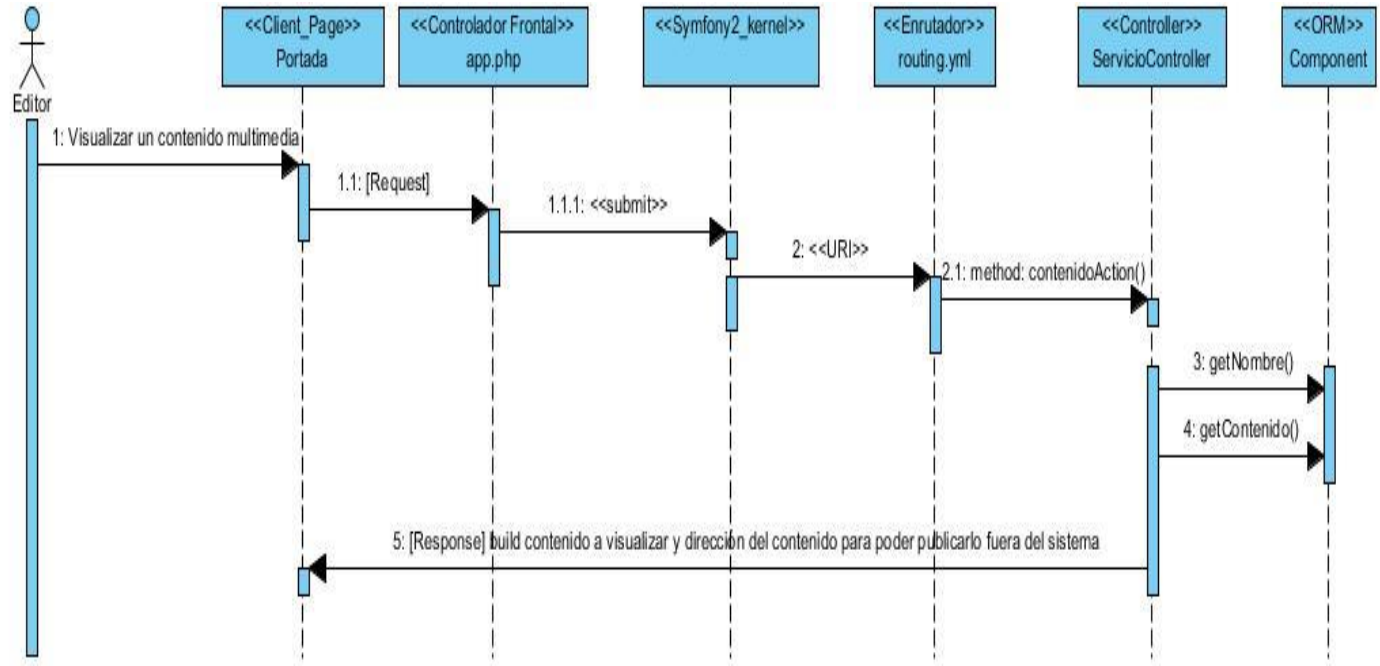


Figura 8. *Diagrama de secuencia del CU "Publicar Contenido"*

2.7. Conclusiones parciales

- El levantamiento de los requisistos 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*.

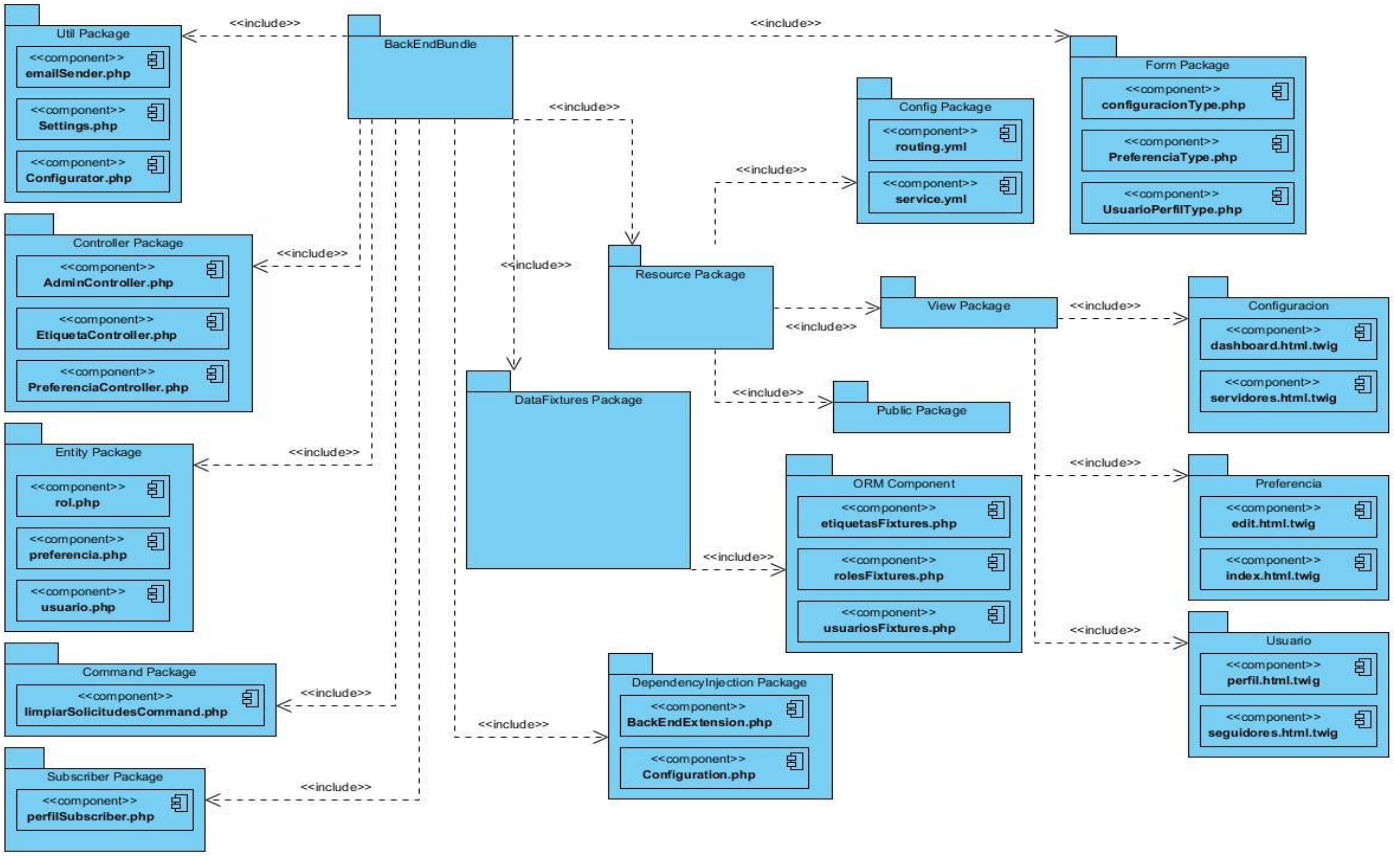


Figura 9. *Diagrama del BackEndBundle*

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.

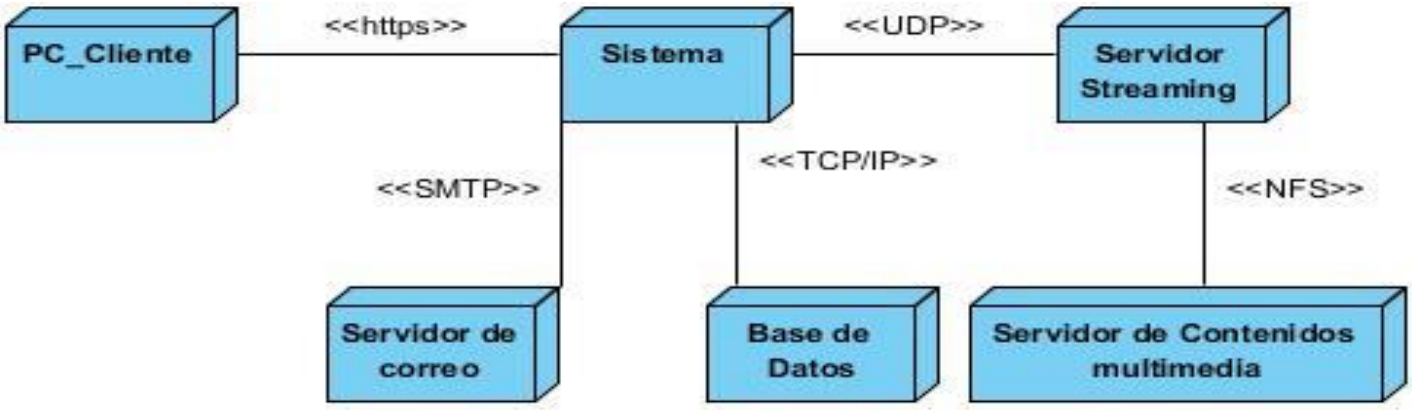


Figura 10. *Diagrama de despliegue*

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[34].
- **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.

34 Sistema de archivos en la red (NFS, por sus siglas en Inglés).

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.

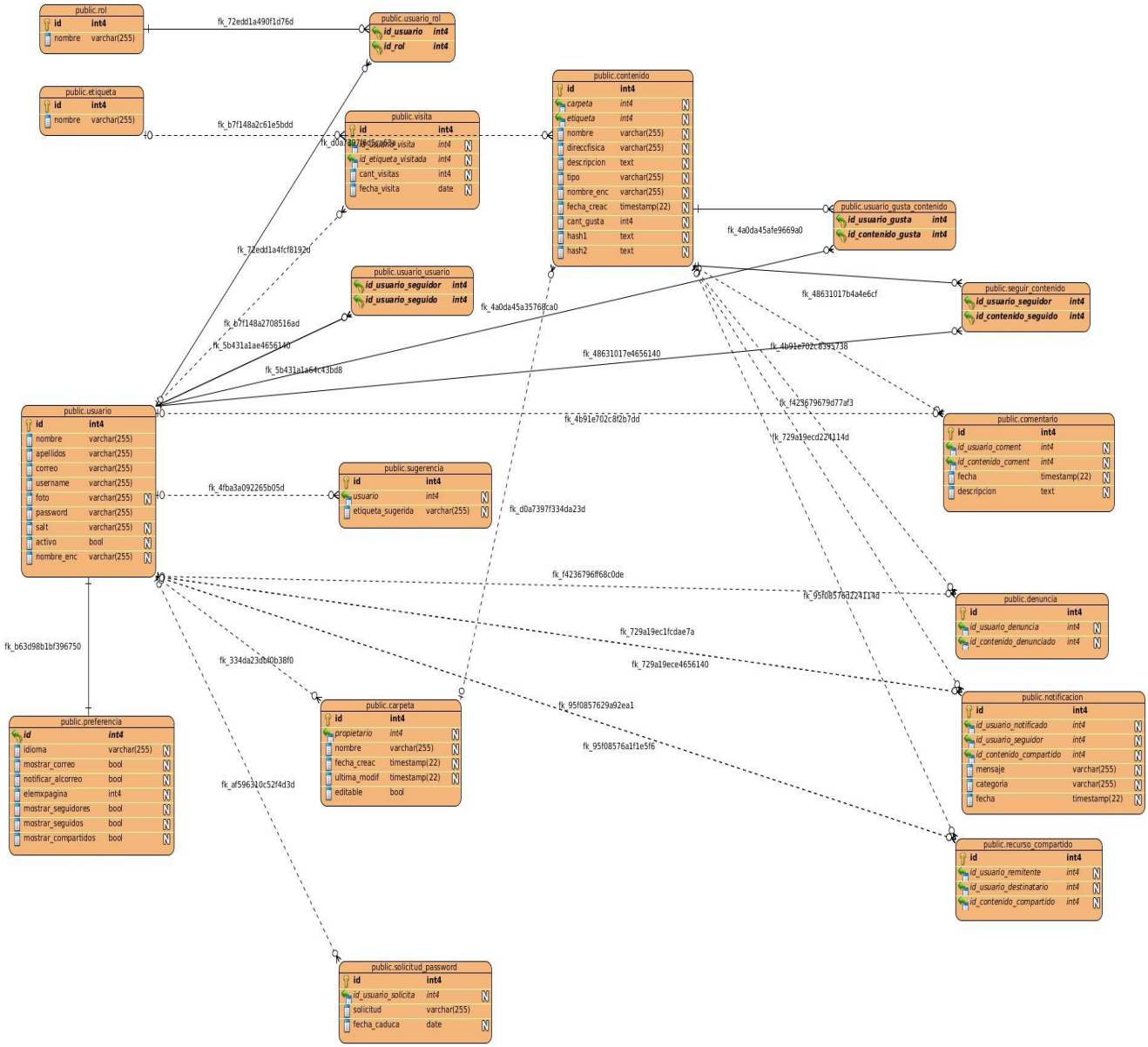


Figura 11. *Modelo físico de la base de datos del sistema*

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[35], PHP, Perl[36], 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 (<?).

35 *Java* lenguaje de programación simple, orientado a objetos, distribuido, interpretado, robusto, seguro, de arquitectura neutra, portable, de altas prestaciones, multitarea y dinámico que se ejecuta sobre máquina hipotética o virtual, ***Java Virtual Machine*** (García, 2000).

36 Perl es un acrónimo de *Practical Extracting and Reporting Languaje*, lenguaje de programación muy práctico para extraer información de archivos de texto y generar informes a partir del contendido de los ficheros (Álvarez, 2001).

- No finalizar las clases con la etiqueta de cierre (*?>*).
- La identación se realiza utilizando cuatro espacios (las tabulaciones no están permitidas).
- 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.

3.5. 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:

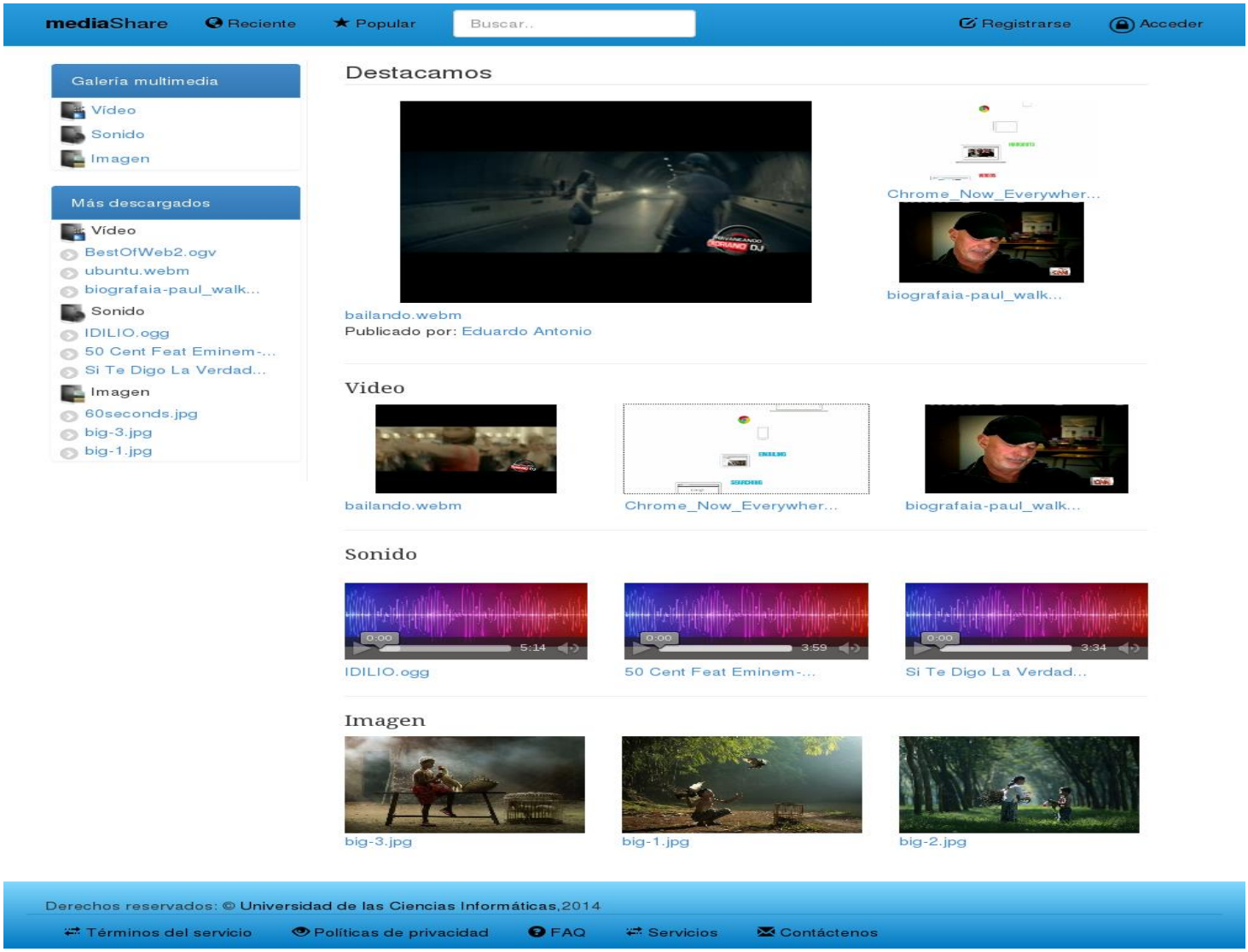


Figura 12. *Pantalla principal del sistema*

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.

3.6. 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://proveec.uci.cu/.
- 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.

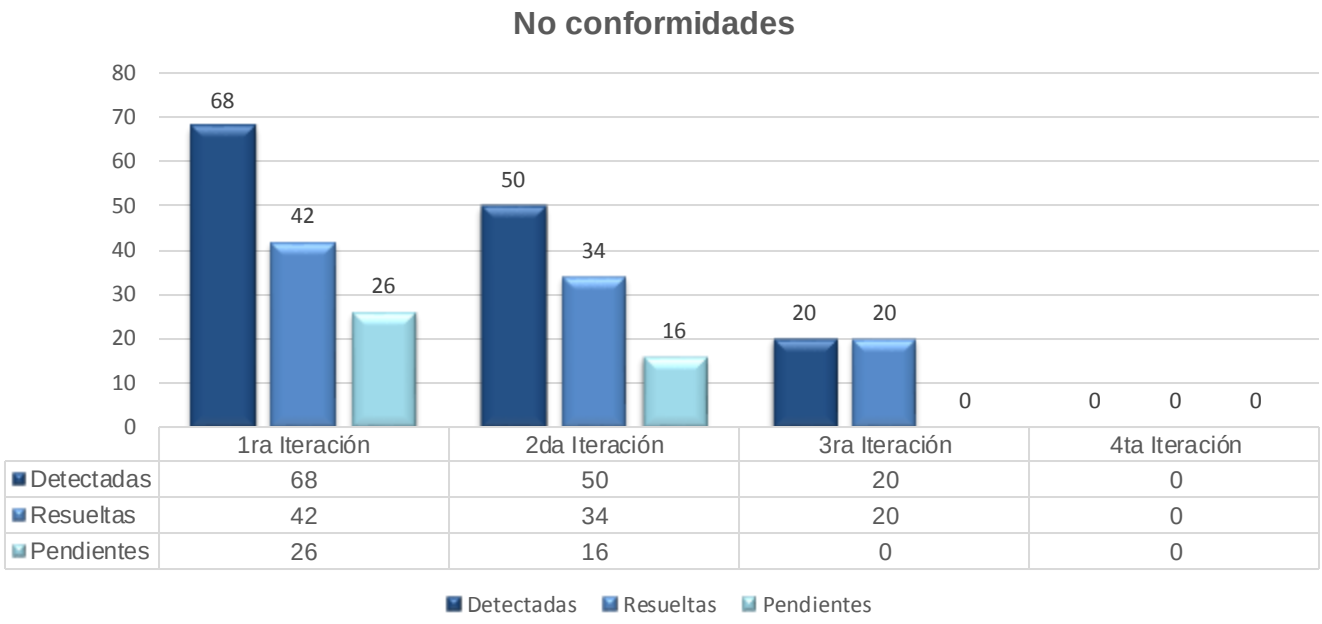


Figura 13. *Comportamiento de las no conformidades por iteraciones*

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.

Los especialistas del grupo de Seguridad del DEPSW[37] de la UCI 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

37 Departamento de software.

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 cliente):

- Tipo de procesador: *Intel (R)* Core(TM)2 Duo CPU T5800 @ 2.00Hz (2CPUs)
- Memoria RAM: 2,00 GB
- Tipo de Red: Ethernet 10/100Mbps

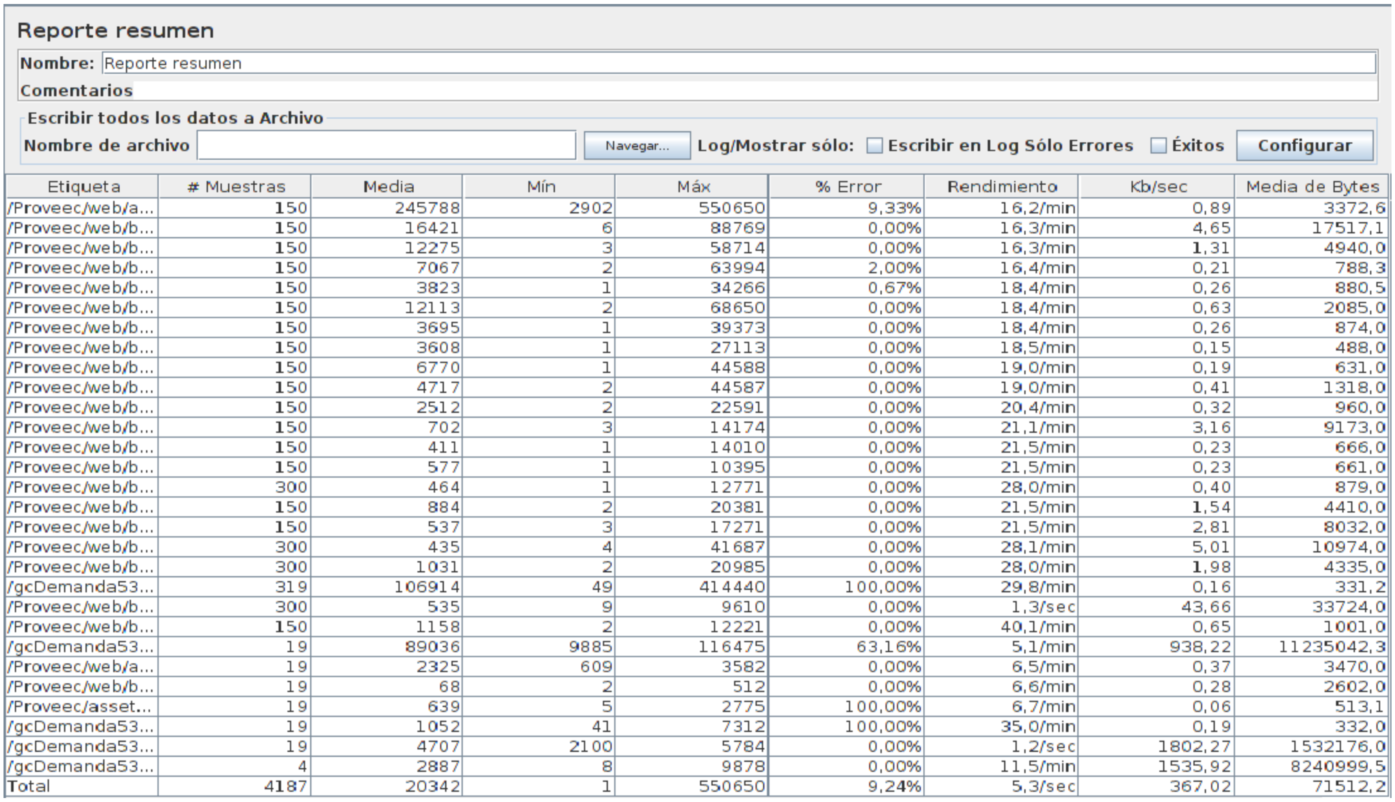
Hardware de prueba (PC servidor):

- Tipo de procesador: *Intel (R)* Core(TM) i3-2100 CPU @ 3.10Hz
- Memoria RAM: 2,00 GB
- Tipo de Red: Ethernet 10/100Mbps

Software instalado en ambas PC:

- Tipo de servidor web: Apache/2.2.22
- Memoria máxima: 1024 MB
- Máximo de hilos concurrentes: 150
- Plataforma: SO Ubuntu 12.04.2 LTS (precise)
- Servidor de BD: PostgreSQL v9.1

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 | Configurar

Etiqueta	# Muestras	Media	Mín	Máx	% Error	Rendimiento	Kb/sec	Media de Bytes
/Proveec/web/a...	150	245788	2902	550650	9,33%	16,2/min	0,89	3372,6
/Proveec/web/b...	150	16421	6	88769	0,00%	16,3/min	4,65	17517,1
/Proveec/web/b...	150	12275	3	58714	0,00%	16,3/min	1,31	4940,0
/Proveec/web/b...	150	7067	2	63994	2,00%	16,4/min	0,21	788,3
/Proveec/web/b...	150	3823	1	34266	0,67%	18,4/min	0,26	880,5
/Proveec/web/b...	150	12113	2	68650	0,00%	18,4/min	0,63	2085,0
/Proveec/web/b...	150	3695	1	39373	0,00%	18,4/min	0,26	874,0
/Proveec/web/b...	150	3608	1	27113	0,00%	18,5/min	0,15	488,0
/Proveec/web/b...	150	6770	1	44588	0,00%	19,0/min	0,19	631,0
/Proveec/web/b...	150	4717	2	44587	0,00%	19,0/min	0,41	1318,0
/Proveec/web/b...	150	2512	2	22591	0,00%	20,4/min	0,32	960,0
/Proveec/web/b...	150	702	3	14174	0,00%	21,1/min	3,16	9173,0
/Proveec/web/b...	150	411	1	14010	0,00%	21,5/min	0,23	666,0
/Proveec/web/b...	150	577	1	10395	0,00%	21,5/min	0,23	661,0
/Proveec/web/b...	300	464	1	12771	0,00%	28,0/min	0,40	879,0
/Proveec/web/b...	150	884	2	20381	0,00%	21,5/min	1,54	4410,0
/Proveec/web/b...	150	537	3	17271	0,00%	21,5/min	2,81	8032,0
/Proveec/web/b...	300	435	4	41687	0,00%	28,1/min	5,01	10974,0
/Proveec/web/b...	300	1031	2	20985	0,00%	28,0/min	1,98	4335,0
/gcDemanda53...	319	106914	49	414440	100,00%	29,8/min	0,16	331,2
/Proveec/web/b...	300	535	9	9610	0,00%	1,3/sec	43,66	33724,0
/Proveec/web/b...	150	1158	2	12221	0,00%	40,1/min	0,65	1001,0
/gcDemanda53...	19	89036	9885	116475	63,16%	5,1/min	938,22	11235042,3
/Proveec/web/a...	19	2325	609	3582	0,00%	6,5/min	0,37	3470,0
/Proveec/web/b...	19	68	2	512	0,00%	6,6/min	0,28	2602,0
/Proveec/asset...	19	639	5	2775	100,00%	6,7/min	0,06	513,1
/gcDemanda53...	19	1052	41	7312	100,00%	35,0/min	0,19	332,0
/gcDemanda53...	19	4707	2100	5784	0,00%	1,2/sec	1802,27	1532176,0
/gcDemanda53...	4	2887	8	9878	0,00%	11,5/min	1535,92	8240999,5
Total	4187	20342	1	550650	9,24%	5,3/sec	367,02	71512,2

Figura 14. *Resultados de la prueba de rendimiento*

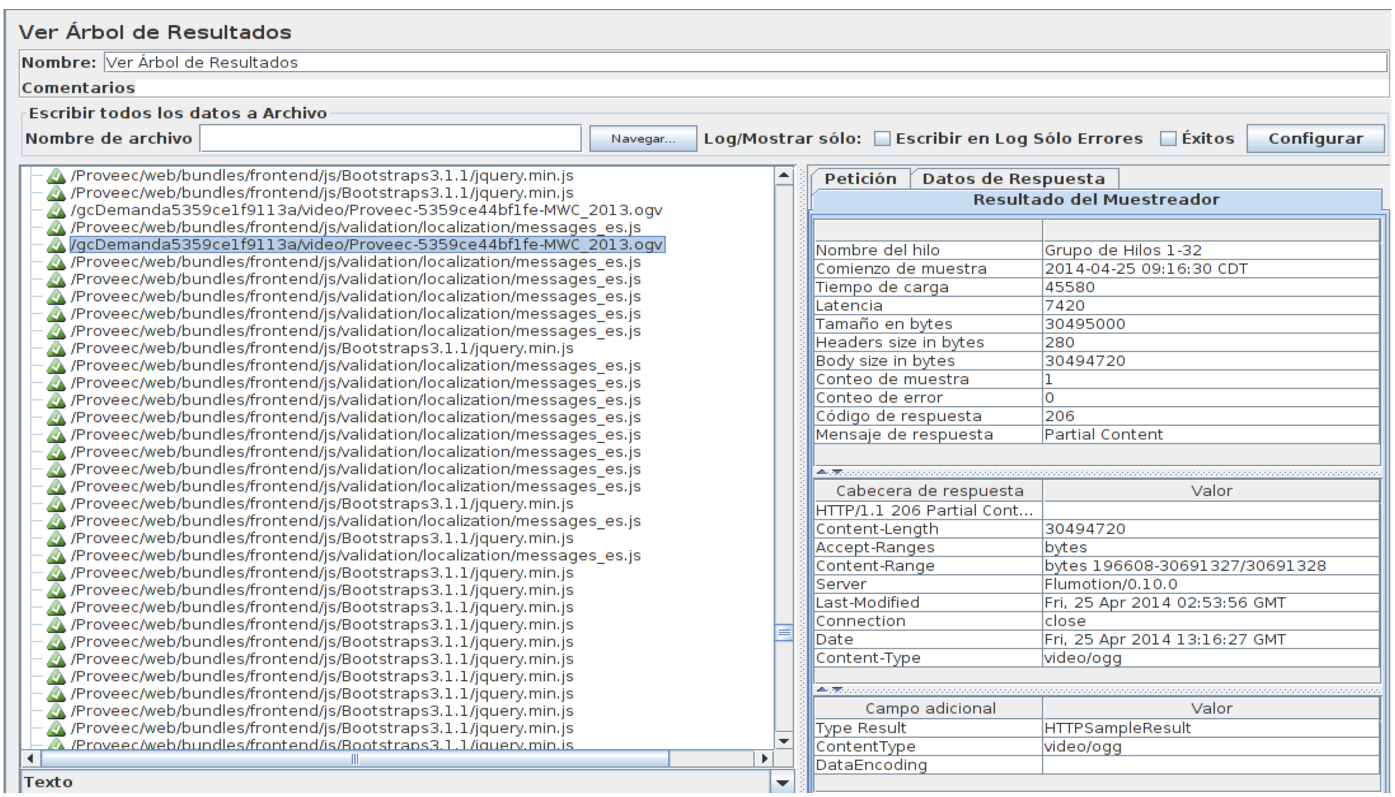


Figura 15. *Resultados de las peticiones realizadas a la aplicación y sus respuestas*

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 un 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.

3.7. Conclusiones parciales

- 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

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): Fecha de consulta: 28 de noviembre de 2013] Disponible en: http://www.redalyc.org/articulo.oa?id=199518706065 ISSN 1605-4806

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): [Fecha de consulta: 28 de abril de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=199520798007 ISSN 1605-4806

Alonso, Jaime. 2005. Comunicar en Internet: el papel interactivo de los sujetos en los nuevos mediosOpción [en línea] 2005, 21 (diciembre): [Fecha de consulta: 8 de marzo de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=31004803 ISSN 1012-1587

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

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

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

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

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

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

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): [Fecha de consulta: 20 de enero de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=31601407 ISSN 1405-2210

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

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

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.

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

Cores Prado, Fernando. 2003. Arquitecturas Distribuidas para Sistemas de Video-bajo-Demanda a gran escala.

Doctor en ciencias. Departament d´Informática. Universitat Autónoma de Barcelona, 2003.

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

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

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

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

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/

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

Fleischman, Luciana, Ginesta, Xavier, López Calzada, Miguel. 2009. LOS MEDIOS ALTERNATIVOS E INTERNET: UN ANÁLISIS CUALITATIVO DEL SISTEMA MEDIÁTICO ESPAÑO LAndamios. Revista de Investigación Social [en línea] 2009, 6 (Agosto-Sin mes): [Fecha de consulta: 28 de enero de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=62812720011 ISSN 1870-0063

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

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/

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.

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): [Fecha de consulta: 22 de abril de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=179014344008 ISSN 0120-0976

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

Islas, Octavio. 2008. INTERNET EN 2008 Razón y Palabra [en línea] 2009, 14 (Marzo-Abril): [Fecha de consulta: 28 de febrero de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=199520725021 ISSN 1605-4806

Jódar Marín, Juan Ángel. 2010. LA ERA DIGITAL: NUEVOS MEDIOS, NUEVOS USUARIOS Y NUEVOS PROFESIONALES Razón y Palabra [en línea] 2010, 15 (Febrero-Abril): [Fecha de consulta: 28 de noviembre de 2013.] Disponible en: http://redalyc.org/articulo.oa?id=199514914045 ISSN 1605-4806

Larrondo Ureta, Ainara. 2005. Reseña de "Ciberperiodismo"Sphera Pública [en línea] 2005, [Fecha de consulta: 20 de enero de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=29700521 ISSN 1180-9210

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

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/

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

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

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

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

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

Pastor Sánchez, Juan Antonio. 2000. *Panorama actual de los Portales,* 2000.

Pastor Pérez, Lluís. 2006. Hacia una gestión integral de los contenidosRUSC. Universities and Knowledge Society Journal [en línea] 2006, 3 (octubre): [Fecha de consulta: 28 de noviembre de 2013.] Disponible en: http://www.redalyc.org/articulo.oa?id=78030203 ISSN 1698-580X

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

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

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

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

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

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.

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

Ribelles García, A. 2013. *Plataformas de publicación y distribución de audio y vídeo.* Cataluña. España: FUOC.

Fundación para la Universitat Oberta de Catalunya, 2013.

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

Sandoval, Martín, María Teresa. 2000. Medios de comunicación y publicidad en Internet. Revista Latina de Comunicación Social [en línea] 2000, 3 (diciembre): [Fecha de consulta: 28 de mayo de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=81933610

Sanguineti de, Brasesco, Susana. 2001. Estética en la comunicación audio. Revista Latina de Comunicación Social [en línea] 2001, 4 (enero): [Fecha de consulta: 28 de mayo de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=81943706

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

Soto Arango, Diana, Puig-Samper Mulero, Miguel Ángel. 2012. Documentos Revista Historia de la Educación Latinoamericana [en línea] 2012, 14 (Julio-Diciembre): [Fecha de consulta: 25 de abril de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=86926976016 ISSN 0122-7238

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

Tubella, Imma, Tabernero, Carlos y Dwyer, Vincent. 2008. *Internet y televisión: la guerra de las pantallas.* Barcelona, Ariel, 2008. 210 p.

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

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

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

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

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/mozilla/

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

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

Zapata, Ros, Miguel. 2002. Las buenas maneras en Internet RED. Revista de Educación a Distancia [en línea] 2002, (octubre): [Fecha de consulta: 28 de enero de 2014] Disponible en: http://www.redalyc.org/articulo.oa?id=54700501

Anexos

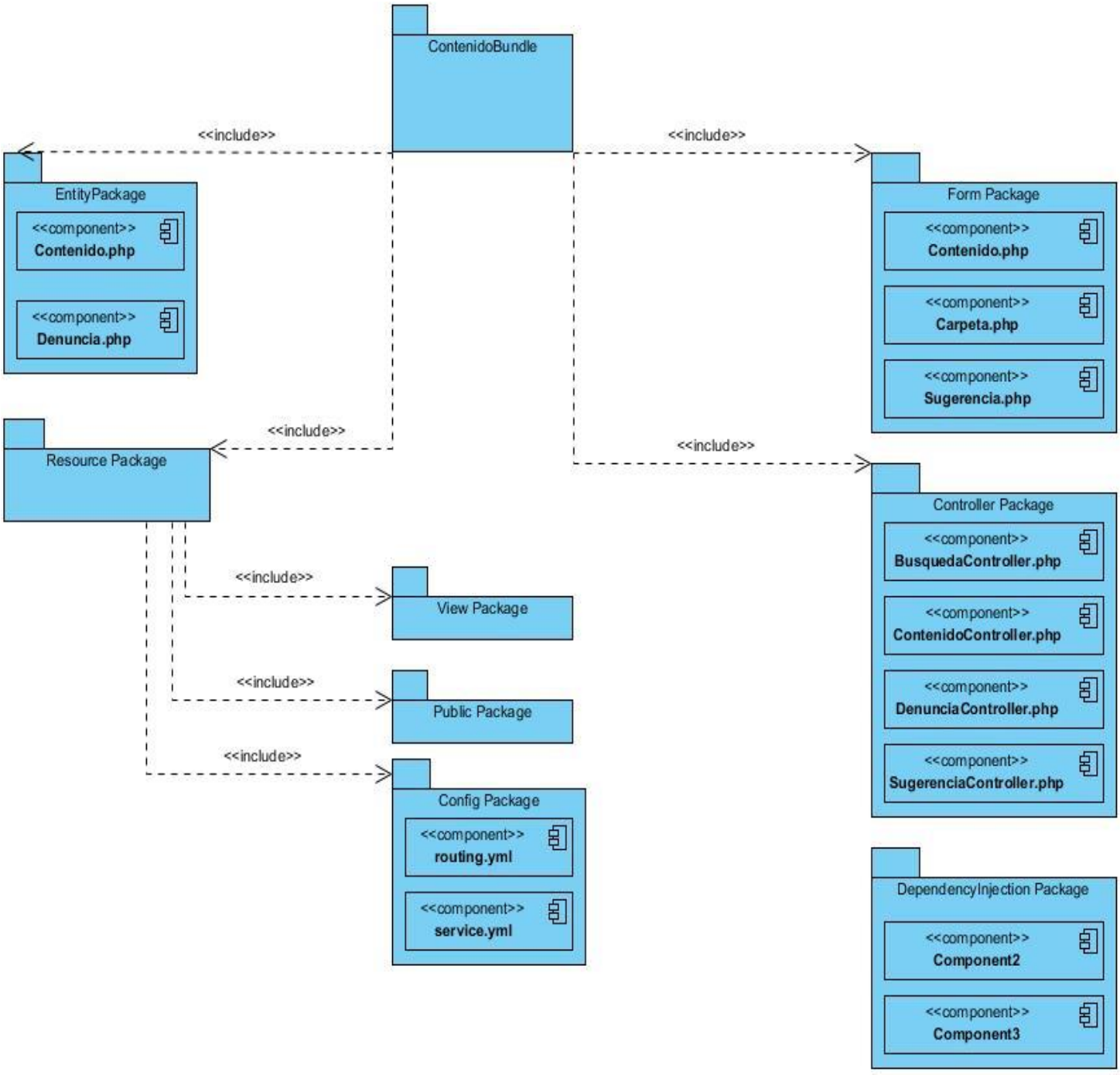


Figura 16. *Diagrama de componentes de ContenidoBundle*

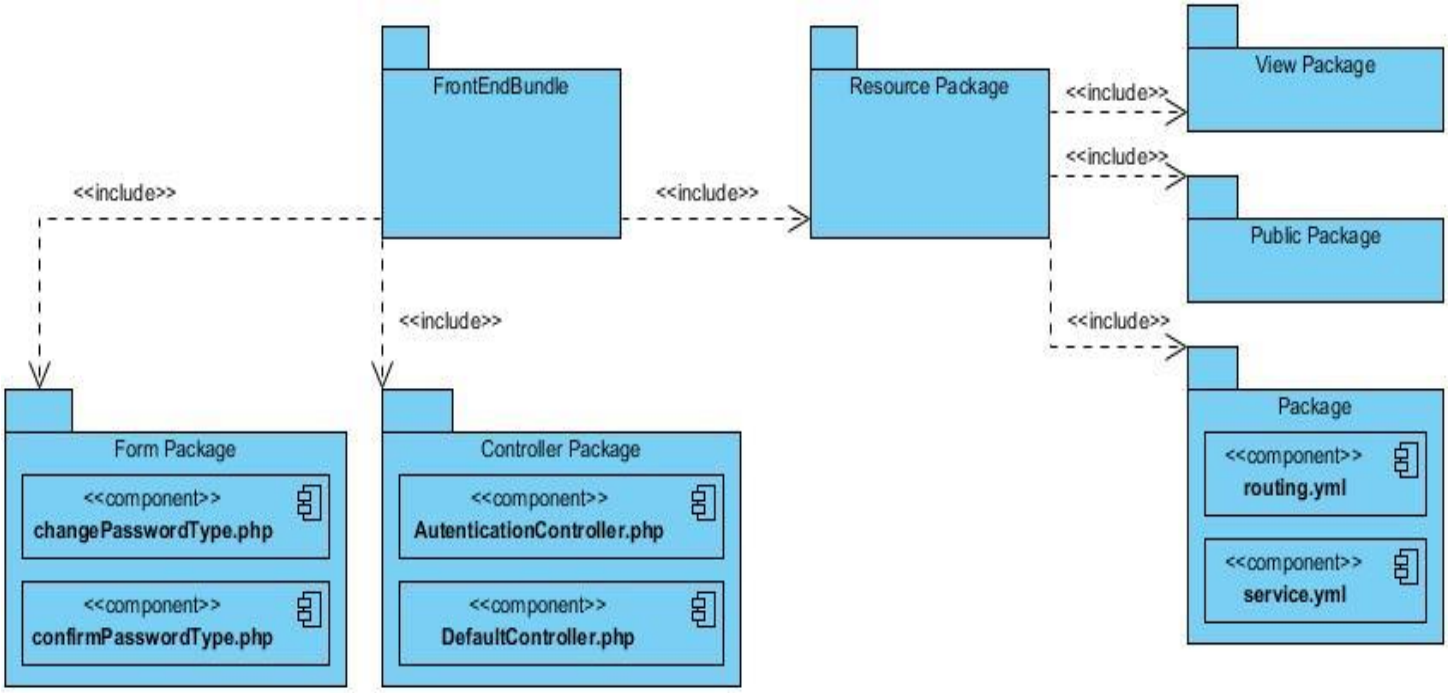


Figura 17. *Diagrama de componentes de FrontEndBundle*

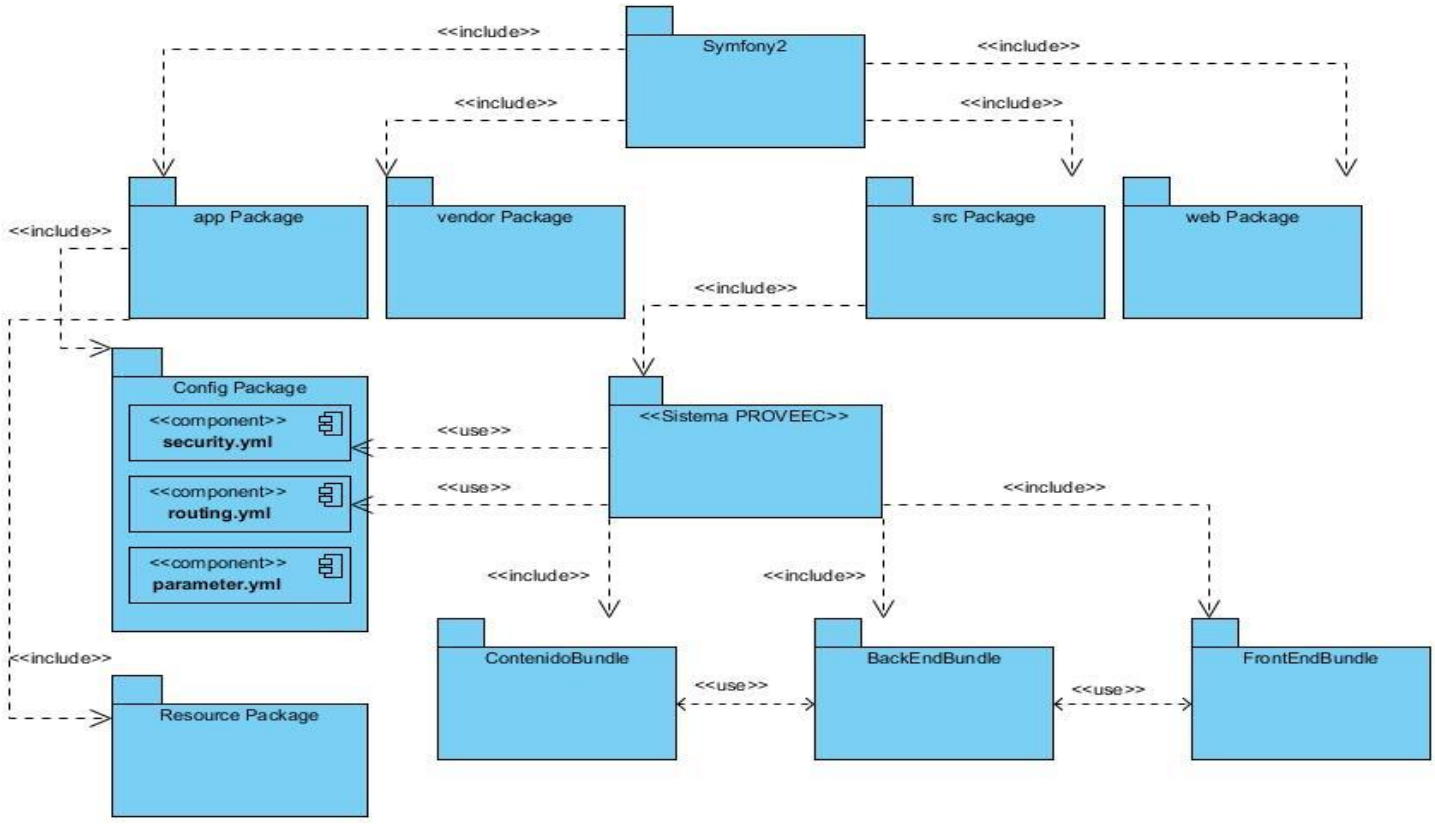


Figura 18. *Diagrama de componente general de Symfony2*

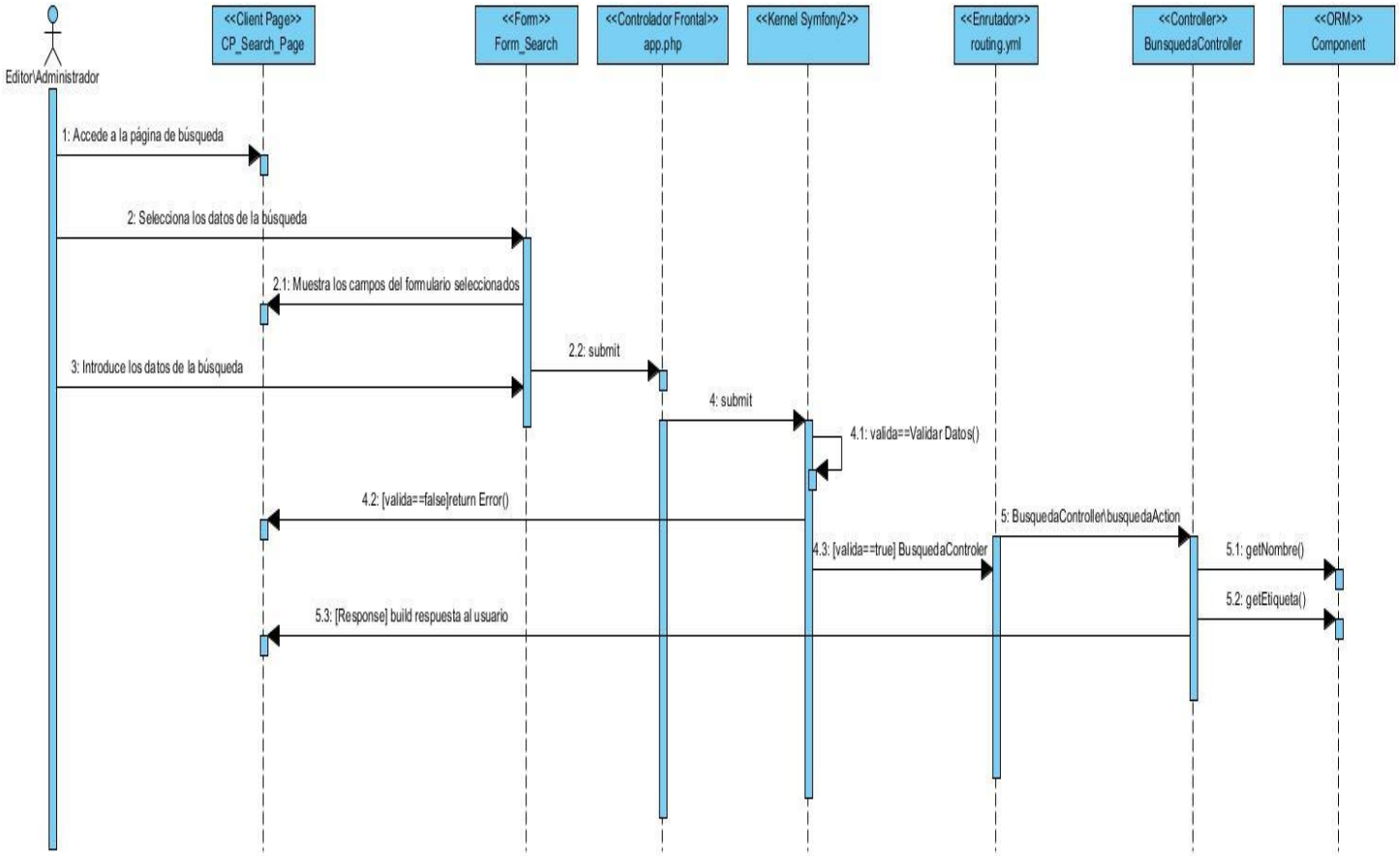


Figura 19. *Diagrama de Secuencia del CU "Publicar contenido"*

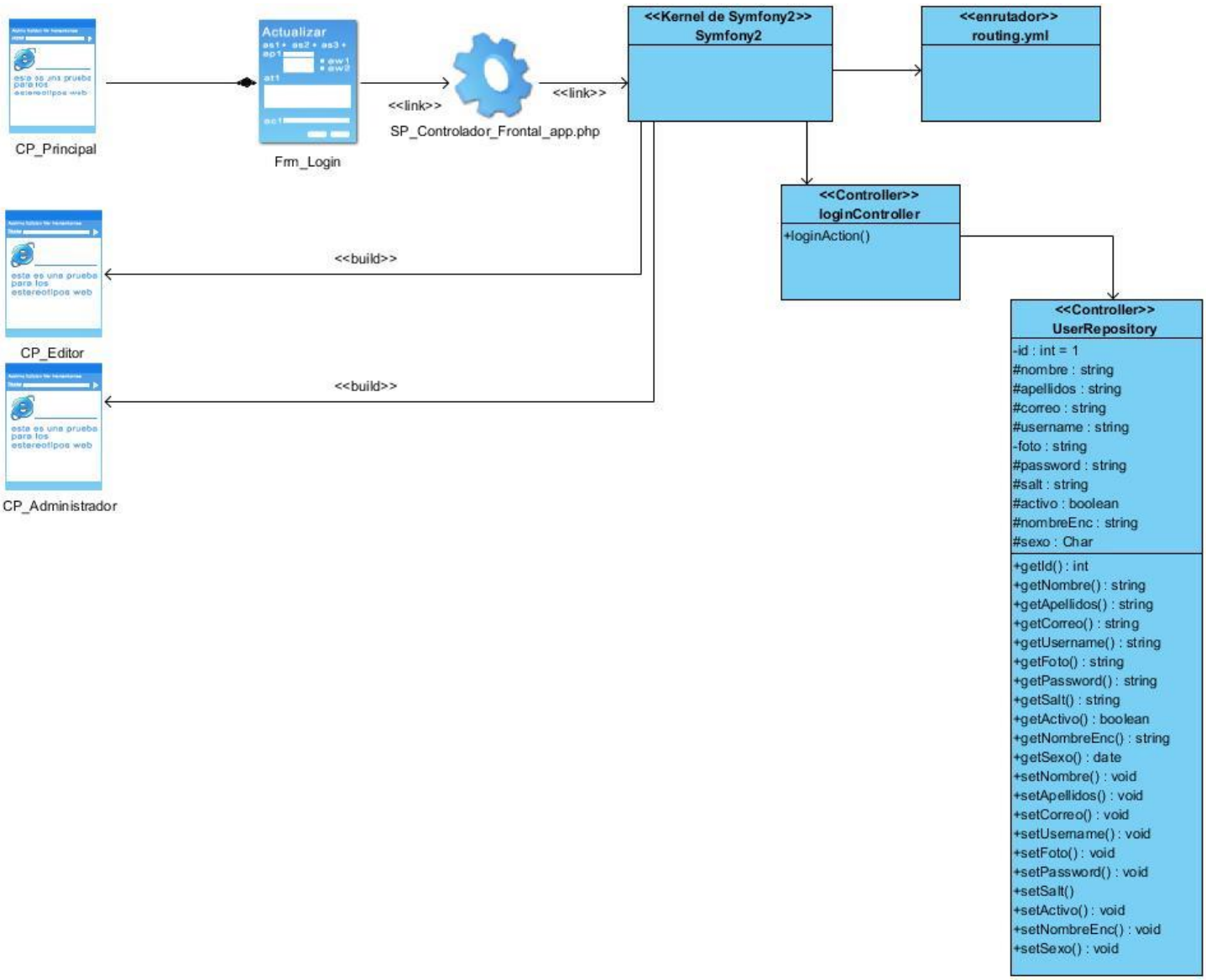


Figura 20. *Diagrama, modelo de clase de diseño del CU Autenticar Usuario*

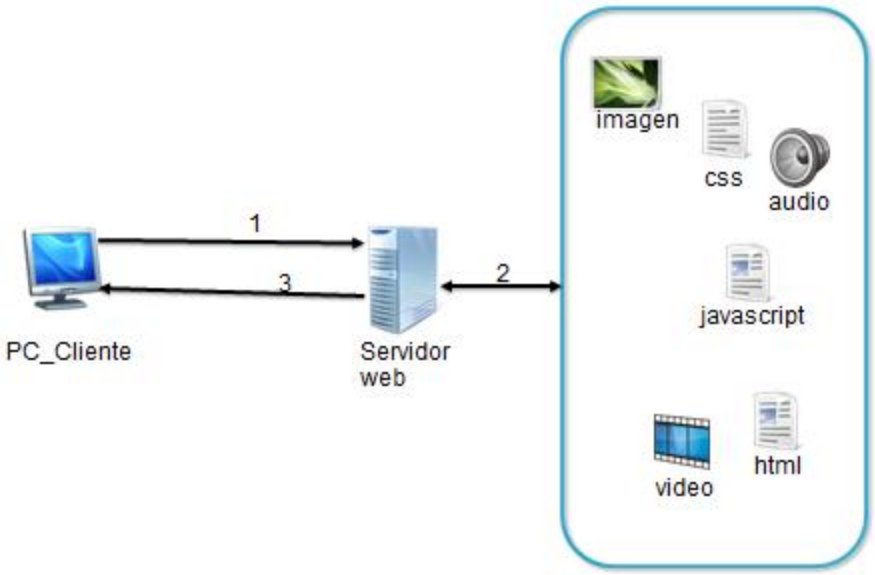


Figura 21. *Sistema web no distribuido*

Inicialmente **(1)** el navegador realiza una petición *HTTP* al servidor web sobre una página específica. El servidor web **(2)** procesa la petición y devuelve un número limitado de elementos de la página (generalmente, dos), por ejemplo, el código *HTML* y una de las imágenes. Estos son transmitidos **(3)** al navegador, el cual comienza la composición en pantalla. Para recibir a continuación dos elementos más, como por ejemplo un *script*, *css*, imagen, video o audio. Se ha de repetir todo el proceso **(1) (2) (3)** (Ribelles García, 2013). Tal como puede deducirse, este mecanismo está comprometido en situaciones de alta congestión, pues el número de peticiones puede superar la capacidad de gestión del servidor web. Para evitar o minimizar los efectos negativos de esta arquitectura en el lado del servidor se tiende a utilizar diferentes técnicas como: añadir una cache[38] o servidores en paralelo. Estas técnicas deberían ser suficiente para que el servidor no detenga el servicio, pero si el número de peticiones aumenta entonces es necesario utilizar sistemas proveedores de contenidos **distribuidos,** los cuales brindarían un servicio como muestra la Figura 22.

38 Memoria rápida que almacena una copia de cada elemento que es pedido por primera vez por cualquier cliente. El siguiente pedido que realiza cualquier otro cliente se lee de esta memoria rápida en vez de hacerlo desde el disco.

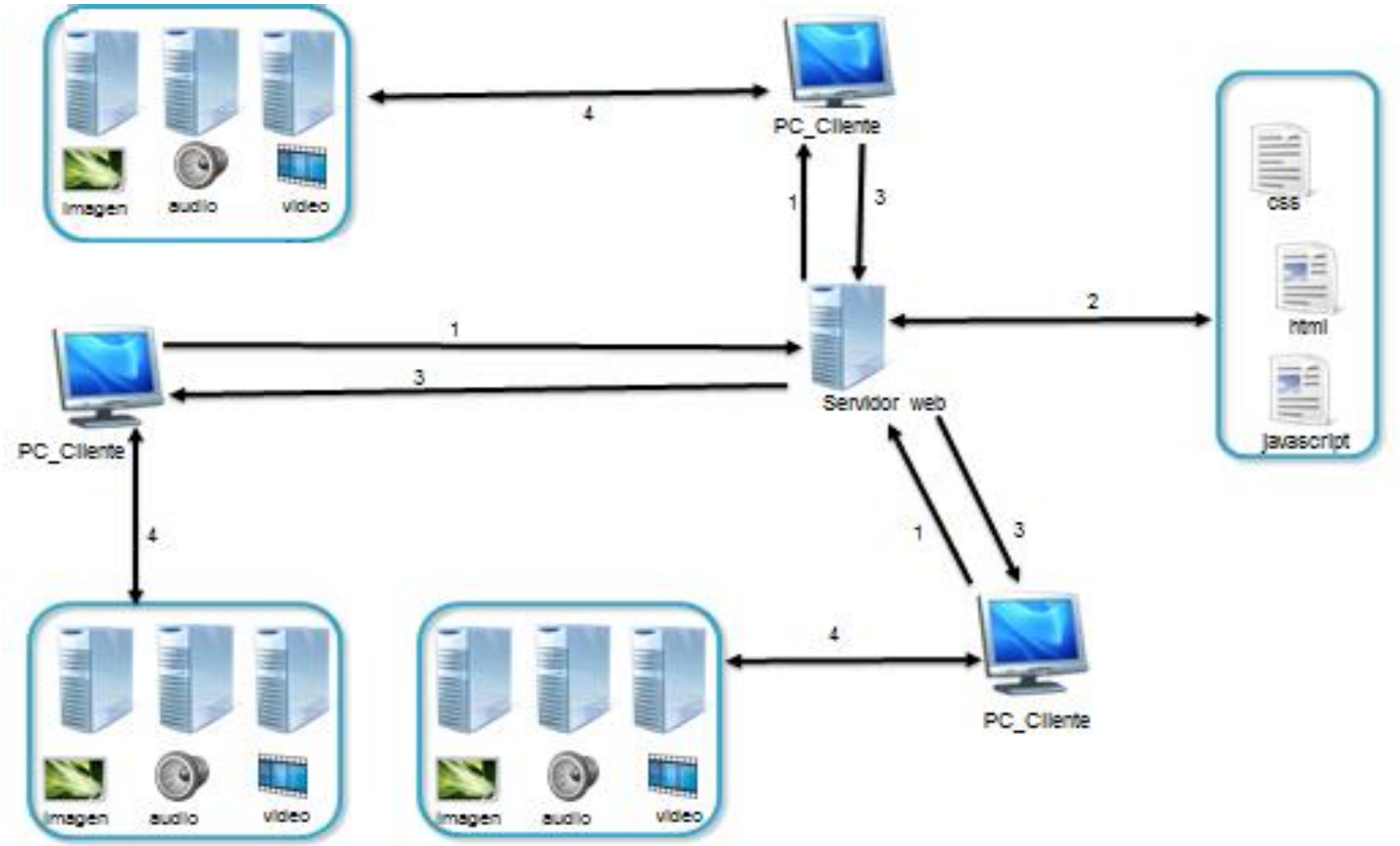


Figura 22. *Sistema web distribuido*

Cada cliente lee **(3)** la página HTML recogida **(2)** como respuesta a su petición **(1)**, en la página HTML se indican las direcciones genéricas de ubicación de los elementos, y mediante un mecanismo de direccionamiento a los servidores proveedores de contenidos se realizan el resto de peticiones **(4)**. Sin duda, se acelera dramáticamente la lectura de la página web, incluso en condiciones de alta carga. Esto es debido a que el servidor web solo se dedica a mostrar al usuario las páginas web, la información y los contenidos multimedia son entregados a los usuarios desde servidores distribuidos que se encuentran separados del servidor web. Todo se realiza de forma totalmente transparente a los usuarios.

UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 1
2013-2014

RECONOCIMIENTO

Otorgado a: Sistema para la Publicación y Distribución de Contenidos Multimedia

Por haber obtenido DESTACADO

en El Trabajo en Comisiones.

12ma jornada científica estudiantil

UCI FACULTAD 1

Claudia Durán Rodríguez
Presidente de la FEU

Damian Cervantes Rondón
Decano de la Facultad

Figura 23. *Jornada Científica Estudiantil*

Universidad de las Ciencias Informáticas
Federación Estudiantil Universitaria

Otorgan el presente:

RECONOCIMIENTO

A Sistema para la publicación de contenido multimedia

Por obtener el premio

PRODUCTO TECNOPLUS

en la Feria Estudiantil de Soluciones Informáticas

"Potenciando el intercambio para el desarrollo..."

Dado a los 22 días del mes de mayo de 2014.

Víctor Gabriel González Cardoso
Presidente de la FEU

Figura 24. *Feria Estudiantil de Soluciones Informáticas*

Escenario	Descripción	Usuario	Contraseña	Respuesta del sistema	Flujo central
EC 1.1 El usuario se autentica ante el sistema	En el formulario de autenticación el usuario introduce un par nombre de usuario y contraseña válidos, luego hace clic en el botón "Entrar".	V Nombre de usuario válido	V Contraseña válida, correspondiente al nombre de usuario	El usuario es autenticado, se le concede los permisos asociados a su rol y se muestra la página principal.	1 El usuario solicita acceder a una página del sistema protegida. 2. El sistema muestra al usuario el formulario de autenticación. 3. El usuario se dispone a llenar el campo nombre de usuario y contraseña.
EC 1.2 El usuario falla en la autenticación ante el sistema	En el formulario de autenticación el usuario introduce un par usuario y/o contraseña incorrecta, luego hace clic en el botón "Entrar".	I Nombre de usuario no válido	I Contraseña no válida.	Muestra la página de autenticación con el mensaje de error "Credenciales Incorrectas. Verifique que el bloqueo de mayúsculas no esté activado e inténtelo de nuevo".	

Tabla 8. *Caso de prueba de CU "Autenticar usuario"*

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Usuario	Entrada de texto	No	Corresponde al nombre de usuario. Puede ser cualquier cadena de caracteres alfanuméricos, que no comience con un número, y tenga un límite de 20 caracteres. También corresponde al correo electrónico del usuario.
2	Contraseña	Entrada de contraseña	No	Corresponde a la contraseña del usuario. Puede ser cualquier cadena de caracteres, dicha cadena puede tener un mínimo de 6 caracteres.

Tabla 9. *Variables de caso de prueba de CU "Autenticar usuario"*

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 El usuario tiene al menos un seguidor.	El usuario desea compartir un contenido si al menos tiene un seguidor.	El contenido se comparte con los seguidores del usuario y se muestra un mensaje "El contenido ha sido compartido".	1. El usuario selecciona el contenido que desea compartir haciendo clic en el nombre del contenido. 2. El usuario da clic en el link "Compartir"
EC 1.2 El usuario no tiene seguidores.	El usuario desea compartir un contenido y no posee seguidores.	El sistema muestra un mensaje de error "Usted no posee seguidores".	

Tabla 10. *Caso de prueba de CU "Compartir contenido"*

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 El usuario no ha dado "LIKE" al contenido seleccionado.	Al usuario le ha gustado el contenido seleccionado.	El sistema aumenta el número de "LIKE" que tiene el contenido seleccionado.	1. El usuario selecciona el contenido que desea observar. 2. El usuario da clic en el link "Me gusta" ubicado en la parte inferior derecha del contenido seleccionado
EC 1.2 El usuario ha dado "LIKE" al contenido seleccionado.		El sistema disminuye el número de "LIKE" que tiene el contenido seleccionado.	

Tabla 11. *Caso de prueba de CU "Dar me gusta, a un contenido"*

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 El usuario desea seguir contenidos de su interés.	El usuario desea seguir un contenido de su interés.	El sistema agrega el contenido seleccionado a la sección "Contenido de interés" del "Perfil" del usuario.	1. El usuario se encuentra en la página principal del sistema y selecciona el contenido que desea seguir. 3. El usuario da clic en el menú desplegable "Opciones" y da clic en la opción "Seguir".(En caso que el usuario se encuentre en las páginas "Reciente" o "Popular" solamente realizar el paso 3)

Tabla 12. *Caso de prueba de CU "Seguir contenido"*