UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD #4

“Entorno de desarrollo para el Marco de trabajo de creación de multimedia”

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

Autores:

Rosmelys Díaz Sendín

Richard Romero Rosell

Tutores:

MSc. Iván Pérez Mallea

Ing. Roberto Alejandro García Rodríguez

Cotutores:

Ing. Lizandra Hernández Hernández

Ing. Alejandro Rodríguez de la Cruz

La Habana, junio de 2016

“Año 58 de la Revolución”

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) a hacer uso del mismo en su beneficio.

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

Autores:

_____ _____

Rosmelys Díaz Sendín Richard Romero Rosell

Tutores:

_____ _____

MSc. Iván Pérez Mallea Ing. Roberto Alejandro García Rodríguez

Cotutores:

_____ _____

Ing. Lizandra Hernández Hernández Ing. Alejandro Rodríguez de la Cr

De Rosmelys:

*A **Fidel y la Revolución**: Por ser el eslabón principal en esta cadena de logros alcanzados.*

*A mi madre **Elsa**: Por su esfuerzo y apoyo para que finalmente llegara este momento.*

*A mi padre **Ramón**: Por apoyarme y enseñarme en todos los momentos de mi vida.*

*A mi hermana **Raimelys** que la quiero con la vida y sé que también me quiere mucho. La considero madre y amiga: Por estar siempre pendiente, apoyarme y cuidar de mí.*

*A mi compañero de tesis **Richard:** Pues sin el este trabajo no se habría realizado.*

*A mis Tutores **Iván Mallea y Roberto**: Por ayudarme, apoyarme y contribuir a realizar mis metas.*

*A mis cotutores: **Lizandra** y **Alejandro:** Por ser una guía importante para nuestro resultado.*

*A mi **Tata**, **tío Manolo**, **Mami** y **Papi**: Por quererme tanto y estar tan pendientes de mí. Por ser como otros padres para mí.*

*A mi novio **Alain**: Por su comprensión y cariño.*

*A mi suegra **Alexis**: Por ser como una madre y ocuparse tanto de mí.*

Al resto de mi familia tanto materna como paterna, así como a la de mi novio que considero mía también: Por apoyarme siempre de una manera u otra.

*A mis amigos que me han dedicado tanto, entre ellos a **Lien**, **Grisell**, **Isbel** , **Luis Miguel** y **Ariel**: Por ser tan buenos amigos y ayudarme con una lágrima, más que con una sonrisa*

Todos esos compañeros que a lo largo de mis 5 años de carrera han formado parte de mi vida: Por todo el tiempo compartido. Por lograr que mis años de universidad se hagan inolvidables.

A los profesores que me han regalado su enseñanza y aún más a los que fueron más que profesores, amigos.

A tantas otras personas que se me haría difícil mencionar por nombres pero que me han ayudado tanto y les debo estar hoy aquí. Siempre les guardo un lugar en mi corazón, muchas gracias.

De Richard:

A mi madre ***María Cristina*** *y mi padre* ***Guillermo****, por todo su esfuerzo y dedicación hacia mis estudios, y por todo su amor y felicidad que me han dado durante toda mi vida. Los quiero mucho.*

A mi hermano ***Alejandro****, por sus enseñanzas y aportes a mis estudios, y por ser un amigo en el que confiar.*

A mi tío ***Vicente****, mi amigo, que siempre lo tengo presente, en especial a la hora de tomar decisiones.*

A toda mi familia por su apoyo, comprensión y amor durante todos los momentos difíciles.

A todos mis amigos, en especial a ***Amalia****,* ***Carlos*** *y* ***Eduardo****, por ser tan buenos amigos y ayudarme durante todos estos años.*

A todos mis compañeros de aula que a lo largo de estos 5 años hemos compartido muy buenos momentos. Por lograr que mis años de universidad se hagan inolvidables.

A mis tutores ***Iván Mallea*** *y* ***Roberto****: por ayudarme, apoyarme y contribuir a realizar esta ardua labor.*

A mis cotutores: ***Lizandra*** *y* ***Alejandro****: Por ser una guía para nuestro resultado.*

A todos los profesores que he tenido durante todos mis años de estudios, me enseñaron a estudiar y descubrir nuevas experiencias.

Al profesor ***Piña****, por esclarecerme muchas dudas y guiarme en el desarrollo de la tesis.*

A todas las amistades que han estado presente durante el desarrollo de esta tesis, todas me han ayudado mucho y aportado su granito de arena.

Y a mi compañera de tesis ***Rosmelys****, por ser una excelente compañera, su apoyo y ayuda en la creación de esta tesis fue de primera.*

Gracias *a todos.*

Resumen

En el Centro de Tecnologías para la Formación (FORTES) de la Universidad de las Ciencias Informáticas (UCI) surgió el proyecto “Marco de trabajo para el desarrollo de multimedia con tecnologías libres”, una herramienta libre y multiplataforma que constituye una alternativa al uso de herramientas privativas que se utilizan para la creación de estos productos. En la actualidad el marco solo posee un conjunto de componentes aislados y no existe un mecanismo que los integre y gestione, lo que implica que el proceso sea lento, engorroso y demande un mayor esfuerzo del equipo de desarrollo. El presente trabajo se enfocó en elaborar un entorno de desarrollo que permita crear productos multimedia en un menor tiempo y con mayor facilidad, a partir de la integración y gestión de los componentes existentes. El desarrollo de la propuesta de solución fue guiado por la metodología de desarrollo AUP, donde el cliente fue participe en el proceso de obtención de la herramienta. Además de las funcionalidades descritas por el cliente, fueron determinadas otro grupo a través del estudio de herramientas similares. También fueron definidas las tecnologías, herramientas y lenguajes a utilizar en el desarrollo del sistema. Fueron generados los artefactos adecuados para documentar la herramienta. Como resultado se obtuvo un entorno de desarrollo que permitió mejorar el proceso de creación de multimedia en el centro FORTES. Fueron realizadas las pruebas de software a la propuesta de solución, que comprobaron la correcta implementación de las funcionalidades y un alto grado de satisfacción por parte del cliente.

Palabras clave: entorno de desarrollo, marco de trabajo, multimedia

Índice

INTRODUCCIÓN

Las Tecnologías de la Información y las Comunicaciones (TIC) han alcanzado un gran auge en la sociedad, haciéndose evidente su utilización en innumerables sectores del país y el mundo. A diario surgen nuevas necesidades que pueden ser satisfechas con la informatización. Para ello se desarrollan nuevas tecnologías o son reutilizadas otras ya existentes para contribuir a la solución del problema.

Desde los inicios de la Revolución cubana en 1959 hasta la actualidad se ha logrado expandir la educación hacia cada rincón del país. Como evidencia de este logro se halla la Universidad de las Ciencias Informáticas (UCI), la cual surge en el 2002 como primer centro de estudios universitarios de la Batalla de Ideas. Fidel Castro Ruz convocó a que se convirtiera en una "universidad de excelencia", donde ingresan estudiantes de todo el país que vinculan los procesos de Investigación más Desarrollo. La UCI desarrolla un modelo de formación con un componente laboral que comienza en el tercer año de la carrera vinculando a sus estudiantes a centros de desarrollo de software. Su misión es producir software y servicios informáticos generando soluciones informáticas para Cuba y otros países. El desarrollo de software multimedia constituye un ejemplo de estas soluciones informáticas, brinda al usuario la interacción a través de imágenes, sonido, videos, textos y animaciones.

El Centro de Tecnologías para la Formación (FORTES) de la UCI tiene como misión desarrollar tecnologías que permitan ofrecer servicios y productos, para brindar soluciones de formación aplicando las TIC, a todo tipo de instituciones con diferentes modelos de formación y condiciones tecnológicas. Cuenta con varias líneas de desarrollo, dentro de estas se encuentra la Línea de Producción de Recursos Educativos (PRE), que se encarga de producir software multimedia. Entre los resultados obtenidos en la Línea se encuentran los productos multimedia Reflexiones de Fidel, Guía náutica y Mis mejores cuentos.

Para el desarrollo de estos productos multimedia se utilizan diversas herramientas como: Articulate y Lectora, pero fundamentalmente Adobe Flash CS6. Este software es privativo y no se cuenta con una licencia para su uso. La adquisición de esta licencia es costosa y está afectada por el bloqueo impuesto por los EUA al país que no puede negociar de forma directa con la compañía Adobe. Además, esta herramienta no es multiplataforma lo que ata al desarrollador a la plataforma Windows que también es privativa. Otro aspecto negativo que posee Flash es que la curva de aprendizaje del lenguaje ActionScript es elevada, requiriendo por parte de los especialistas una cantidad considerable de tiempo para poder dominarla.

Como política la Universidad exige la utilización de tecnologías libres para el desarrollo de sus productos como garantía de la soberanía tecnológica en sus soluciones, este es otro aspecto que atenta contra el uso de este producto de Adobe.

Por esta razón surge dentro de esta Línea el proyecto “Marco de trabajo para el desarrollo de multimedia con tecnologías libres”, que en la actualidad solo posee un conjunto de componentes que constituyen la base para la creación de los productos multimedia. (1)

Pero esto no es suficiente, ya que no existe una herramienta que integre y gestione estos componentes. Esto trae consigo que el proceso sea lento, engorroso y demande un mayor esfuerzo del equipo de desarrollo.

Es por esto que surge la necesidad de dar solución a la situación antes expuesta, en tal sentido el **problema a solucionar** consiste en: el proyecto Marco de trabajo para el desarrollo de multimedia con tecnologías libres del centro FORTES no posee ningún mecanismo que permita integrar y gestionar los componentes existentes, imposibilitando así la creación de este tipo de productos.

Para dar respuesta a este problema se define como **objeto de estudio**: el proceso de desarrollo de multimedia.

Teniendo como **campo de acción**: el proceso de desarrollo de multimedia con tecnología web.

Cumpliendo con lo antes expuesto, se propone como **objetivo general**: elaborar un entorno de desarrollo para el Marco de trabajo de creación de multimedia con tecnologías libres, que permita la integración y gestión de componentes existentes.

Del mismo se derivaron los siguientes **objetivos específicos**:

1. Elaborar el diseño teórico-metodológico de la investigación.
2. Realizar el análisis y diseño del entorno de desarrollo para la creación de multimedia.
3. Implementar la propuesta de solución.
4. Realizar pruebas para garantizar la funcionalidad y calidad de la herramienta.

Para dar respuesta a los objetivos se plantearon las siguientes **tareas**:

1. Elaboración del diseño teórico de la investigación.

2. Confección de artefactos generados por la metodología de desarrollo de software seleccionada.
3. Implementación de la solución propuesta.
4. Realización de las pruebas pertinentes a la solución propuesta.

La **hipótesis** que se plantea es: la creación de un entorno de desarrollo para el Marco de trabajo de creación de multimedia con tecnologías libres, que integre y gestione los componentes existentes, permitirá disminuir el tiempo de desarrollo de estos productos.

El **resultado** que se obtendrá con esta investigación será: entorno de desarrollo para el Marco de trabajo de creación de multimedia con tecnologías libres que permitirá crear multimedia, a través de la integración y gestión de componentes. El mismo proveerá una secuencia para la creación del producto y contará con valores añadidos como la exportación del producto a diferentes formatos. Además, generará la estructura base de cada multimedia y brindará funcionalidades que permitan ajustar el diseño de la multimedia a los requerimientos del usuario.

Los **métodos de investigación científica** seleccionados para conseguir el objetivo trazado son:

Métodos Teóricos:

Analítico - Sintético: fue utilizado para extraer información sobre cada una de las partes que componen el objeto de estudio en forma individual, y luego llegar a estudiarlas de manera integral aportando al desarrollo de la propuesta de solución.

Histórico - Lógico: permitió estudiar de forma analítica la evolución de las herramientas utilizadas para el desarrollo de las multimedia, con el objetivo de extraer características y funcionalidades que se podrían incluir en el entorno de desarrollo.

Métodos Empíricos:

Entrevista: constituyó un medio significante para la obtención de información por parte de expertos en el tema. Se identificaron las necesidades reales de la línea de desarrollo PRE del centro FORTES, permitiendo así fijar el objetivo por el que se trabajó en la propuesta de solución.

Observación: posibilitó tomar las características y funcionalidades de sistemas similares a incluir en la herramienta a desarrollar. Así como apreciar datos durante la validación de la investigación.

La **estructura capitular** de este documento se definió de la siguiente forma:

Capítulo 1: Fundamentación teórica.

En este capítulo se plantean los principales conceptos que fundamentan la investigación. Además, presenta el estado del arte y un breve repaso de los antecedentes sobre el tema de la investigación. El capítulo cuenta con un análisis de las soluciones similares existentes y se describe la metodología de desarrollo de software. Por último, se definen las herramientas y tecnologías que serán empleadas para dar cumplimiento a la propuesta de solución.

Capítulo 2: Descripción de la propuesta de solución.

En el presente capítulo se exponen los requisitos funcionales y no funcionales que deberá cumplir la propuesta de solución. Agrupa una serie de diagramas de negocio generados a partir del problema definido. Además, se confeccionan los artefactos de ingeniería concebidos por la metodología de desarrollo de software seleccionada.

Capítulo 3: Implementación y pruebas.

Este capítulo presenta una descripción del proceso de implementación de la propuesta de solución. Además, se describen las pruebas realizadas para el análisis de la aplicación con el fin de verificar que el resultado obtenido satisfaga las necesidades existentes.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

Este capítulo tiene como objetivo describir teóricamente los principales conceptos relacionados con el problema a resolver, constituyendo una base para un mejor entendimiento y desarrollo del presente trabajo.

1.2 Generalidades de la multimedia

1.2.1 Multimedia

A finales de la década de los 90 surge la multimedia, definida como la combinación de varios medios de información en una computadora (textos, imágenes, audio, videos y animaciones).

En el campo de las nuevas tecnologías al término multimedia se le ha dado diversos enfoques. Han sido analizados algunos conceptos para un mejor entendimiento de qué es una multimedia:

- Según la Real Academia Española el vocablo multimedia se refiere a "*que utiliza conjunta y simultáneamente diversos medios, como imágenes, sonidos y texto, en la transmisión de una información*." (2)

- Myriam Ruíz Sánchez y María Isabel Leal Rugama ofrecen la siguiente definición en su artículo Multimedia: *"Multimedia es cualquier combinación de texto, arte gráfico, sonido, animación y video que llega al usuario por computadora u otros medios electrónicos. Es un centro de información tan poderoso, expresivo y natural que logra que se capte en forma mucho más efectiva la información que se recibe, estimulando increíblemente los sentidos, haciendo que el usuario esté mucho más alerta y receptivo. Todo esto es porque permite interactuar con los sonidos, las imágenes, los colores y la acción."* (3)

- María Pinto, en la página web Alfamedia, expone la siguiente manera: "*En el ámbito de la informática, la expresión multimedia se utiliza para calificar y describir las cualidades de diversos sistemas, aplicaciones, documentos y productos –programas educativos, juegos, obras de referencia, etc.– que emplean una combinación de textos, gráficos, imágenes, vídeos, animaciones en 3D y sonidos, e incluyen conjuntos de relaciones predefinidas que enlazan unos elementos a otros –denominadas hipervínculos para permitir a los usuarios desplazarse por la información de forma intuitiva e*

interactiva. Estos enlaces intentan emular la forma de procesar y comunicar información empleada por la mente humana que opera mediante la asociación dinámica de ideas similares." (4)

Partiendo de estas definiciones fue elaborado un concepto de multimedia más general. Se llegó a la conclusión de que una multimedia es un sistema informático que combina medios de información (imagen, texto, sonido, animación y video), posibilitando la interacción del usuario con los mismos a través de un dispositivo electrónico. Se construye sobre un tema que debe transmitir un conocimiento que llegue al receptor de una manera dinámica y comprensible.

Algunas de las **características** que debe tener una multimedia son (5):

1. **Ramificación:** capacidad que tienen los sistemas para responder al usuario.
2. **Transparencia**: debe de estar organizada de forma que la audiencia se centre más en lo que comunican, que en el medio empleado para ello.
3. **Navegación**: la capacidad que posee para que el usuario pueda desplazarse por la misma, en forma no secuenciada y lineal.
4. **Interactividad**: ha transformado a las multimedia en algo más atrayente. Cambió a aquel simple conjunto de medios de información, permitiendo que evolucionaran junto al vertiginoso avance que han adquirido las tecnologías con el paso del tiempo. La interactividad apunta al grado con el cual el ser humano se "integra" con las herramientas automatizadas para crear un proceso de ingeniería inversa efectivo. (6)

1.2.2 Medios de información

La integración de diferentes medios de información en la multimedia bajo una presentación interactiva, proporciona una gran riqueza en los tipos de dato. Esto dota de mayor flexibilidad a la expresión de la información. Diferentes textos, imágenes y otros tipos de contenidos se van secuenciando de una forma dinámica. (7)

A continuación se presenta una descripción de los medios de información que suelen utilizarse en una multimedia (8):

- **Texto** en todas sus formas: La inclusión de texto en las aplicaciones multimedia permite desarrollar la comprensión lectora, discriminación visual, fluidez verbal, vocabulario, etc. El texto tiene como función principal favorecer la reflexión y profundización en los temas, potenciando el pensamiento de más alto nivel. En las aplicaciones multimedia, además puede ser utilizado para aclarar la información gráfica o icónica. Atendiendo al objetivo y usuarios a los que va destinada la aplicación multimedia se puede reforzar el componente visual del texto. Esto se logra mediante modificaciones en su formato, resaltando la información más relevante y añadiendo claridad al mensaje escrito.

- **Audio** (música y sonidos): Los sonidos se incorporan en las aplicaciones multimedia para facilitar la comprensión de la información clarificándola. Los sonidos que se incorporan pueden ser locuciones orientadas a completar el significado de las imágenes, música y efectos sonoros. Esto consigue un efecto motivador captando la atención del usuario, siendo relevantes para algunas temáticas como aprendizaje de idiomas y música. Otra aplicación fundamental es en las multimedia cuya finalidad es la intervención en problemas de comunicación y/o lenguaje. Así mismo, la inclusión de locuciones y sonidos favorece el refuerzo de la discriminación y memoria auditiva. Algunos programas incluso permiten grabar, modificar e incorporar efectos a los archivos de sonido.

- **Iconográficos**: Un elemento habitual en las aplicaciones multimedia son los elementos iconográficos. Los mismos garantizan la representación de palabras, conceptos, ideas mediante dibujos o imágenes, representando lo esencial de la idea a transmitir. Su carácter visual le da un carácter universal, no sólo particular, son por ello adecuadas para la comunicación de ideas o conceptos en disímiles aplicaciones. Estas pueden ser utilizadas por personas que hablan diferentes idiomas o con distintos niveles en el desarrollo del lenguaje.

- **Imágenes estáticas** (fotografías, gráficos e ilustraciones): Las imágenes estáticas tienen gran importancia en las aplicaciones multimedia. Su finalidad es ilustrar y facilitar la comprensión de la información que se desea transmitir. Se pueden distinguir diferentes tipos de imágenes: fotografías, representaciones gráficas, fotogramas, ilustraciones, etc.

- **Imágenes dinámicas** (vídeo y animaciones): Las imágenes en movimiento son un recurso de gran importancia, puesto que transmiten de forma visual secuencias completas de contenido. Además, ilustran un apartado de contenido con sentido propio. Mediante ellas, en ocasiones pueden simularse eventos difíciles de conocer u observar de forma real. Pueden ser videos o animaciones. La animación permite a menudo un control mayor de las situaciones mediante esquemas y figuraciones.

Algo fundamental en el trabajo con multimedia es la utilización de una red de nodos. Estos son conectados por enlaces donde se almacenan los datos permitiendo manejar y organizar la información. Si los nodos contienen texto entonces se está haciendo referencia al concepto de **hipertexto**. Por otra parte si contienen además gráficos, imágenes, audio, animaciones y video, así como código ejecutable u otra forma de datos, se les da el nombre de **hipermedia**.(9)

1.3 Entorno de desarrollo para la creación de multimedia

Para el desarrollo de un producto de software no basta solo con los recursos humanos. Se necesita además de un entorno de desarrollo que proporcione servicios al desarrollador o programador para facilitarle este proceso.

Un ambiente o entorno de desarrollo, es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software. Normalmente, consiste de un editor de código fuente y herramientas de construcción automáticas. Muchos garantizan realizar la construcción o ensamblaje del software de manera visual permitiendo integrar componentes y disminuyendo la escritura de código fuente.

Cuando un entorno de desarrollo integra un grupo de herramientas como completamiento de código, depuradores, compiladores e intérpretes entre otros se dice que es un entorno de desarrollo integrado (IDE). El límite entre un IDE y un entorno de desarrollo de software más amplio no está bien definido. Muchas veces, a los efectos de simplificar la construcción de la interfaz gráfica de usuario (GUI, por sus siglas en inglés) se integran un sistema controlador de versión y varias herramientas. Muchos IDE modernos también cuentan con un navegador de clases, un buscador de objetos y un diagrama de jerarquía de clases, para su uso con el desarrollo de software orientado a objetos. Los IDE proveen un espacio de trabajo amigable para la mayoría de los lenguajes de programación. (10)

Los entornos clásicos de desarrollo de multimedia se centran en el diseño visual de la misma a partir de un grupo de componentes relegando a un segundo plano la escritura de código.

1.4 Estudio comparativo de herramientas similares

Normalmente los programas para la creación de multimedia, como se afirmó en el epígrafe anterior, están formados por un grupo de herramientas (subsistemas), cada uno de ellos responsable de determinadas tareas. El número y características de estos elementos varían según el programa en concreto, aunque la mayoría de estos sistemas proporcionan varias de las siguientes características (11):

- Procesamiento de texto, que permiten escribir tanto el contenido (los datos e informaciones) como la programación (instrucciones de funcionamiento) de las aplicaciones.
- Gestión de base de datos: herramienta para organizar la información y permitir su posterior consulta y utilización.
- Edición de video, hacen posible la digitalización de imágenes y su edición (“montaje”), incluyendo la aplicación de efectos de vídeo (“postproducción”).
- Edición y generación de sonido.
- Producción de gráficos o dibujos para imagen fija. Ilustración (3D): modelado y renderizado de objetos.
- Animación.

Teniendo en cuenta estas características, se ha incrementado el desarrollo de herramientas para la creación de multimedia. Las mismas varían desde plataformas en la web, hasta aplicaciones de escritorio, todas estas con características y funcionalidades diferentes. Desde hace varios años se difundió el uso de algunas herramientas privativas como Adobe Flash, Macromedia Director y Asymetrix ToolBook II.

Flash, que antes pertenecía a la compañía Macromedia y ahora pertenece a Adobe es un programa de edición multimedia que utiliza gráficos vectoriales, imágenes, sonido, código de programa, flujo de vídeo y audio bidireccional para crear proyectos multimedia Flash es el entorno de desarrollo y Flash Player es el programa que se utiliza para ejecutar los archivos generados con Flash. Los proyectos multimedia pueden ser desde simples animaciones hasta complejos programas. Además de los gráficos, videos y sonidos, Flash incorpora ActionScript, un completo lenguaje de programación que expande las posibilidades en los

proyectos. Los archivos de Flash suelen tener formato SWF y aparecen con frecuencia en páginas web en forma de animaciones y aplicaciones. (12)

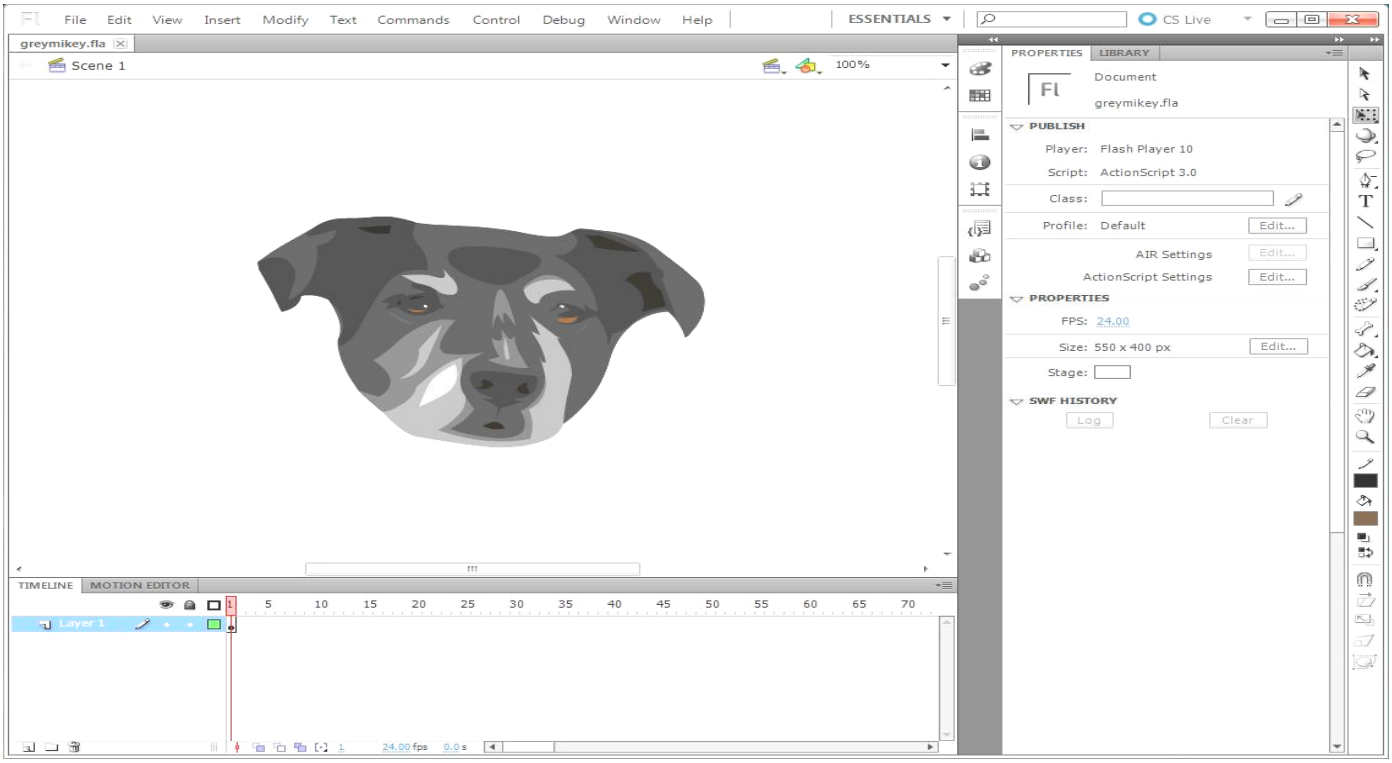


Figura 1 Área de trabajo de Adobe Flash CS6

El **Director** de Macromedia es una herramienta de creación centrado en multimedia interactiva, animación y juegos, CD e Internet. Se ha ganado una aproximación amplia a Internet, donde sus aplicaciones se ejecutan a través del plugin[1] web para Macromedia Shockwave. Es una herramienta multiplataforma, siendo capaz de generar archivos ejecutables para Windows y Macintosh a partir de los mismos archivos de origen, siempre y cuando tenga la versión de la herramienta para ambas plataformas. El plugin de Shockwave también está disponible para PC / Mac. El Director se basa en la metáfora de "películas". Cada archivo es una película en la que el progreso de la aplicación se produce en un script que muestra la secuencia de marcos que se muestran en la pantalla con el tiempo. Los objetos se llaman actores y se agrupan en escuadrones. (13)

[1] Accesorio, Añadido, Módulo

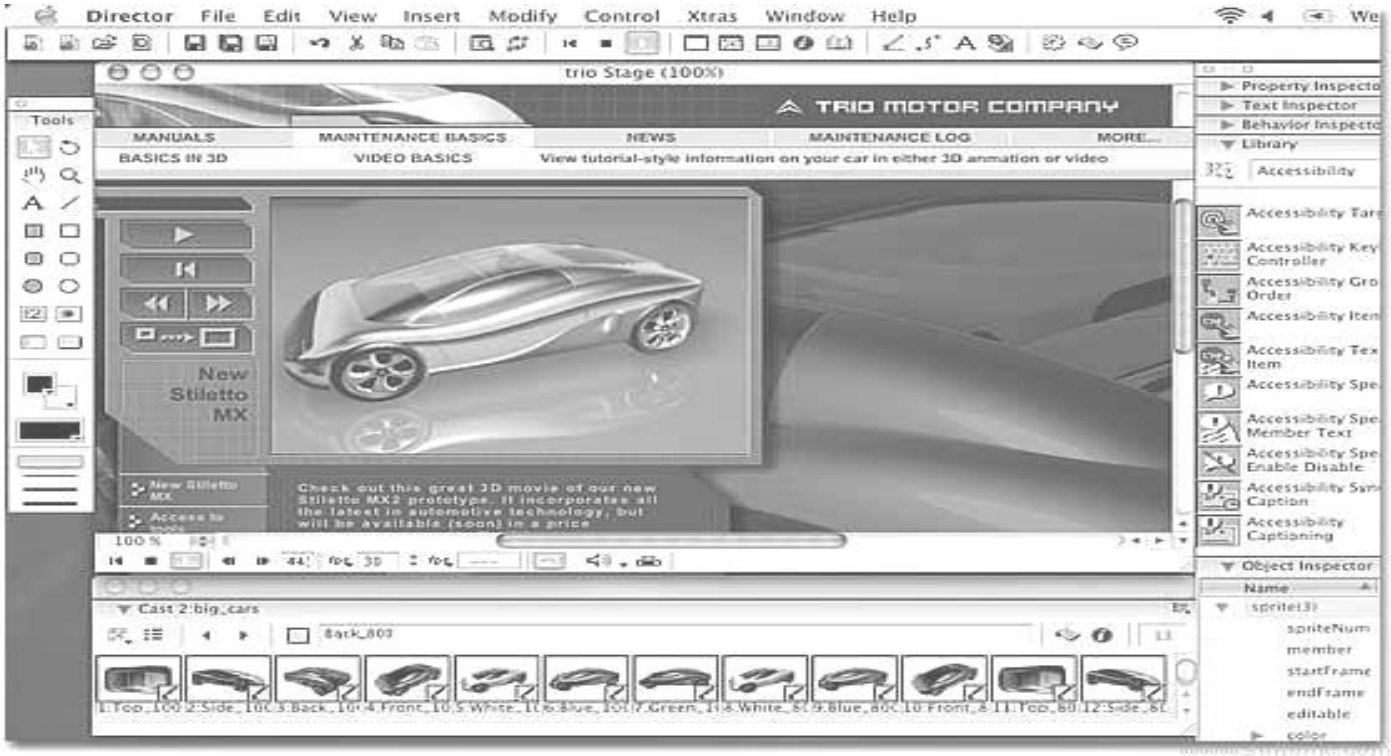


Figura 2 Área de trabajo de Macromedia Director

El **ToolBook** de SumTotal[2] ha emergido como una herramienta de creación centrado en multimedia en general (la primera versión llegó a ser comparado con Visual Basic), pero se centra actualmente en el desarrollo de aplicaciones de aprendizaje en línea (e-Learning), formación asistida por ordenador (TBC / TCC), simulaciones, tutoriales y cursos en línea a través de la web o en CD. Tiene un potente lenguaje de programación propia, el OpenScript. Es una herramienta específica para la plataforma Windows, que tiene muchas características de integración con el sistema, tales como el uso de controles ActiveX y la inserción de objetos OLE, plena interacción con las aplicaciones con DDE (Dynamic Data Exchange), el acceso a bibliotecas de programación DLL (32 y 16 bits), soporte para DirectX y MCI. Estas características son muy útiles, por ejemplo, cuando se desea crear un tutorial interactivo en cualquier aplicación de Windows.

Su paradigma de desarrollo se basa en la analogía a los libros. Cada expediente de solicitud es un libro, que comprende páginas (pantallas) en la que están dispuestos los objetos. También hay capítulos (superposiciones), que agrupan conjuntos de páginas similares, así como objetos gráficos que pueden ser

[2] SumTotal es una compañía producto de la fusión de Asymetrix, desarrolladora original de Toolbook, y Docent Software

agrupados juntos (como la característica que existe en los programas de dibujo u objetos basados en vectores, tales como CorelDraw). El contenido de ToolBook por lo tanto puede ser distribuido a los diferentes sistemas de gestión de aprendizaje (LMS) basadas en estándares abiertos. En este momento hay dos variantes ToolBook: El instructor ToolBook, la herramienta más completa, y ToolBook Asistente, una versión única con componentes predefinidos y sin acceso a la programación de scripts gratuitos, frente a los autores sin ninguna familiaridad con la programación. (13)

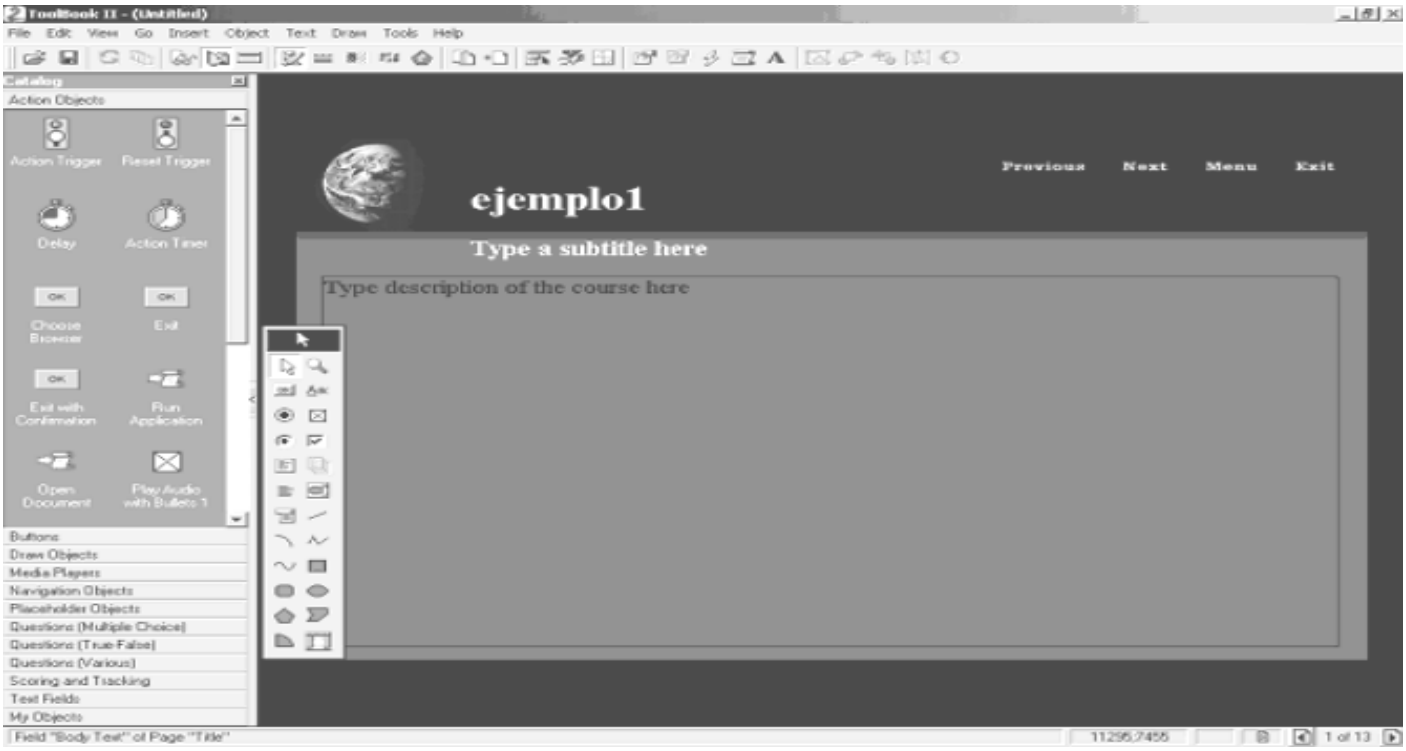


Figura 3 Área de trabajo de Asymetrix ToolBook II

Como alternativas a las herramientas privativas surgieron algunas para software libre como PowToon y Meograph. Estas son multiplataforma, consumen pocos recursos de hardware y se visualizan en la mayoría de los dispositivos móviles ya que utilizan las tecnologías HTML, CSS y JavaScript.

Powtoon es considerado una alternativa a PowerPoint y otras herramientas similares, se utiliza para la creación de presentaciones multimedia y videos. Es una opción válida para realizar presentaciones profesionales, exposiciones de productos, trabajos escolares, vídeos de animación entre otros. La ventaja

es que en este caso no es necesario descargar e instalar software, todo el proceso se realiza en el navegador web.

Es importante destacar que su uso es gratuito y que además dispone de modalidades de suscripción bastante completas. Crear presentaciones con PowToon puede llegar a ser divertido y sencillo, gracias a su intuitiva interfaz y su soporte para arrastrar y soltar diferentes elementos. Para utilizar la herramienta es necesario registrarse, proceso que puede ser bien ágil utilizando las credenciales de Facebook, Google o LinkedIn. Una vez que se haya realizado el vídeo se puede compartir de forma directa en las redes sociales y también publicarlo en la conocida plataforma YouTube. (14)

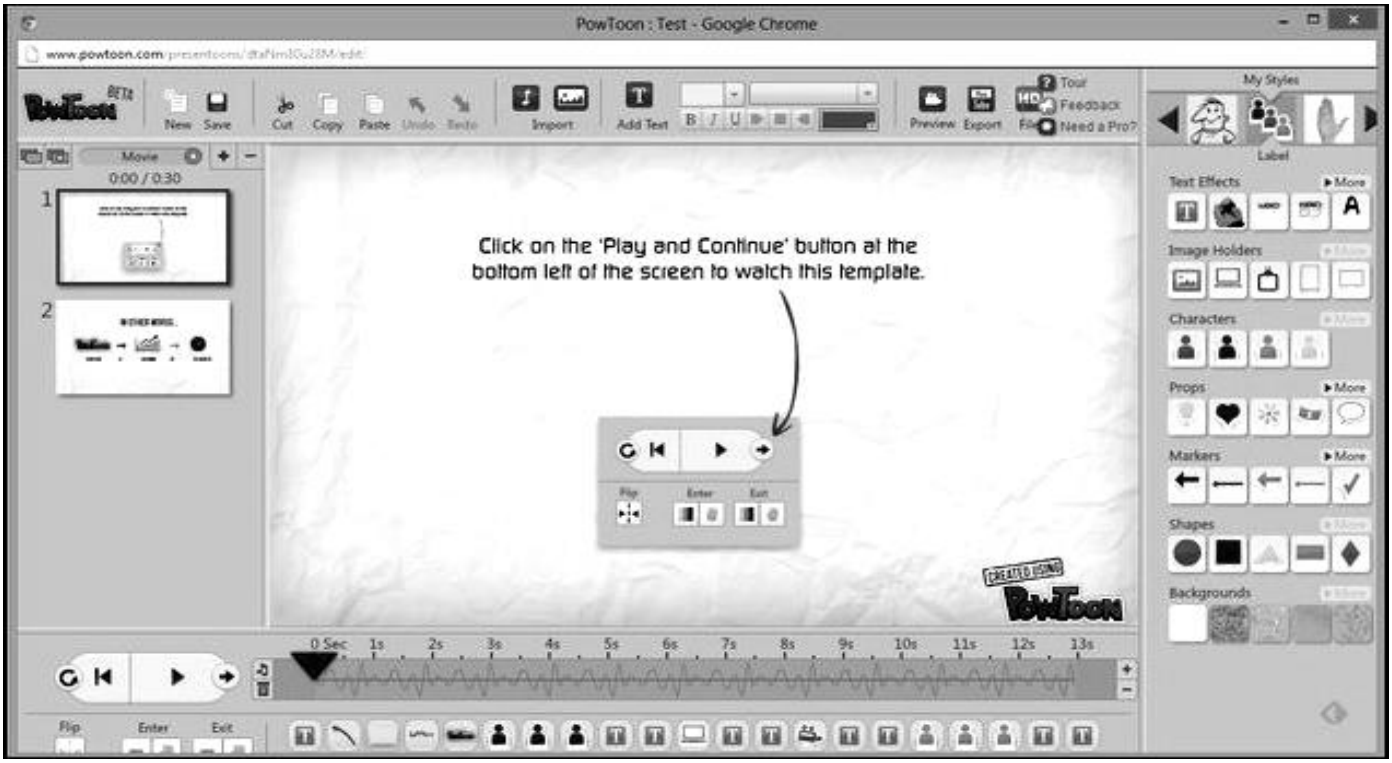


Figura 4 Área de trabajo de PowToon

Meograph es un interesante recurso online que permite crear, reproducir y compartir historias en presentaciones en las que puedes incluir videos e imágenes. La plataforma trabaja con narraciones de voz, ya sea importadas en un archivo .mp3 o grabadas desde la propia interfaz de trabajo. Esto constituye una gran ventaja para realizar tutoriales o explicar la información que se presenta. Cuenta con un área de trabajo en la que brinda diferentes opciones para agregar los contenidos a la presentación de una manera

sencilla. Esto se logra al dar la posibilidad de arrastrar los elementos al área. Otro elemento importante es que en el desarrollo de la presentación se utiliza una línea de tiempo en la que se señalan momentos clave.

El resultado se puede insertar en cualquier sitio web o compartirlo en las principales redes sociales. Está disponible de manera gratuita y solo es necesario sincronizar la plataforma con una cuenta en G+, Facebook o Twitter. (15), (16)

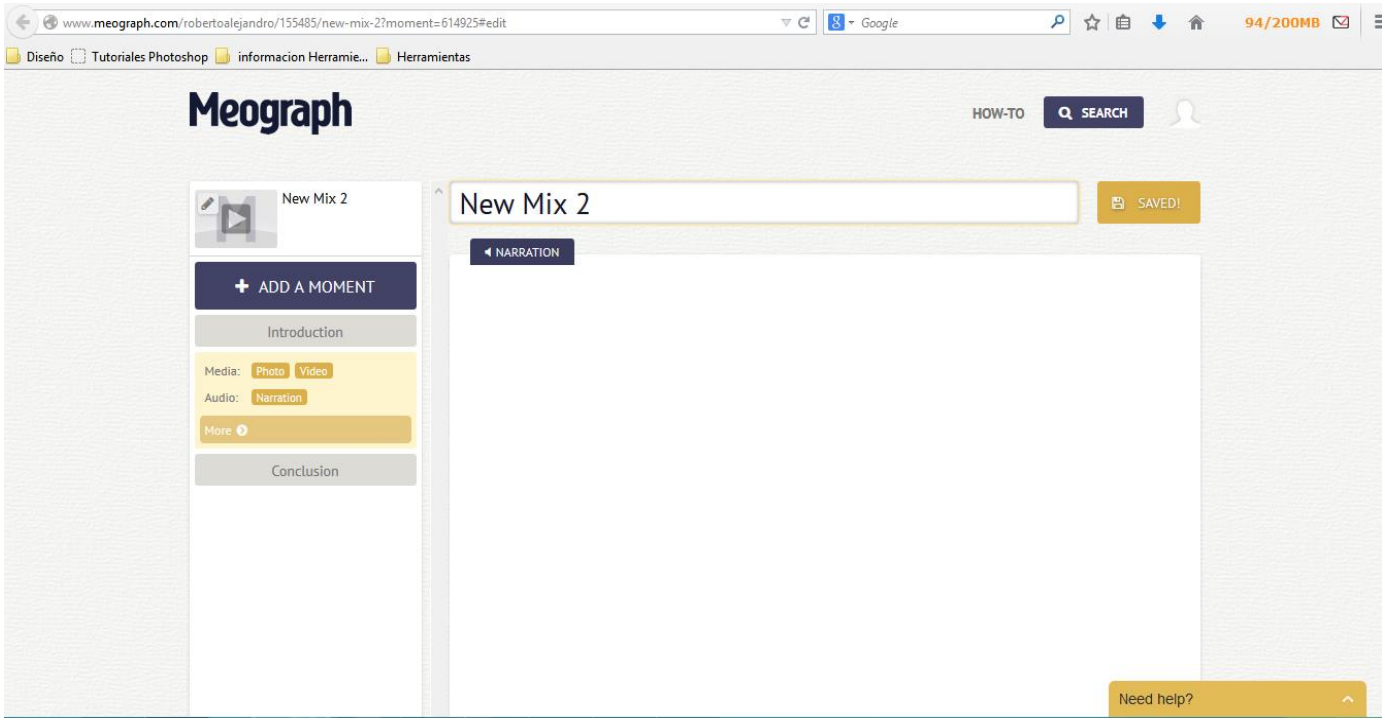


Figura 5 Área de trabajo de Meograph

Los sistemas presentados se basan en diferentes paradigmas, entre estos están los basados en tarjetas, Diagramas de flujo, Modelos de objeto, Presentaciones, Libros y Líneas de tiempo. (17)

Según las características de las aplicaciones analizadas en el epígrafe y otros datos de interés se presenta el siguiente cuadro comparativo: (13,18,19)

Tabla 1 Comparación entre herramientas similares

	Sistemas				
Características	**ToolBook II**	**Flash CS6**	**Director 6.0**	**PowToon**	**Meograph**
Fabricante	Asymetrix	ADOBE	Macromedia	Powtoon	Misha Leybovich
Precio	781.400 USD	699.000 USD	255.200 USD	Libre	Libre
Tipo de aplicación	Escritorio	Escritorio	Escritorio	Web	Web
Hipertexto	Sí	Sí	Sí	Sí	Sí
Edición dibujos	Sí	Sí	Sí	No	No
Uso de imágenes	Sí	Sí	Sí	No	No
Uso de sonido	Sí	Sí	Sí	No	No
Uso de video	Sí	Sí	Sí	No	No
Animaciones	Sí	Sí	Sí	Sí	Sí
Componentes personalizables	No	No	No	No	No
Autoguardado	Sí	Sí	Sí	Sí	Sí
Paradigma	Libro	Línea de Tiempo	Línea de Tiempo/ Presentaciones	Línea de Tiempo/ Presentaciones	Línea de Tiempo
Lenguaje script	Sí	Sí	Sí	No	No

Empaquetador de proyectos	Sí	Sí	Sí	No	No
Exporta HTML	No	Sí	No	No	No
Formato Nativo	Sí	Sí	Sí	No	No
Optimización de recursos	Sí	No	Sí	No	No

Como se puede observar a partir de la tabla anterior son los softwares propietarios los que cubren, aunque no en su totalidad, la mayoría de las características para este tipo de aplicación. Algunos de estos productos han marcado los estándares del mercado en este tipo de herramienta. Las herramientas libres estudiadas aún carecen de muchas características esenciales para tratar los diferentes tipos de media y solo están pensadas para proyectos que no sean complejos.

Otra de las conclusiones importantes es que los principales softwares propietarios analizados son herramientas de escritorio con licencias por estaciones de trabajo, mientras que las libres son aplicaciones web que pueden ser accedidas desde cualquier ordenador y que solo ponen como requisito la utilización de un navegador.

El software libre que se propone construir será una herramienta web, que debe cumplir al menos con las principales funcionalidades que se listan a continuación:

- Crear, guardar, recuperar y eliminar un proyecto realizado en la herramienta.
- Gestión de los componentes básicos (imagen, sonido, texto, video).
- Los componentes básicos y personalizables podrán ser seleccionados y se podrán realizar sobre estos las acciones de trasladar, rotar y modificar tamaño.
- Añadir animaciones de entrada y salida a los componentes.
- Visualizar la multimedia hasta donde se haya trabajado en ella.
- Se secuenciarán los componentes dentro de cada página mediante una línea de tiempo, y a su vez, estas páginas conformarán una presentación.

- Se utilizará el paradigma Línea de Tiempo/Presentación.

Además de las funcionalidades anteriormente expuestas el sistema deberá contar con la funcionalidad de autoguardado, gestionar componentes personalizables en las páginas de la multimedia y exportar a formato que pueda ser utilizado en entornos sin conexión, lo que lo diferencia del resto de las aplicaciones web de este tipo.

1.5 Metodología de desarrollo de software

Las metodologías de desarrollo de software surgen ante la necesidad de un conjunto de procedimientos, técnicas, herramientas y soporte documental. Estos elementos son ineludibles para la creación de un producto de software. De esta manera se contribuye a la obtención de un producto con calidad que cumpla con los requerimientos del cliente. Esto es posible debido a que las metodologías son las encargadas de controlar de forma transparente todo el proceso de desarrollo. Para ello propone una serie de directrices por las que se debe regir todo el equipo. Las metodologías de desarrollo se clasifican en ágiles o tradicionales. Debido a las ventajas que ofrecen, las metodologías ágiles son las más usadas en la actualidad. Ejemplo de este tipo de metodologías son XP (EXTREME PROGRAMMING), AUP (RUP Ágil), entre otras.

Para el desarrollo de la solución propuesta no se necesitó un análisis profundo de metodologías, tecnologías y herramientas ya que fueron definidas por el proyecto. La investigación se guía por las especificaciones expuestas en el documento Plan de desarrollo de software de la versión 1.0 del Marco de trabajo de desarrollo de multimedia con tecnologías libres. (20)

1.5.1 Fundamentación de la metodología a utilizar

La línea PRE utiliza una adaptación que ofrece la UCI de la metodología de desarrollo de software Proceso Unificado Ágil (AUP) para el desarrollo de sus productos. De esta manera se garantiza la uniformidad entre todos los proyectos productivos de la UCI debido a que esta metodología es la propuesta a ser aplicada en los mismos. Es por esto que se define como metodología a utilizar en el presente trabajo AUP.

Proceso Unificado Ágil (AUP)

AUP surge como una versión ágil de Proceso Unificado del Rational (RUP) por lo que hereda muchas de sus características tales como la flexibilidad. Además, propone los mismos roles y artefactos que RUP,

pero difiere en que no se necesita generar toda la documentación que se requiere en cada flujo de trabajo. No existe la necesidad de que sea adaptada y es una metodología que se ajusta y aprovecha las ventajas que brindan las metodologías ágiles.

Esta metodología determina el nivel de prioridad de cada componente según el riesgo que representen, proponiendo que el que mayor riesgo constituya la mayor prioridad y sea desarrollado en etapas tempranas del proyecto. Durante la fase de elaboración del producto se desarrollan prototipos ejecutables lo que demuestra la validez de la arquitectura para los requisitos clave del mismo y determina los riesgos técnicos. Adopta las técnicas ágiles de RUP, entre las que se encuentra (21):

- El desarrollo dirigido por pruebas.
- La modelación ágil.
- Gestión de cambios ágil.
- Refactorización de base de datos para mejorar la productividad.

La metodología AUP también adopta de RUP sus cuatro fases de desarrollo. Estas fases se llevan a cabo una a continuación de la otra:

- Inicio
- Elaboración
- Construcción
- Transición

Durante el proceso de adaptación de esta metodología a los proyectos desarrollados en la universidad, se realizó una modificación a las fases. Se mantiene la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes tres fases de AUP en una sola, que se nombra Ejecución y se agrega la fase de Cierre.

AUP define siete disciplinas (cuatro ingenieriles y tres de gestión de proyectos), las disciplinas son (21):

1. Modelo

2. Implementación
3. Prueba
4. Despliegue
5. Gestión de configuración
6. Gestión de proyectos
7. Entorno

Antes se mencionaba cómo AUP heredaba los mismos roles de RUP. La adaptación que le realiza la Universidad a la metodología propone once roles, manteniendo algunos de AUP y unificando otros para una mejor organización del equipo de desarrollo. A continuación se muestran los roles resultantes (21):

1. Jefe de proyecto
2. Planificador
3. Analista
4. Arquitecto de la información
5. Desarrollador
6. Administrador de la configuración
7. Stakeholder (cliente/proveedor de requisitos)
8. Administrador de calidad
9. Probador
10. Arquitecto de software
11. Administrador de base de datos

1.6 Herramientas y tecnologías

1.6.1 Herramientas

Visual Paradigm es una herramienta para el desarrollo de aplicaciones con lenguaje de modelado UML[3], resulta muy eficiente para un ambiente de software libre. Estos elementos se hicieron muy significativos a la hora de escoger la herramienta para el sistema a desarrollar. También se considera muy práctica para ingenieros de software, analistas y arquitectos de sistemas, pues brinda confiabilidad y estabilidad en el desarrollo orientado a objetos. Al mismo tiempo incorpora el soporte para el trabajo en equipo, permitiendo que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros, lo que hace que la colaboración sea más eficaz y fructífera. Además, hace más sencilla la traducción de requisitos en software de calidad. Visual Paradigm brinda la posibilidad de desarrollar aplicaciones de calidad de una manera rápida y fácil, incluso cuando se pretende resolver un problema bastante complejo. (22) La versión utilizada de esta es 8.0.

WebStorm es el IDE de JavaScript que está equipado para el complejo desarrollo del lado del cliente y del servidor con NodeJS. WebStorm simplifica la escritura de código gracias al autocompletamiento inteligente de código, la detección de errores dinámicamente, la navegación de gran alcance y la refactorización. El IDE proporciona soporte para JavaScript, NodeJS, HTML y CSS, así como sus sucesores modernos. Los marcos de trabajo compatibles incluyen AngularJS, React y Meteor. (23) Este se utiliza en su versión 11.0.

Bower es un módulo de NodeJS que te permite administrar los assets[4] de una forma cómoda. Elimina el problema de estar buscando online (o local en tu PC) las dependencias del proyecto (bibliotecas y marcos de trabajo). Es un sirviente, solo con ordenarle (desde la terminal) la biblioteca o marco de trabajo que se necesita, se encarga de descargar la última versión disponible. Tiene además opciones de búsqueda, actualización y eliminación de assets. (24) La versión utilizada de este es 1.7.7.

Grunt, al igual que Bower, es un módulo de NodeJS que se debe instalar global para que pueda ser ejecutado como un programa del Sistema Operativo. Con Grunt se puede automatizar casi cualquier tarea

[3] Del inglés Unified Modeled Language
[4] Ficheros para usar en una aplicación como imagen, audio y video

presente en el desarrollo de aplicaciones web (frontend[5]), pues al estar basado en plugins, se han creado muchos (incluso con distintas variantes) para las diferentes tareas que se desean acometer. (24) Este se utiliza en su versión 0.1.13.

Yeoman permite generar proyectos web con un solo comando. Estos proyectos web pueden ser de todo tipo, como Ruby, PHP y Javascript. (24) La versión utilizada de este es 1.6.0.

1.6.2 Lenguajes de desarrollo del lado del cliente

Un lenguaje de programación es un lenguaje formal diseñado para realizar procesos que pueden ser llevados a cabo por máquinas como las computadoras. También puede ser considerado como una herramienta o mecanismo que posibilita la creación de software, mediante una serie de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Posibilitan además la comunicación entre el programador y los dispositivos. Existen dos grupos en los que se clasifican los lenguajes de programación: los del lado del cliente y los del lado del servidor.

Los lenguajes del lado del servidor son aquellos lenguajes que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él. Por otro lado, los lenguajes de lado del cliente son aquellos que pueden ser "digeridos" por el navegador sin necesidad de un pretratamiento. (25)

Entre los disímiles lenguajes del lado del cliente existentes, fueron utilizados en el desarrollo de la presente investigación: HTML 5, JavaScript y CCS 3.

HTML (HyperText Markup Language**)** es el lenguaje predominante para el desarrollo de páginas webs. La traducción de sus siglas al español sería “lenguaje de marcado de hipertextos”. Este no es un lenguaje de programación como C++, Visual Basic, etc.; sino un sistema de etiquetas donde se trabaja de la siguiente forma: se abre y cierra la etiqueta, dentro de la cual va el contenido que se desee. El etiquetado HTML no presenta ningún compilador, por lo tanto, algún error de sintaxis que se presente éste no lo detectará y se visualizará en la forma como éste lo entienda. (26)

[5] Programación en el lado del cliente

HTML5 es un nuevo concepto para la construcción de sitios web y aplicaciones en una era que combina dispositivos móviles, computación en la nube y trabajos en red. Este lenguaje es una mejora de la combinación de JavaScript, HTML y CSS. HTML provee los elementos estructurales, CSS proporciona estilos a la estructura haciéndola utilizable y atractiva a la vista, y JavaScript brinda el dinamismo necesario y la posibilidad de construir aplicaciones web funcionales.HTML5 propone estándares para cada aspecto de la web y también un propósito claro para cada una de las tecnologías involucradas. (27)

JavaScript es un lenguaje de scripting[6] multiplataforma y orientado a objetos, con una sintaxis muy semejante a Java y a C. Es un lenguaje pequeño y liviano que puede conectarse a los objetos de un ambiente host y proporcionar control programático sobre ellos. Fue creado por Brendan Eich en la empresa Netscape Communications.

Es un lenguaje dinámico, las variables no necesitan ser introducidas antes de su uso y los tipos de variables se resuelven de forma dinámica durante su ejecución. Contiene una biblioteca estándar de objetos, tales como Array, Date, y Math, y un conjunto central de elementos del lenguaje, tales como operadores, estructuras de control, y sentencias. El núcleo de JavaScript puede extenderse para varios propósitos, complementándolo con objetos adicionales, por ejemplo: JavaScript del lado del cliente extiende el núcleo del lenguaje proporcionando objetos para controlar un navegador y su modelo de objetos (o DOM, por las iniciales de Document Object Model). Las extensiones del lado del cliente permiten que una aplicación coloque elementos en un formulario HTML y responda a eventos del usuario, tales como clics del ratón, ingreso de datos al formulario y navegación de páginas. (28)

Cascading Style Sheets (CSS), traducido al español como Hojas de Estilo en Cascada, fueron diseñadas y desarrolladas por la World Wide Web Consortium (W3C). Una hoja de estilos CSS es el documento que utiliza un navegador web para redefinir las propiedades de los distintos elementos y las etiquetas en el código HTML. (29) Es un lenguaje que trabaja junto con HTML para proveer estilos visuales a los elementos del documento, como tamaño, color, fondo y bordes. (27)

La **versión 3 de CSS** sigue el mismo concepto, pero con un mayor compromiso. La especificación de HTML5 fue desarrollada considerando CSS a cargo del diseño. Debido a esta consideración, la integración entre HTML y CSS es vital para el desarrollo web. Esta es la razón por la que cada vez que se

[6] Secuencia de comandos

menciona HTML5 también se está haciendo referencia a CSS3, aunque de manera oficial se trate de dos tecnologías separadas.

En la actualidad las nuevas características incorporadas en CSS3 están siendo implementadas e incluidas junto al resto de la especificación en navegadores compatibles con HTML5. (27)

1.6.3 Lenguaje de desarrollo del lado del servidor

Del lado del servidor se utilizó JavaScript como lenguaje de desarrollo. El **JavaScript (JS)** del lado del servidor puede ser un concepto nuevo para cualquiera que haya trabajado exclusivamente con JavaScript del lado del cliente, pero la idea en sí no es tan inverosímil.

JavaScript no está limitado sólo a los navegadores web, en realidad puede correr en cualquier ambiente que tenga instalado el intérprete para sus instrucciones (también conocido como motor). Esta flexibilidad ha hecho que a partir del año 2000 haya una creciente proliferación de implementaciones de JavaScript en el lado del servidor, siendo NodeJS una de las más populares.

Ejecutar JavaScript en el servidor también significa que las mismas instrucciones que se usan para construir la interfaz de una aplicación web pueden usarse para construir el núcleo del sistema, la interacción con la capa de datos y de lógica del negocio.

Además, JavaScript es un lenguaje muy poderoso y tiene una fuerte inclinación hacia paradigmas funcionales, esto lo hace muy atractivo para algunos desarrolladores. (30)

1.6.4 Tecnologías del lado del cliente

jQuery es una biblioteca rápida, pequeña y rica en funciones de JavaScript. Permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos y desarrollar animaciones dentro de la página. Proporciona un conjunto complejo de capacidades Ajax[7] mucho más simple, con un API[8] fácil de usar que funciona a través de una multitud de navegadores. Además, presenta una combinación de versatilidad y capacidad de ampliación. jQuery ha cambiado la forma en que millones de personas escriben JavaScript. (31) La versión utilizada de este es 2.2.0.

[7] JavaScript asíncrono y XML
[8] Application Programming Interface (en español, interfaz de programación de aplicaciones)

jQuery UI es una biblioteca de componentes para el marco de trabajo jQuery, que le añade un conjunto de extensiones, widgets[9] y efectos visuales para la creación de aplicaciones web. Estos componentes posibilitan crear desde aplicaciones altamente interactivas hasta insertar controles simples a un formulario. (32) Esta se utiliza en su versión 1.11.4.

AngularJS es un marco de trabajo JavaScript de código abierto desarrollado por Google. Sirve para crear aplicaciones web en lenguaje cliente con JavaScript ejecutándose con el conocido single-page applications (aplicación de una sola página) que extiende el tradicional HTML con etiquetas propias. Es extensible y funciona bien con otras bibliotecas. Cada función puede ser modificada o reemplazada para satisfacer las necesidades de flujo de trabajo de desarrollo y características únicas. (33) La versión utilizada de este es 1.4.9.

Twitter Bootstrap es una colección de herramientas de software libre para la creación de sitios y aplicaciones web. Contiene plantillas de diseño basadas en HTML y CSS con tipografías, formularios, botones, gráficos, barras de navegación y demás componentes de interfaz, así como extensiones opcionales de JavaScript. Además, ofrece la adaptación de la interfaz dependiendo del tamaño del dispositivo en el que se visualice sin que el usuario tenga que hacer nada, esto se denomina Diseño Web Adaptable. (34) Este se utiliza en su versión 3.1.1.

1.6.5 Tecnologías del lado del servidor

NodeJS es una plataforma construida para el motor JavaScript V8, el cual se ejecuta en el navegador Chrome, para construir de una forma fácil aplicaciones de red que sean rápidas y escalables. Posee un entorno de ejecución multiplataforma para el desarrollo del lado del servidor. Además, contiene un entorno de programación dirigido por eventos, basado en el lenguaje de programación ECMAScript. (35) La versión utilizada de esta es 0.12.2.

Express es un marco de trabajo para crear aplicaciones web minimalistas y flexibles que suministra un conjunto robusto de opciones, incluyendo para tecnología móvil. Este marco ofrece un amplio conjunto de métodos, que funcionan como una capa intermedia entre NodeJS y la aplicación, permitiendo que el desarrollo sea de manera rápida y sencilla. (36) Este se utiliza en su versión 4.13.4.

[9] Componentes de interfaz gráfica de usuario

Node Webkit es un motor de ejecución de aplicaciones. NodeJS proporciona un marco de trabajo basado en JavaScript con el que desarrollar una aplicación, y webkit ofrece el entorno sobre el que ejecutarlo. Dado que se dispone de versiones de Node Webkit para distintas plataformas, el código que se desarrolle en cualquiera de ellas será totalmente portable al resto. (37) La versión utilizada de este es 0.12.3.

1.7 Conclusiones del capítulo

El desarrollo del capítulo posibilitó la elaboración del marco teórico de la investigación, para un mejor entendimiento de la aplicación que se desea desarrollar. El estudio de soluciones similares permitió que se identificaran algunos elementos importantes a incluir en el entorno de desarrollo para el Marco de trabajo de creación de multimedia. Se describió la metodología de desarrollo de software AUP. Se definieron las herramientas, los lenguajes de programación y las tecnologías que serán empleadas en el desarrollo de la propuesta de solución.

CAPÍTULO II: DESCRIPCIÓN DE LA PROPUESTA DE SOLUCIÓN

2.1 Introducción

En este capítulo se describe la propuesta de solución. Se muestra un modelo conceptual para comprender los conceptos relacionados con el proceso de creación de multimedia. Se realiza el levantamiento de los requisitos funcionales y no funcionales con los que deberá cumplir la aplicación. Además, se describe la arquitectura y los patrones de diseño que fueron utilizados para la propuesta de solución. Se presenta el modelo de clases del diseño y por último se muestra una descripción de la solución.

2.2 Propuesta de solución

Para la creación de multimedia con tecnologías libres en el centro FORTES, se propone un entorno de desarrollo que permita integrar y gestionar los componentes existentes. De esta manera quedará implementado el marco de trabajo, el cual constituye una herramienta multiplataforma donde los usuarios podrán acceder de una manera sencilla e intuitiva. El entorno de desarrollo hará posible la creación de multimedia ofreciendo al usuario la selección de un área para el diseño del producto, generando así su estructura base. Además, proveerá una secuencia para la creación del mismo y brindará funcionalidades que permitan ajustar el diseño de la multimedia a los requerimientos del usuario. Será posible el exportado del producto resultante a formatos que sean ejecutables en Windows, Linux y Mac OS, además de exportar también como proyecto web HTML.

2.3 Modelo de dominio

El modelo de dominio tiene como principal objetivo identificar y describir los conceptos que se manejan en el dominio del problema, mostrándolos al usuario en forma de diagrama de clases, lo cual ayuda a definir los procesos y roles más trascendentales para el sistema a desarrollar. Contiene las clases conceptuales del mundo real y proporciona a los usuarios, desarrolladores y clientes un vocabulario común. (38)

Multimedia: sistema informático que permite la interacción. Es una combinación de medios de información, entiéndase como imagen, texto, sonido, animación y video que se ofrece al usuario a través de un dispositivo electrónico. Se construye sobre un tema que debe transmitir un conocimiento que llegue al receptor de una manera dinámica y comprensible.

Entorno de desarrollo: es un espacio optimizado que proporciona servicios al desarrollador para facilitarle el proceso de creación de multimedia. El mismo garantiza la integración y gestión de componentes personalizables.

Componente personalizable: son los componentes que se ajustan a los requerimientos de los usuarios para desarrollar una multimedia. Para el marco de trabajo de creación de multimedia con tecnologías libres se contará con los componentes personalizables:

- ✓ Galería de imágenes: el usuario puede visualizar las imágenes de acuerdo a sus requerimientos.
- ✓ Galería de videos: permite a los usuarios visualizar los videos de acuerdo a sus requerimientos.
- ✓ Glosario de términos: ayuda a entender los términos deseados por el usuario, empleados en una multimedia.
- ✓ Animaciones: es posible establecer efectos de animaciones a una imagen.
- ✓ Tema: componente que brinda la posibilidad de definir el espacio de trabajo para el desarrollo de una multimedia.
- ✓ Menú: ofrece la navegabilidad dentro de una multimedia y brinda la posibilidad de establecer un estilo para el componente.
- ✓ Mapa: permite conocer la localización y descripción de un lugar ubicado por el usuario en el mapa.

Marco de trabajo o *Framework*: es un esquema, esqueleto o patrón que contiene componentes personalizables, permitiendo a través del entorno de desarrollo la creación de multimedia con tecnologías libres.

Medios de información: diferentes tipos de contenidos que se van secuenciando de una forma dinámica creando así una multimedia. Estos medios pueden ser:

- **Texto** (en todas sus formas): puede usarse para aclarar la información gráfica o icónica. Es utilizado en el menú, en el glosario de términos y en el mapa describiendo el lugar señalado por el usuario.

- **Imágenes estáticas** (fotografías, gráficos e ilustraciones): ilustran y facilitan la comprensión de la información que se desea transmitir. Estas imágenes pueden ser visualizadas en formatos jpg, png o gif.
- **Imágenes dinámicas** (videos y animaciones): transmiten de forma visual secuencias completas de contenido. Además, ilustran un apartado de contenido con sentido propio. Los videos se pueden visualizar a través de los formatos mpg, avi, mp4 o webm.
- **Audio** (música y sonido): transmiten de forma sonora contenido que facilita la comprensión de la información clarificándola. Algunos tipos frecuentes de audio son archivos con formato wav y mp3.
- **Iconográfico**: representación de palabras, conceptos, ideas mediante dibujos o imágenes, representando lo esencial de la idea a transmitir. Es utilizado en el componente mapa para diferenciar los puntos que serán ubicados en el mismo.

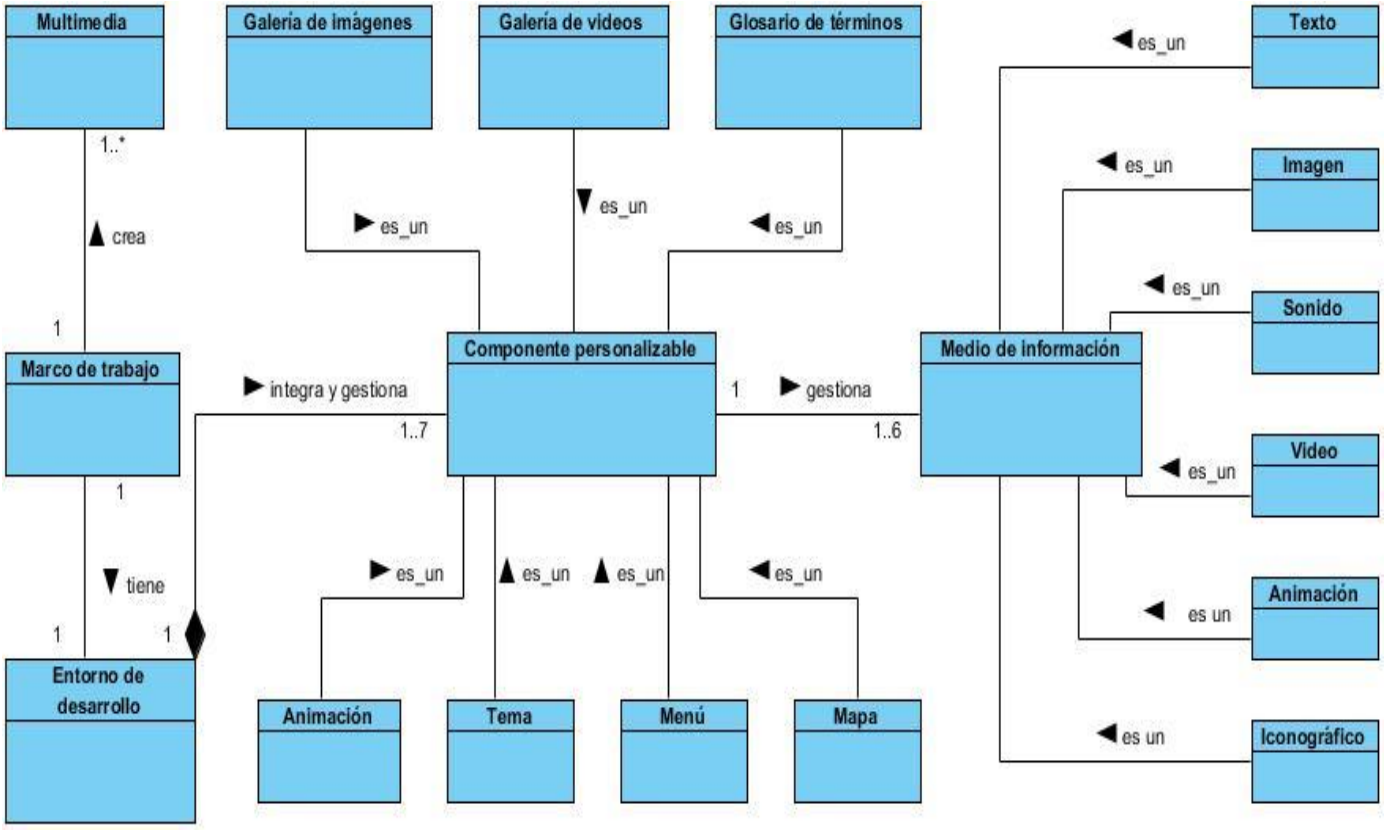


Figura 6 Diagrama de dominio de la propuesta de solución

2.4 Requisitos del sistema

La captura de requisitos es la actividad mediante la que el equipo de desarrollo de un sistema de software extrae, de cualquier fuente de información disponible, las necesidades que debe cubrir dicho sistema. (39)

2.4.1 Requisitos funcionales

Los requisitos o requerimientos funcionales (RF) son capacidades o condiciones que el sistema debe cumplir. Definición de los servicios que el sistema debe proporcionar, cómo debe reaccionar y comportarse ante situaciones particulares. Precisan las funciones que el sistema será capaz de realizar. Describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. (40)

Los requisitos funcionales que describen las capacidades que el entorno de desarrollo debe cumplir son:

RF1. Crear nuevo proyecto.

RF2. Abrir proyecto.

RF3. Importar proyecto.

RF4. Eliminar proyecto.

RF5. Autoguardar proyecto.

RF6. Guardar proyecto.

RF7. Exportar el producto.

RF8. Crear página.

RF9. Eliminar página.

RF10. Modificar página.

RF11. Guardar página.

RF12. Incluir componentes dentro de las páginas de la multimedia.

RF13. Modificar componentes dentro de las páginas de la multimedia.

RF14. Eliminar componentes dentro de las páginas de la multimedia.

RF15. Modificar propiedades de transformación de los componentes.

RF16. Controlar animaciones.

RF17. Incluir hipervínculos en la multimedia.

RF18. Visualizar multimedia.

RF19. Secuenciar los componentes.

RF20. Gestionar fotogramas.

RF21. Incluir fondo a las páginas de la multimedia.

2.4.2 Requisitos no funcionales

Los requisitos o requerimientos no funcionales (RNF) se basan en las características que de una forma u otra pueden limitar al sistema. Describen una restricción sobre el mismo que incide sobre la elección en la construcción de una solución. (40)

Durante el proceso de captura de los requisitos se detectaron los siguientes requisitos no funcionales:

Usabilidad

RNF1. El sistema debe contar con una interfaz interactiva e intuitiva para el cliente.

RNF2. El sistema debe mostrar la información de forma lógica y correctamente estructurada.

RNF3. El entorno de desarrollo debe contar con un documento que refleje la descripción de sus funcionalidades y cómo usarlas.

Hardware

RNF4. Para su correcto funcionamiento, la computadora cliente que se va a conectar al servidor para ejecutar la aplicación, deberá tener dispositivo de red, soporte de video y aditamentos para la reproducción de sonido.

RNF5. El servidor debe tener soporte para servir aplicaciones hechas en NodeJS.

Requisitos de licencia

RNF6. Las bibliotecas y tecnologías utilizadas para la confección del entorno de desarrollo deben estar bajo alguna de las siguientes licencias: General Public License (GPL) o Massachusetts Institute of Technology (MIT).

Portabilidad

RNF7. Navegadores para acceder: Chrome 32.0, Firefox 29.0, Opera 19.0, Safari 7.1, o versiones superiores a cada uno.

RNF8. El entorno debe ser independiente de la plataforma. Debe ejecutarse tanto en Microsoft Windows como en GNU/Linux y Mac OS.

Escalabilidad

RNF9. El entorno de desarrollo debe permitir la adición de nuevos componentes o la modificación de alguno de los existentes.

2.5 Historias de usuario

Según el documento Metodología de desarrollo para la Actividad productiva de la UCI existen tres formas de encapsular los requisitos (21):

- ✓ Casos de Uso del Sistema (CUS).
- ✓ Historias de usuario (HU).
- ✓ Descripción de requisitos por proceso (DRP).

Estos se encuentran agrupados en cuatro escenarios condicionados por el Modelado de negocio. En el presente trabajo se seleccionó el escenario 4 para la disciplina Requisitos, ya que en proyectos que no modelen negocio solo se puede modelar el sistema con HU. En el proyecto en cuestión se ha evaluado el negocio a informatizar y como resultado se ha obtenido un negocio muy bien definido. El cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos. Este escenario se recomienda en proyectos no muy extensos, ya que una HU no debe poseer demasiada información. (21)

A continuación, se muestran la historia de usuario Crear nuevo proyecto, para ver el resto remítase al Anexo 3.

Tabla 2 HU: Crear nuevo proyecto

Número: 1	**Nombre del requisito:** Crear nuevo proyecto.
Programador: Richard Romero Rosell	**Iteración Asignada:** 1era
Prioridad: Alta	**Tiempo Estimado:** 56 horas
Riesgo en Desarrollo: 3, 5, 8, 10, 11	**Tiempo Real:** 24 horas

Descripción:

1- Objetivo: Permitir crear un nuevo proyecto para que el usuario pueda desarrollar en el mismo.

2- Acciones para lograr el objetivo (precondiciones y datos):

Para crear un nuevo proyecto hay que:

-Tener en cuenta el dato: nombre.

-Tener en cuenta el dato: autor.

-Tener en cuenta: el formato de la imagen.

-Tener en cuenta: el tamaño de la imagen.

3- Comportamientos válidos y no válidos (flujo central y alternos):

El campo nombre es obligatorio y debe tener más de tres caracteres.

El campo autor debe tener más de tres caracteres.

La imagen debe poseer formato .png.

La imagen debe tener un tamaño menor de 200kb.

4- Flujo de la acción a realizar:

-El sistema debe permitir crear un nuevo proyecto, esta acción se puede realizar dando clic en el hipervínculo

"Nuevo Proyecto".

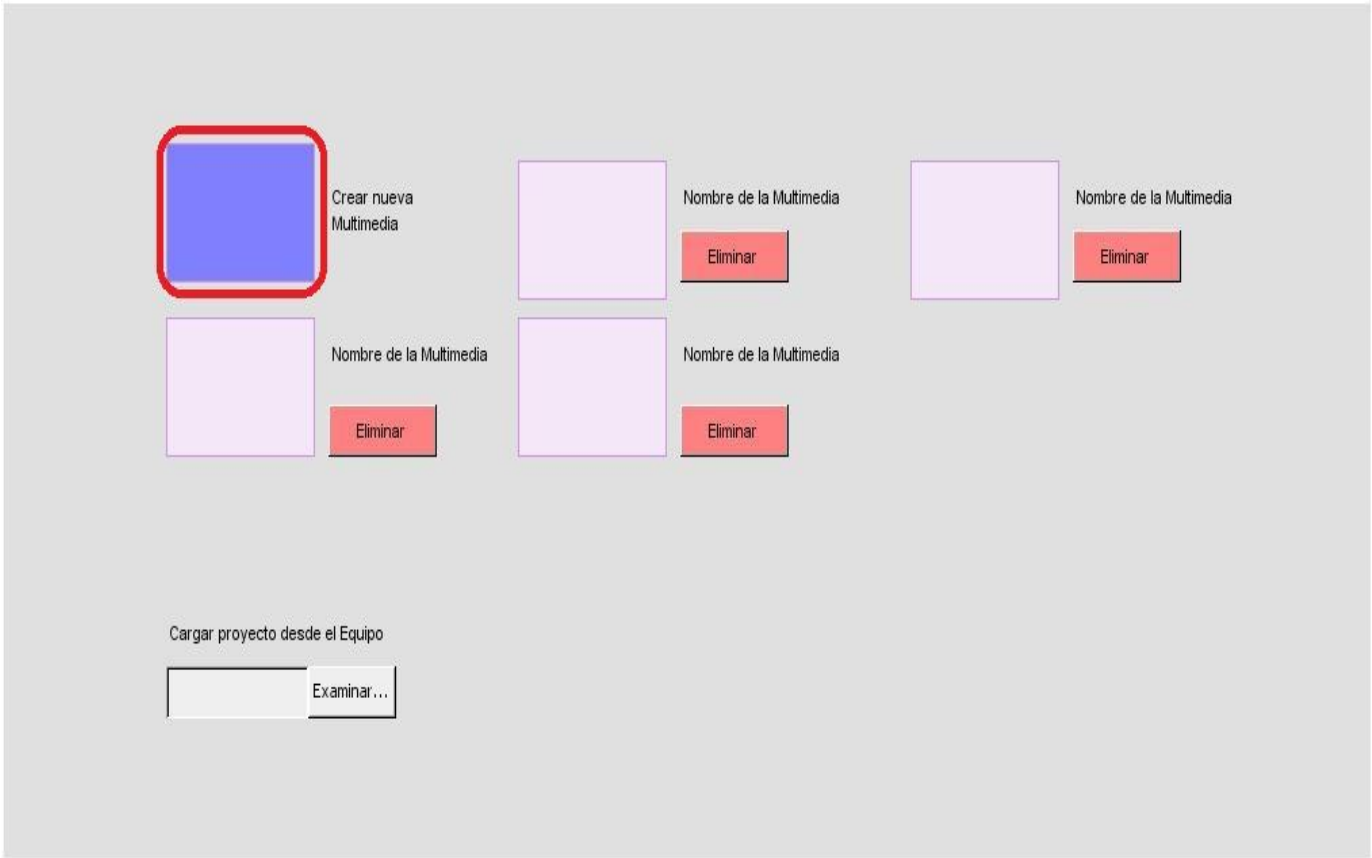
- Posteriormente se introduce el nombre deseado para el proyecto y el autor, además se puede elegir desde el ordenador una imagen que lo acompañe y hacer una descripción del mismo.

- Hasta que el campo nombre no se haya llenado no se habilita el botón Aceptar.

-Una vez que se dé clic en la opción de aceptar se procede a seleccionar una plantilla para la creación de la multimedia.

Observaciones:

Prototipo de interfaz:



2.6 Descripción de la arquitectura

La Arquitectura de Software se refiere a las estructuras de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos. (41)

Para la propuesta de solución se utilizó la tecnología AngularJS, la cual utiliza la arquitectura **Modelo Vista Controlador** (MVC) para el desarrollo de las aplicaciones web. Por tanto, está será la arquitectura que se seguirá para la implementación del entorno de desarrollo.

MVC es una de las arquitecturas de aplicaciones web más comunes. Está conformada como su nombre lo indica, por tres capas, las cuales se describen a continuación (42):

Modelo

Es la representación de la información con la cual el sistema opera, por lo tanto, gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la vista aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al modelo a través del controlador.

Vista

La renderización de código en el nivel de presentación, debería aspirar a producir la salida HTML para el usuario con poco o nada de lógica de aplicación. Todas las entradas de los usuarios se encuentran redireccionadas hacia los controladores en la lógica de la aplicación. Tiene procedimientos de actualización para recibir nuevos datos.

Controlador

El controlador (o lógica de la aplicación) toma entradas de los usuarios mediante la vista y las dirige a través de varios flujos de trabajo que llaman a los objetos del modelo de la aplicación para extraer, procesar, o almacenar información. Los controladores bien codificados, validan información centralmente en el servidor contra problemas de seguridad comunes antes de pasar la información al modelo de procesamiento y se aseguran que la salida de datos sea segura o en un formato preparado para una salida protegida por parte del código de visualización.

2.7 Patrones de diseño

Un patrón de diseño es una descripción, que se codifica en un formato estructurado, de un problema y su solución que recibe un nombre y que puede emplearse en otros contextos; en teoría, indica la manera de utilizarlo en circunstancias diversas. Los patrones no se proponen descubrir ni expresar nuevos principios de la ingeniería del software. Al contrario, intentan codificar el conocimiento, las expresiones y los principios ya existentes. (43)

2.7.1 Patrones GRASP

Se propone emplear algunos de los patrones generales de software para la asignación de responsabilidades (GRASP, por su acrónimo en inglés), ya que describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones. (43)

Dichos patrones son:

- **Patrón Experto:** Se utiliza en diseños donde un objeto de software hace ciertas operaciones, que también son realizadas en el mundo real por el objeto que representa. Se encarga de asignar una responsabilidad al experto en información, o sea, aquella clase que cuenta con la información necesaria para cumplir la responsabilidad. Este patrón favorece el bajo acoplamiento.
- **Patrón Creador:** El creador guía la asignación de responsabilidades relacionadas a la creación de objetos, una tarea muy común en sistemas orientados a objetos. El intento básico de este patrón es encontrar un creador que necesite estar conectado al objeto creado en un evento en específico. Brinda soporte al bajo acoplamiento.
- **Patrón Controlador:** Asigna la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. El controlador es un intermediario entre la interfaz de usuario y el núcleo de las clases donde reside la lógica de la aplicación. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación para aumentar la reutilización de código.
- **Alta cohesión:** Asigna una responsabilidad de modo que la cohesión siga siendo alta.
- **Bajo acoplamiento:** Asigna una responsabilidad para mantener bajo acoplamiento.

2.7.2 Patrones GOF

Los patrones GOF (del inglés Gang Of Four, en español grupo de los cuatro) son patrones de diseño que están divididos fundamentalmente en tres grandes grupos: creacionales, estructurales y de comportamiento. (42)

En el desarrollo de la propuesta de solución se aplican los patrones GOF que se detallan a continuación:

- **Singleton:** Está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella. Las situaciones más habituales de aplicación de este patrón son aquellas en las que dicha clase controla el acceso a un recurso físico único o cuando cierto tipo de datos debe estar disponible para todos los demás objetos de la aplicación. (44)

  Para ver los ejemplos tanto de los patrones GOF como los de JavaScript ver Anexo 4.

2.7.3 Patrones JavaScript

Además, fueron utilizados otros patrones propios de JavaScript que se mencionan a continuación: (46)

- **Fachada:** Simplifica los accesos a un conjunto de objetos relacionados proporcionando un objeto que todos los objetos de fuera del conjunto utilizan para comunicarse con el conjunto. Define una interface de más alto nivel que permite usar el sistema más fácil. (45)

- **Cadena de Responsabilidad:** Permite a un objeto enviar un comando sin conocer que objeto u objetos lo recibirán, evitando el acoplamiento entre el que envía y el que recibe una petición. Esto permite el paso del comando a un objeto de la cadena que es parte de una gran estructura. Cada objeto de la cadena podría manejar el comando, pasar el comando al siguiente objeto de la cadena, o las dos cosas. (45)

- **Módulo:** Su principal atractivo es que resulta extremadamente útil para conseguir código reusable y, sobre todo, modular. Su estructura básica es sencilla: se trata de una función que actúa como contenedor para un contexto de ejecución. Esto quiere decir que, en su interior se declaran una serie de variables y funciones que solo son visibles desde dentro del mismo.

- **Inyección de dependencias:** es un patrón de diseño que deriva de un patrón más genérico llamado Inversión de Control. Hace uso de la modularidad y la reutilización, las cuales siempre se deberían tener en cuenta si la aplicación va a estar dotada de mayor funcionalidad. (47)
- **Promesas:** la idea central detrás de las promesas es que una promesa representa el resultado de una operación asincrónica. (48)
- **Prototipo:** permite a un objeto crear objetos personalizados sin conocer su clase exacta o los detalles de cómo crearlos. Su tarea es dar objetos prototipo a un objeto que inicializa la creación de objetos. El objeto de creación e inicialización crea objetos mandando a sus objetos prototipo que hagan una copia de sí mismos.

2.8 Modelo de clases del diseño

El diseño de software abarca el conjunto de principios, conceptos y prácticas que conducen al desarrollo de un sistema de alta calidad o producto. (49)

Los diagramas de clases del diseño (DCD) muestran el diseño del sistema desde un punto de vista estático, a través de las clases y sus relaciones. Este es un diagrama principal de diseño y análisis para un sistema. Durante el diseño, este artefacto se elabora para tener en cuenta los detalles concretos de la implementación del sistema. También para crear el diseño conceptual de la información que se manejará en el sistema, y los componentes que se encargarán del funcionamiento y la relación entre uno y otro (50).

En la Figura 7 se muestra el diagrama de clases del diseño de la propuesta de solución:

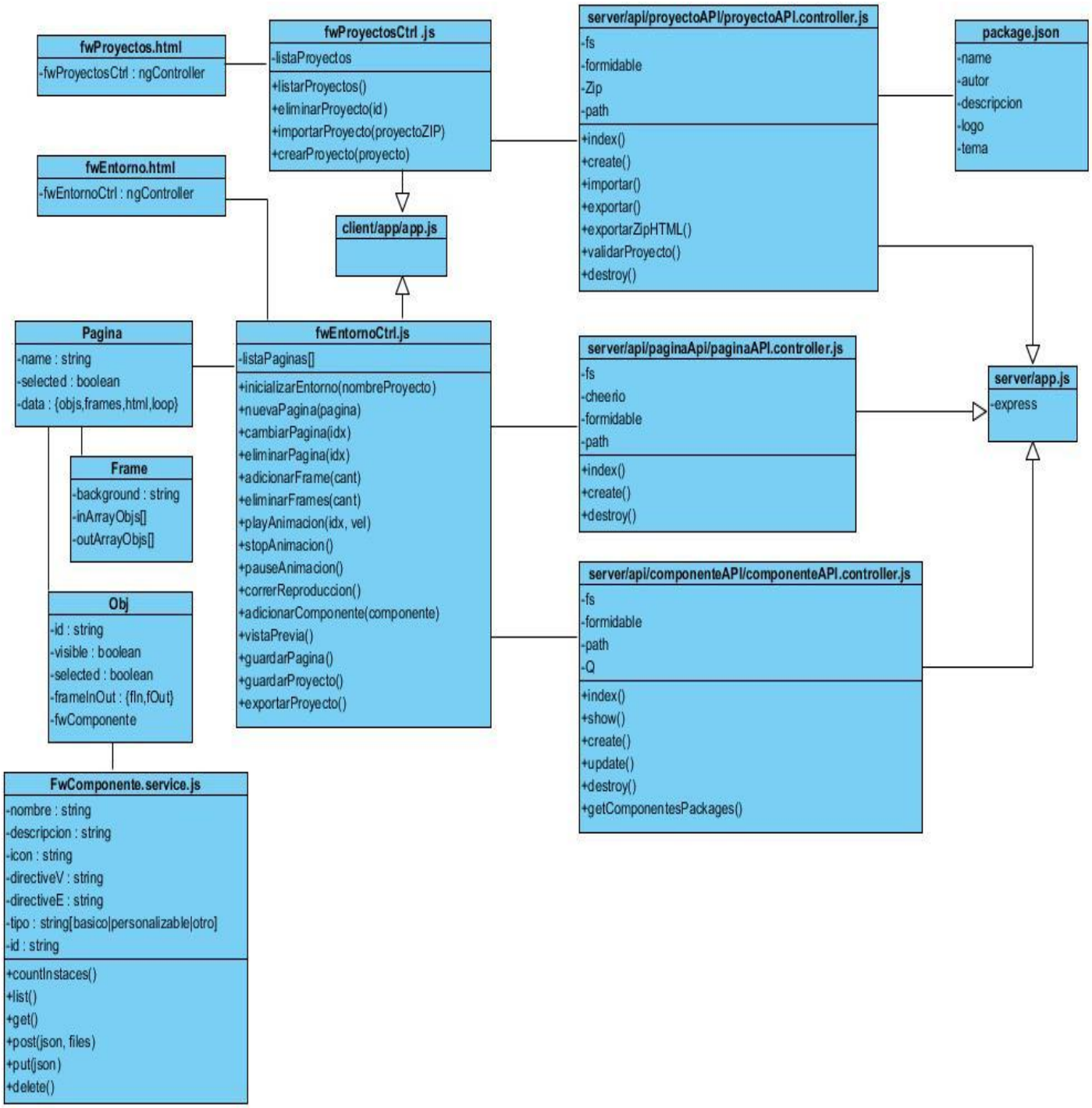

fwProyectos.html
-fwProyectosCtrl : ngController
fwProyectosCtrl .js
-listaProyectos
+listarProyectos()
+eliminarProyecto(id)
+importarProyecto(proyectoZIP)
+crearProyecto(proyecto)
server/api/proyectoAPI/proyectoAPI.controller.js
-fs
-formidable
-Zip
-path
+index()
+create()
+importar()
+exportar()
+exportarZipHTML()
+validarProyecto()
+destroy()
package.json
-name
-autor
-descripcion
-logo
-tema
fwEntorno.html
-fwEntornoCtrl : ngController
client/app/app.js
Pagina
-name : string
-selected : boolean
-data : {objs,frames,html,loop}
fwEntornoCtrl.js
-listaPaginas[]
+inicializarEntorno(nombreProyecto)
+nuevaPagina(pagina)
+cambiarPagina(idx)
+eliminarPagina(idx)
+adicionarFrame(cant)
+eliminarFrames(cant)
+playAnimacion(idx, vel)
+stopAnimacion()
+pauseAnimacion()
+correrReproduccion()
+adicionarComponente(componente)
+vistaPrevia()
+guardarPagina()
+guardarProyecto()
+exportarProyecto()
server/api/paginaApi/paginaAPI.controller.js
-fs
-cheerio
-formidable
-path
+index()
+create()
+destroy()
server/app.js
-express
Frame
-background : string
-inArrayObjs[]
-outArrayObjs[]
Obj
-id : string
-visible : boolean
-selected : boolean
-frameInOut : {fIn,fOut}
-fwComponente
server/api/componenteAPI/componenteAPI.controller.js
-fs
-formidable
-path
-Q
+index()
+show()
+create()
+update()
+destroy()
+getComponentesPackages()
FwComponente.service.js
-nombre : string
-descripcion : string
-icon : string
-directiveV : string
-directiveE : string
-tipo : string[basico|personalizable|otro]
-id : string
+countInstaces()
+list()
+get()
+post(json, files)
+put(json)
+delete()

Figura 7 DCD

2.9 Descripción de la solución

A continuación, se detalla el producto resultante del presente trabajo:

En la ventana principal aparecen listados los proyectos existentes en el servidor para que el usuario tenga la posibilidad de abrir o eliminar alguno de estos si así lo desea. Además, el usuario puede apoyarse en una opción de búsqueda que filtra los proyectos de la lista. En otro caso, también es posible subir al servidor proyectos previamente exportados en formato ZIP que se encuentran en el ordenador, mediante el botón Importar proyecto.

Para crear una multimedia en la plataforma el usuario debe hacer clic en el hipervínculo Nuevo Proyecto.

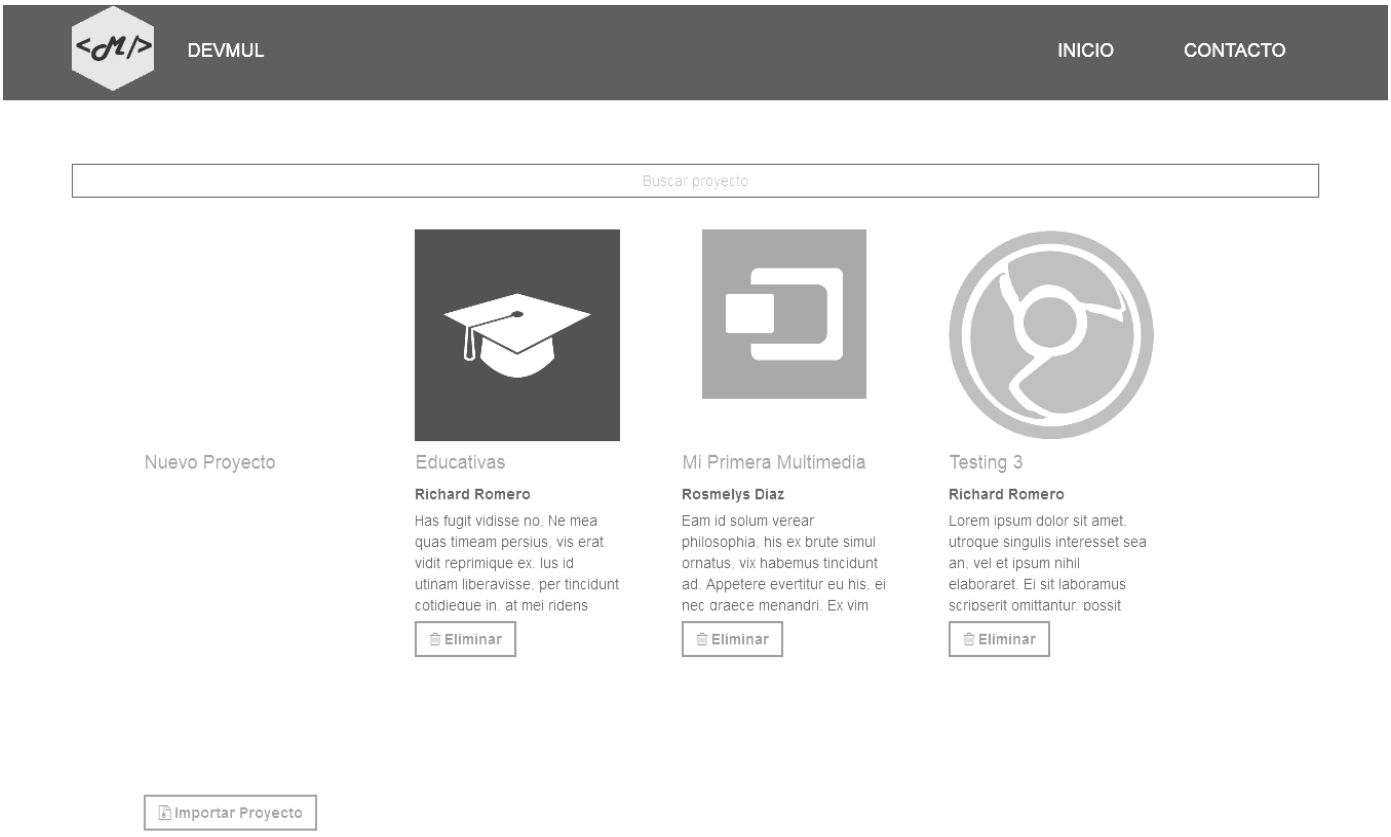


Figura 8 Crear nuevo proyecto

A continuación, aparecerá una ventana en la que introducirá el título deseado para el proyecto, además podrá incluir el autor, hacer una descripción de la multimedia y/o elegir desde el ordenador una imagen que la acompañe.

Figura 9 Formulario del nuevo proyecto

Una vez que se dé clic en la opción de aceptar se ofrecen diferentes tipos de plantillas prediseñadas, disponibles para la creación de una multimedia. Para este proyecto se ha elegido la plantilla en blanco.

Figura 10 Selección de la plantilla

Al hacer clic en la plantilla seleccionada automáticamente se mostrará una ventana emergente donde el usuario deberá introducir el nombre de la primera página que debe crear para poder trabajar sobre ella. Se le debe poner nombres a las páginas con el objetivo de poder acceder a ellas mediante el menú o desde un hipervínculo en otra página.

Figura 11 Crear nueva página

Al aceptar se encontrará con un espacio de trabajo donde se tendrán todos los paneles y opciones necesarios para crear, en este caso, una multimedia sobre las redes sociales.

Figura 12 Espacio de trabajo

En la parte superior del entorno se encuentran el menú desplegable Páginas mediante el que se podrá acceder a las páginas que hayan sido creadas, además de brindar las opciones de crear una nueva página y seleccionar cuál es la primera página de la multimedia. Dándole clic al botón Nueva página se mostrará el formulario para adicionar una nueva a la lista, y al botón se eliminará. La primera página de la multimedia podrá ser seleccionada mediante Página inicio. En este caso solo se ha creado una página, la cual ha sido definida como la primera página de la multimedia En el panel menú se encuentran los botones para guardar página Guardar , guardar proyecto Guardar todo, exportar Exportar y vista previa Vista Previa.

A la derecha se encuentran los componentes básicos y personalizables (excepto el componente Menú) que se pueden utilizar, cada uno se podrá arrastrar y soltar en el área de trabajo para trabajar con ellos.

En la parte inferior se tiene la línea de tiempo, que a la vez cuenta con los botones detener y reproducir , el primero detiene la reproducción regresando al principio, y el segundo la reproduce o pausa en

cualquier punto de la línea de tiempo. El botón  permite determinar que la reproducción de la página se comporte como un ciclo. Seguidamente se encuentra el componente personalizable Menú  , sobre el que se debe dar clic para acceder a su panel de configuración. Se brinda la funcionalidad de seleccionar un fondo para cada página  , además de la opción de adicionar o eliminar una cantidad de fotogramas introducida por el usuario  . Debajo de ésta línea de tiempo se listarán los componentes presentes en cada página.

Para crear la multimedia lo primero que se ha hecho ha sido añadir y modificar los componentes básicos Texto, Imagen y Audio. Fue cargado desde el ordenador una imagen que representa el impacto de las redes sociales en la actualidad y un fondo musical que lo acompaña, además se escribió el texto “Impacto de las redes sociales en la actualidad”.

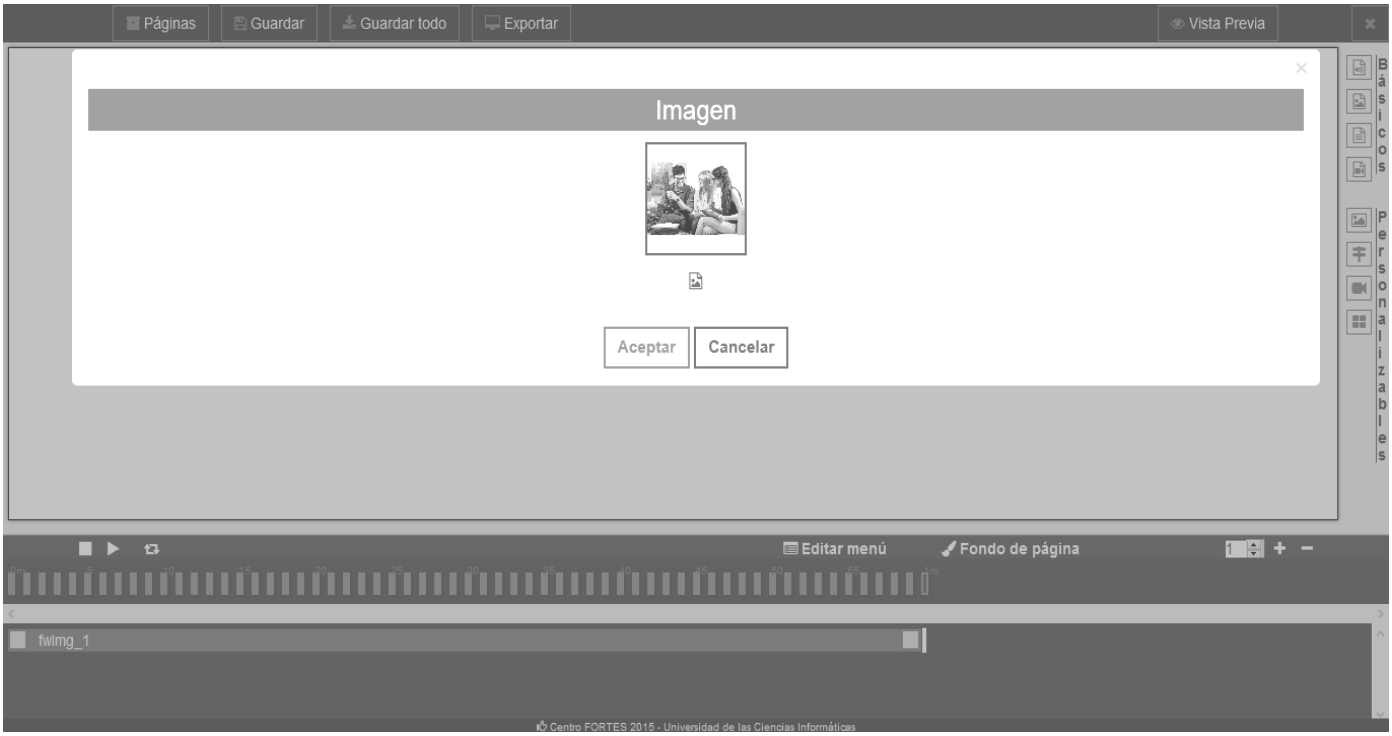


Figura 13 Edición del componente Imagen

También fueron editadas las propiedades de transformación de los componentes hasta obtener el siguiente resultado:

Figura 14 Trabajo con componentes básicos

Posteriormente se ha añadido el componente personalizable Galería de imágenes y se ha editado.

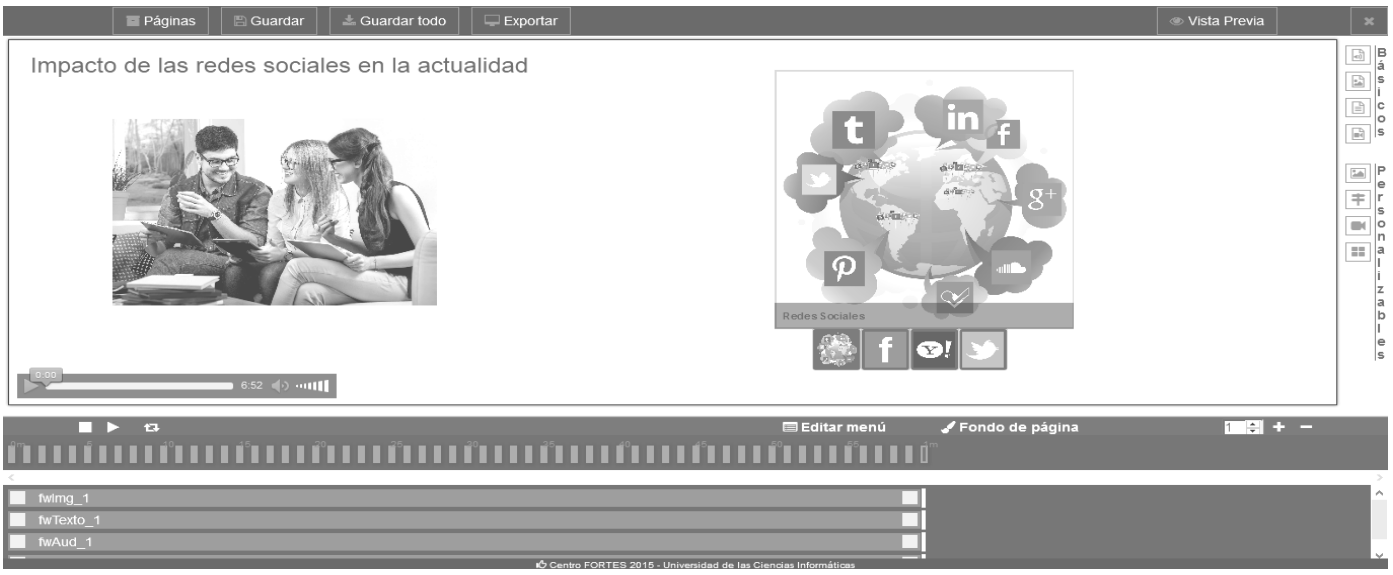


Figura 15 Trabajo con componentes personalizables

Se secuenciaron los componentes para que entren y salgan en los segundos (fotogramas) correspondientes. En la imagen siguiente se puede ver que el componente Texto, entrará en el segundo 7 de esa página y saldrá en el segundo 17 de la misma.

Figura 16 Secuencia de los componentes

Las animaciones de entrada y salida de cada componente vienen representadas por los iconos que se sitúan al principio y al final de la barra que marca el tiempo de ese componente en la página. Para elegir el tipo de animación que se le quiere asignar a los componentes, se hace clic sobre esos iconos y aparecerá una ventana emergente donde se podrá elegir qué efecto tendrá y la duración de mismo.

Figura 17 Selección de las animaciones

El sistema brinda la posibilidad de añadir un fondo a cada página, realizando una combinación entre color e imagen.

Figura 18 Selección de un fondo para la página

Una vez se dé por finalizado el proyecto, se podrá guardar y/o exportar. La primera opción permite guardar el proyecto en el servidor y la segunda brinda la posibilidad de escoger entre exportar como proyecto web HTML o exportarlo para alguna de las plataformas Windows, Linux o Mac OS, además de elegir entre las opciones de pantalla completa o definir una resolución registrada por el usuario. Estas acciones las encontramos en el menú de arriba de la pantalla de trabajo.

Se ha decido exportar el producto final como un ejecutable en el sistema operativo Windows (32 bits), para lograrlo se debe guardar primeramente todo el proyecto y una vez que se selecciona la opción exportar se muestra la siguiente ventana emergente donde se definen las preferencias del usuario:

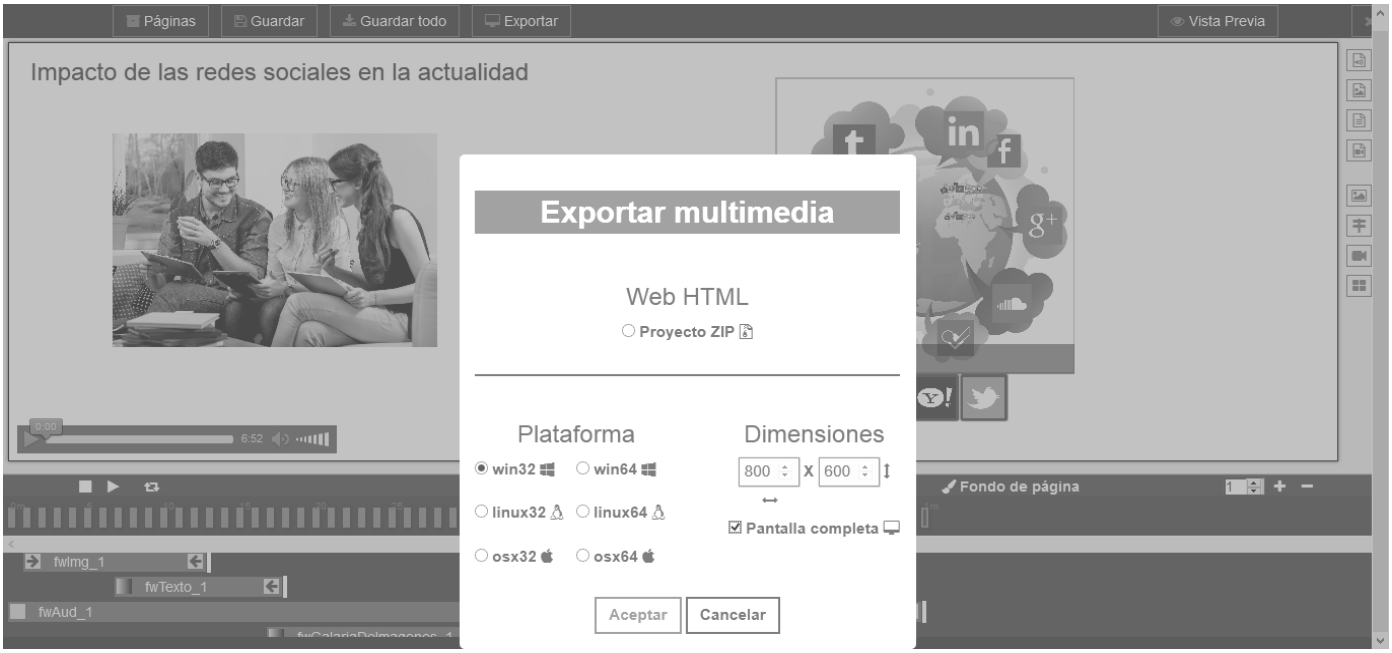


Figura 19 Exportar proyecto

2.10 Conclusiones del capítulo

Para la realización del análisis y el diseño se tomó como guía la propuesta de AUP para el desarrollo de software en la UCI. La elaboración de este capítulo ha logrado un mejor entendimiento del software a desarrollar, así como las condiciones que debe necesarias con las que debe contar para cumplir con los requisitos establecidos por el cliente. Se muestra un listado de requisitos funcionales y no funcionales que debe cumplir el entorno de desarrollo. La arquitectura y patrones de diseño seleccionados constituyeron una guía para la implementación de la propuesta de solución. Además, fueron generados artefactos, tales como modelo conceptual, historias de usuario y diagramas de clases del diseño. Por último se mostró una descripción del sistema con el objetivo de lograr una visión más clara del mismo.

CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS

3.1 Introducción

En el presente capítulo se exponen particularidades de la implementación de la propuesta de solución, con el objetivo de obtener un producto final que esté en correspondencia con los requisitos definidos por el cliente. Se muestran el diagrama de componentes y el de despliegue del entorno de desarrollo, que brindará un mejor entendimiento del mismo. Se define la estructura de la propuesta de solución y los estándares de codificación que debe cumplir el equipo de desarrollo. Además, se hace alusión a la fase de prueba, presentando las interfaces de pruebas creadas para comprobar su funcionamiento y calidad.

3.2 Implementación

Según la metodología establecida para este trabajo, AUP, en la implementación a partir de los resultados del análisis y diseño se construye el sistema. El objetivo de esta disciplina es transformar su modelo en código ejecutable y realizar un nivel básico de las pruebas, en particular, la unidad de pruebas. (21)

3.2.1 Diagrama de componentes

Los diagramas de componentes se utilizan para modelar la vista estática de un sistema. Muestran la organización y las dependencias lógicas entre un conjunto de componentes de software, sean componentes de código fuente, librerías, binarios o ejecutables. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema. Se representa como un grafo de componentes de software unidos por medio de relaciones de dependencia. (49)

A continuación, se presenta el diagrama de componentes del entorno de desarrollo:

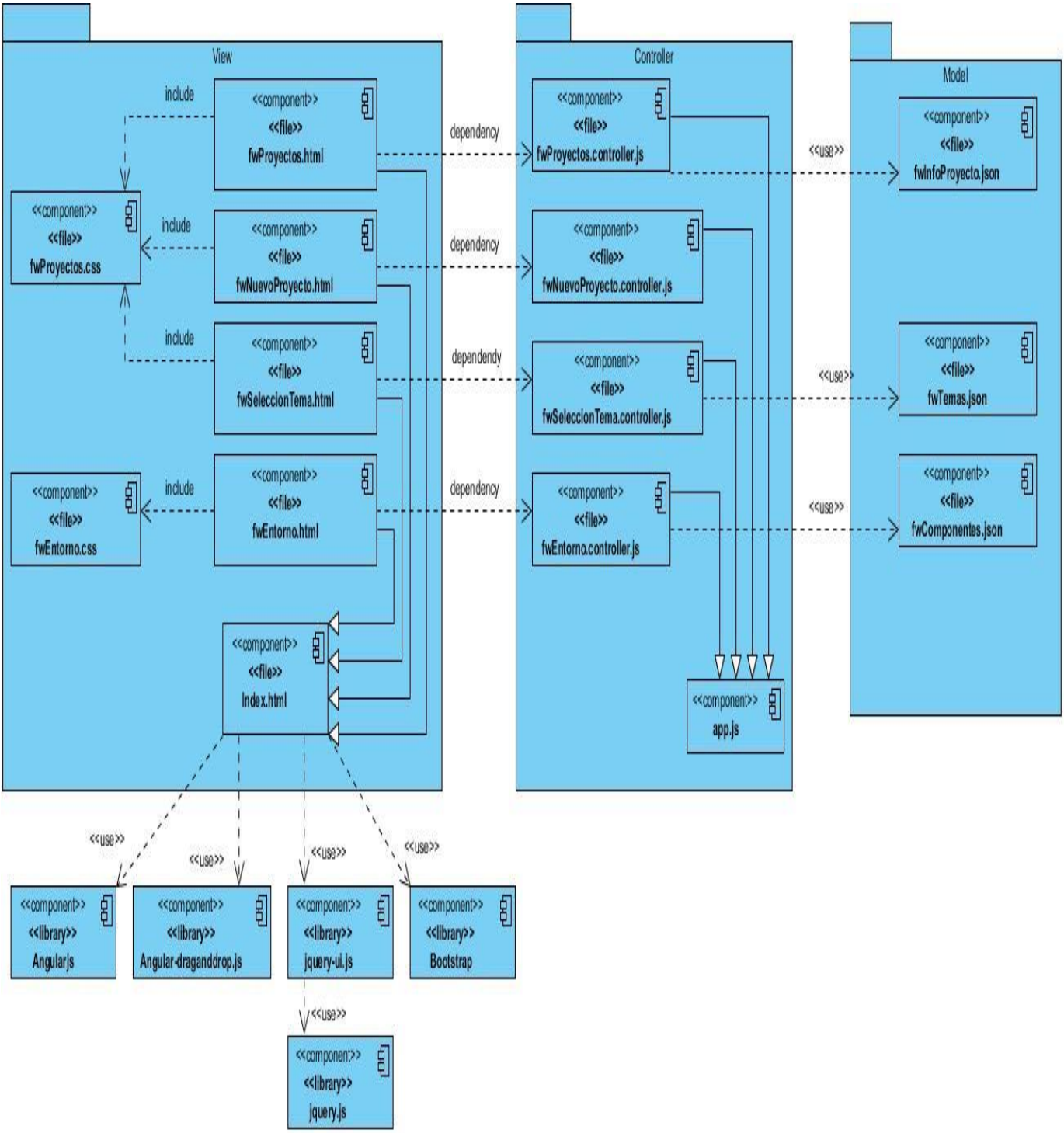


Figura 20 Diagrama de componentes

3.2.2 Diagrama de despliegue

Para comprender como se ejecutará a nivel de hardware el sistema desarrollado y tener una visión clara de la estructura del sistema en ejecución y las relaciones entre los componentes que interactúan en el mismo, se realiza el diagrama de despliegue. Dicho diagrama tiene como objetivo reflejar lo mencionado anteriormente, representando la disposición de las instancias de los componentes de ejecución, en instancias de nodos conectados por enlaces de comunicación.

Los servicios desarrollados se encuentran representados en el siguiente diagrama:

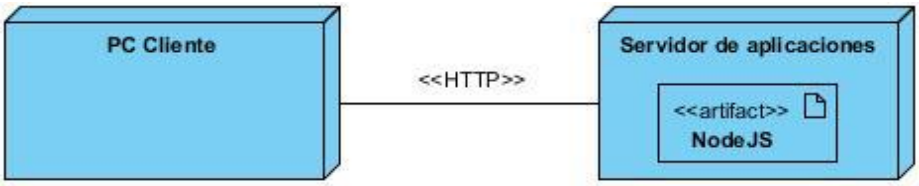


Figura 21 Diagrama de despliegue

3.2.3 Estructura de la propuesta de solución

Para el desarrollo de la propuesta de solución se utilizó el patrón Modelo Vista Controlador (MVC). Este patrón de arquitectura divide una aplicación interactiva en tres componentes distintos, separando los datos de una aplicación, la interfaz de usuario, y la lógica de control. La utilización de esta arquitectura aporta ventajas como la separación de responsabilidades, la reusabilidad de componentes y una mejor organización al código. En el Anexo 4 se muestra la estructura de carpetas de la propuesta de solución.

Para que un proyecto pueda ser importado a la plataforma debe contar con la siguiente estructura:

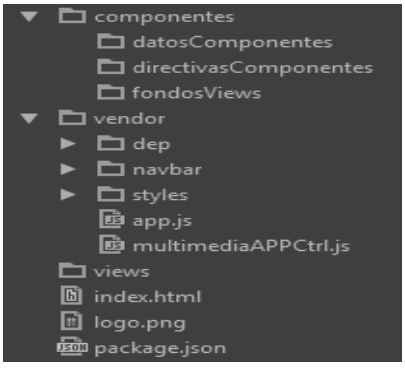


Figura 22 Estructura del proyecto para ser importado a la plataforma

3.2.4 Estándares de codificación

Se definen estándares de codificación para la implementación del entorno de desarrollo para lograr que el código del mismo sea homogéneo, permitiendo una mejor comprensión y facilitando futuras modificaciones.

Esta labor es decisiva para poder plantear con éxito la propiedad colectiva del código, que sería impensable sin una codificación basada en estándares que haga que todo el mundo se sienta cómodo con el código escrito por cualquier otro miembro del equipo. (51)

Para la implementación de la aplicación se tuvieron en cuenta los estándares de codificación definidos en el documento “Estándares de codificación Proyecto Marco de trabajo para el desarrollo de multimedia con tecnologías libres”. (52)

Los estándares de codificación utilizados fueron:

- Declaración del tipo de documento en la primera línea el documento.
- Se utilizó lowerCamelCase para el nombre de las etiquetas, variables y funciones:

El término lowerCamelCase describe una palabra compuesta en la cual la primera palabra siempre empieza con minúscula y el resto de las palabras con mayúscula. (53) (Ejemplo: listaTransiciones).

- Uso de espacios para mejorar la legibilidad.
- Uso de asignaciones, declaraciones, funciones.
- Chequeo de tipos.
- Evaluación condicional.
- Estilo práctico.
- Se hicieron retornos tempranos para mejorar la legibilidad del código y sin tener impactos significativos en el rendimiento.
- Comentarios:

* El estilo de JSDoc es bueno y aceptable (Closure Compiler type hints++)

* Comentar en la línea que está encima del código del que se está hablando

* Comentarios en múltiples líneas.

En la siguiente imagen se muestra un fragmento de código donde se aplicaron algunos de estos estándares:

```
(function inicializarEntorno(nombreP) {
  cfpLoadingBar.start();
  fwConeccionServidor.get(nombreP + '/package.json')
    .then(function (res) {
      $rootScope.nombreProyecto = nombreP;
      $rootScope._Proyecto = res.data;
    }, function (err) {
      console.warn(err);
    });
  fwConeccionServidor.get('/api/archivoCompBasicoAPIs/' + nombreP)
    .then(function (res) {
      $rootScope.compBasicosSource = res.data;
    });
  fwConeccionServidor.get('/api/paginaAPIs/' + nombreP)
    .then(function (res) {
      res.data.forEach(function (el) {
        adicionarPaginaALista(el);
      });
      if ($scope.listaPaginas.length === 0) {
        $scope.formNuevaPagina();
      } else {
        $scope.cambiarPagina(0);
      }
    }, function () {
      console.warn(err);
    });
})($routeParams.nombreProyecto);
```

Figura 23 Fragmento de código con la utilización de estándares de codificación

3.3 Pruebas de software

En muchos sentidos, la prueba es un proceso autónomo, y el número de tipos diferentes de pruebas varía tanto como los diferentes enfoques de desarrollo. Durante muchos años, la única defensa contra los errores de programación fueron el diseño cuidadoso y la inteligencia natural del programador. Ahora estamos en la era en que las técnicas modernas de diseño ayudan a reducir el número de errores iniciales

inherentes al código. De manera similar, diferentes métodos de prueba están empezando a agruparse en varios métodos y filosofías distintos. (54) La metodología AUP plantea que el objetivo de esta disciplina consiste en realizar una evaluación objetiva para garantizar la calidad, lo que incluye la búsqueda de defectos, validar que el sistema funciona tal como está establecido, y verificar que se cumplan los requisitos.

Para llevar a cabo el proceso de pruebas existen 4 niveles que pueden combinarse o reorganizarse en función de la naturaleza del proyecto o de la arquitectura del sistema: pruebas unitarias, pruebas de integración, pruebas de sistema y pruebas de aceptación. Al sistema propuesto se le aplicaron las pruebas definidas en los siguientes subepígrafes.

3.3.1 Pruebas unitarias

Las pruebas unitarias tienen como objetivo verificar la funcionalidad y estructura de cada componente individualmente una vez que ha sido codificado. Las pruebas iniciales constituyen un sistema, y todas las demás pruebas deben apoyarse sobre ellas. Existen dos enfoques principales para el diseño de los casos de pruebas: el enfoque estructural o de caja blanca y el enfoque funcional o de caja negra. (55)

Las pruebas unitarias pueden incluir pruebas de funcionalidad y características no funcionales específicas, tales como el comportamiento de recursos (por ejemplo, la búsqueda de filtraciones de memoria) o pruebas de robustez, además de pruebas estructurales (por ejemplo, cobertura de decisión).

Para la automatización de este tipo de pruebas realizadas al entorno de desarrollo fue utilizado el marco de trabajo Karma. El objetivo principal de esta herramienta es brindar un entorno de pruebas productivas para los desarrolladores, que minimice la cantidad de configuraciones necesarias. Propicia un espacio donde los programadores pueden simplemente escribir el código en marcos de trabajo de pruebas JavaScript y obtener información inmediata de estas. (57)

El marco de trabajo seleccionado para el desarrollo de estas pruebas fue Jasmine ya que no depende de ningún otro marco de trabajo de este lenguaje, no requiere un DOM y tiene una sintaxis limpia y evidente por lo que se pueden realizar fácilmente las pruebas. (58)

En la Figura 24 se presenta un ejemplo de las pruebas unitarias realizadas al cliente, con la herramienta Jasmine, al concluir la tercera iteración:

fwProyectos.controller.spec.js

```
'use strict';

describe('Controller: FwProyectosCtrl', function () {

  // load the controller's module
  beforeEach(module('mdmMeanApp'));

  var FwProyectosCtrl, scope, fwConeccionServidorMock, httpMock;

  // Initialize the controller and a mock scope
  beforeEach(inject(function ($controller, $rootScope, fwConeccionServidor, $httpBackend) {
    scope = $rootScope.$new();
    FwProyectosCtrl = $controller('FwProyectosCtrl', {
      $scope: scope
    });
    fwConeccionServidorMock = fwConeccionServidor;
    httpMock = $httpBackend;

    httpMock.when('GET', '/api/proyectoAPIs').respond([1, 2, 3])
  }));

  it('no debe tener ningun proyecto al comenzar', function () {
    expect(scope.listaProyectos.length).toBe(0);
  });

  it('debe listar los proyectos del servidor', function () {
    fwConeccionServidorMock.get('/api/proyectoAPIs')
      .then(function (data) {
        scope.listaProyectos = data;
      });
    httpMock.flush();
    expect(scope.listaProyectos.length).toBe(3);
  });

  it('debe adicionar un proyecto y luego eliminar un proyecto', function () {
    scope.listaProyectos.push({
      "nombre": 'Nuevo Proyecto',
      "autor": 'yo',
      "descripcion": 'Proyecto de Prueba',
      "tema": 'Plantilla 1',
      "logo": "logo.jpg"
    });
    expect(scope.listaProyectos.length).toBe(1);
    scope.eliminarProyectoDeLista('Nuevo Proyecto');
    expect(scope.listaProyectos.length).toBe(0);
  });
});
```

Figura 24 Prueba unitaria realizada al controlador fwProyectos.js

Por otro lado, para las pruebas unitarias realizadas al servidor fue utilizado el marco de trabajo Mocha ya que presenta características de alto nivel aceptables, es altamente configurable para adaptarse a las necesidades del probador y se puede ejecutar en el navegador. Node.js lo utiliza para ejecutar las pruebas,

y permite al desarrollador llevar a cabo pruebas tanto síncronas como asíncronas. La principal interacción con Mocha se realiza mediante la herramienta de línea de comandos proporcionada, lo que le permite configurar la forma en que se ejecutan las pruebas. (59)

A continuación, se muestran las pruebas realizadas con Mocha al servidor:

```
'use strict';
var proxyquire = require('proxyquire').noPreserveCache();
var proyectoAPICtrlStub = {
  index: 'proyectoAPICtrl.index',
  create: 'proyectoAPICtrl.create',
  destroy: 'proyectoAPICtrl.destroy',
  importar: 'proyectoAPICtrl.importar',
  exportar: 'proyectoAPICtrl.exportar',
  exportarZipHTML: 'proyectoAPICtrl.exportarZipHTML'
};
var routerStub = {
  get: sinon.spy(),
  post: sinon.spy(),
  delete: sinon.spy()
};
// require the index with our stubbed out modules
var proyectoAPIIndex = proxyquire('./index.js', {
  'express': {
    Router: function () {
      return routerStub;
    }
  },
  './proyectoAPI.controller': proyectoAPICtrlStub
});
describe('ProyectoAPI API Router:', function () {
  it('should return an express router instance', function () {
    proyectoAPIIndex.should.equal(routerStub);
  });
  describe('GET /api/proyectoAPIs', function () {
    it('should route to proyectoAPI.controller.index', function () {
      routerStub.get
        .withArgs('/', 'proyectoAPICtrl.index')
        .should.have.been.calledOnce;
    });
  });
  describe('POST /api/proyectoAPIs', function () {
    it('should route to proyectoAPI.controller.create', function () {
      routerStub.post
        .withArgs('/', 'proyectoAPICtrl.create')
        .should.have.been.calledOnce;
    });
  });
  describe('DELETE /api/proyectoAPIs/:id', function () {
    it('should route to proyectoAPI.controller.destroy', function () {
      routerStub.delete
        .withArgs('/:id', 'proyectoAPICtrl.destroy')
        .should.have.been.calledOnce;
    });
  });
  describe('POST /api/proyectoAPIs/importar', function () {
    it('should route to proyectoAPI.controller.importar', function () {
      routerStub.post
        .withArgs('/importar', 'proyectoAPICtrl.importar')
        .should.have.been.calledOnce;
    });
  });
  describe('POST /api/proyectoAPIs/exportar', function () {
```

Figura 25 Pruebas unitarias realizadas a las API RESTful

3.3.1.1 Resultado de las pruebas unitarias

Las pruebas unitarias fueron realizadas en tres iteraciones a todos los controladores, directivas, servicios y filtros que se utilizan en el cliente y a las API RESTful del servidor. Cada iteración fue arrojando errores

que se corrigieron en el instante permitiendo que al concluir las tres iteraciones se obtuviera un producto de calidad libre de errores, lo que se puede corroborar en las siguientes figuras:

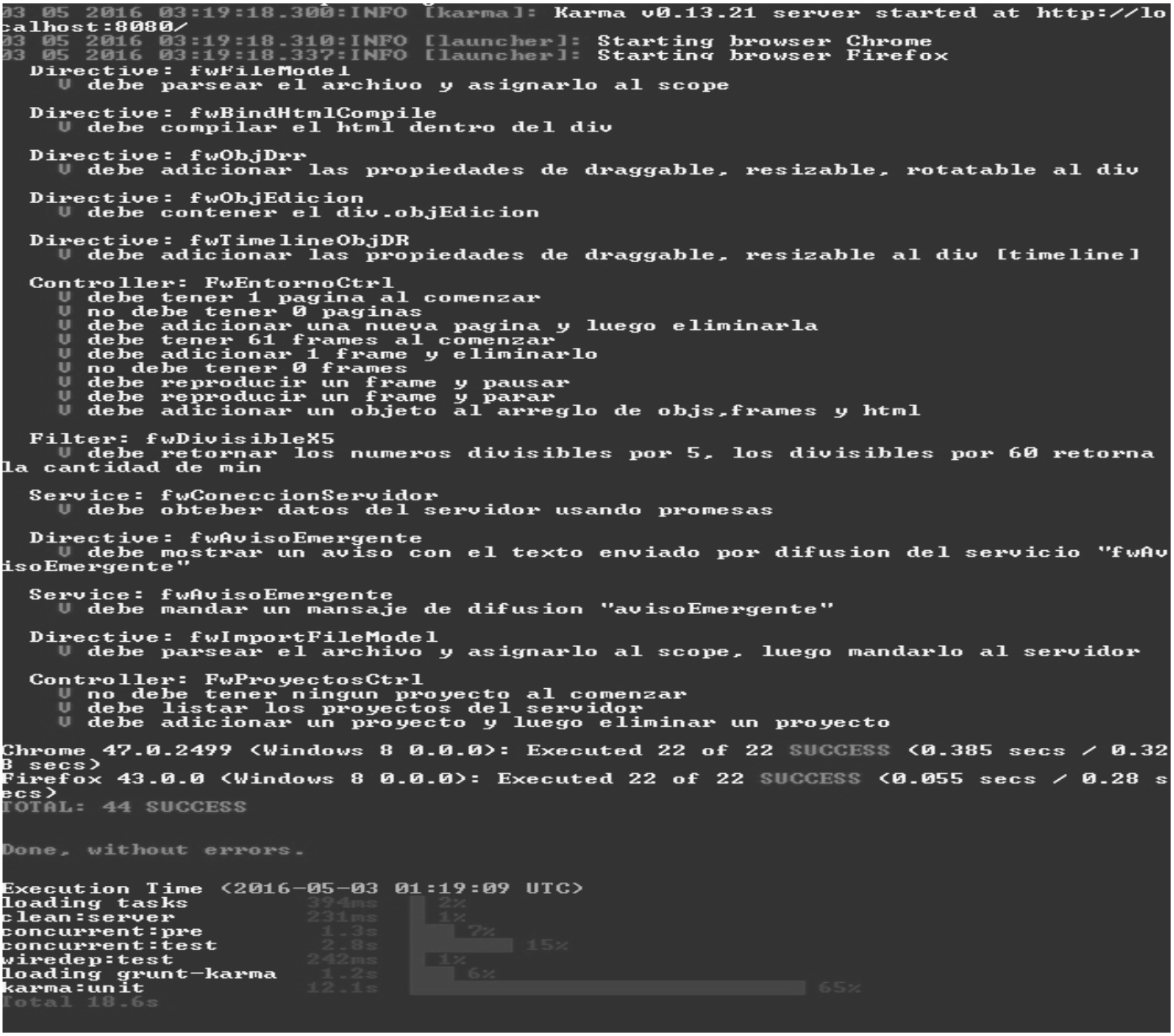


Figura 26 Resultados de las pruebas unitarias del lado del cliente con Jasmine

```
Running "test:server" (test) task

Running "env:all" (env) task

Running "env:test" (env) task

Running "mochaTest:unit" (mochaTest) task


  ComponenteAPI API Router:
    √ should return an express router instance
    GET /api/componenteAPIs
      √ should route to componenteAPI.controller.getComponentesPackages
    GET /api/componenteAPIs/:nombreProyecto/:tComponente
      √ should route to componenteAPI.controller.index
    GET /api/componenteAPIs/:nombreProyecto/:tComponente/:id/:dataType
      √ should route to componenteAPI.controller.show
    POST /api/componenteAPIs/:nombreProyecto/:tComponente/:id/:dataType
      √ should route to componenteAPI.controller.create
    PUT /api/componenteAPIs/:nombreProyecto/:tComponente/:id/:dataType
      √ should route to componenteAPI.controller.update
    DELETE /api/componenteAPIs/:nombreProyecto/:tComponente/:id
      √ should route to componenteAPI.controller.destroy

  PaginaAPI API Router:
    √ should return an express router instance
    GET /api/paginaAPIs
      √ should route to paginaAPI.controller.index
    POST /api/paginaAPIs
      √ should route to paginaAPI.controller.create
    DELETE /api/paginaAPIs/:nombreProyecto/:id
      √ should route to paginaAPI.controller.destroy
    POST /api/paginaAPIs/fondoPagina
      √ should route to paginaAPI.controller.personalizarFondo
    POST /api/paginaAPIs/paginaInicio
      √ should route to paginaAPI.controller.paginaInicio

  ProyectoAPI API Router:
    √ should return an express router instance
    GET /api/proyectoAPIs
      √ should route to proyectoAPI.controller.index
    POST /api/proyectoAPIs
      √ should route to proyectoAPI.controller.create
    DELETE /api/proyectoAPIs/:id
      √ should route to proyectoAPI.controller.destroy
    POST /api/proyectoAPIs/importar
      √ should route to proyectoAPI.controller.importar
    POST /api/proyectoAPIs/exportar
      √ should route to proyectoAPI.controller.exportar
    POST /api/proyectoAPIs/exportar/zipHTML
      √ should route to proyectoAPI.controller.exportarZipHTML


  20 passing (62ms)


Done, without errors.


Execution Time (2016-06-12 23:12:13 UTC)
loading tasks             325ms  ████  11%
loading grunt-mocha-test   78ms  █ 2%
mochaTest:unit               3s  ██████████████████████████████  86%
Total 3.5s
```

Figura 27 Resultados de las pruebas unitarias del lado del servidor con Mocha

3.3.2 Pruebas de aceptación

Las pruebas de aceptación son consideradas como pruebas de caja negra y son tan importantes como las pruebas unitarias dado que significan la satisfacción del cliente con el producto desarrollado, el final de una iteración y el comienzo de la siguiente. Se elaboran a lo largo de la iteración, en paralelo con el desarrollo del sistema, y adaptándose a los cambios que el sistema sufra. (60)

En general, las pruebas de aceptación pueden adoptar, entre otras, las siguientes formas (56):

Pruebas de aceptación de usuario

En general, verifican la idoneidad de uso del sistema por parte de los usuarios comerciales.

Pruebas operativas (de aceptación)

La aceptación del sistema por parte de los administradores del sistema, entre las que se incluyen:

- Pruebas de *backup*/restauración
- Recuperación de desastres
- Gestión de usuarios
- Tarea de mantenimiento
- Carga de datos y tareas de migración
- Comprobaciones periódicas de vulnerabilidades de seguridad

Pruebas de aceptación contractual y normativa

Las pruebas de aceptación contractual toman como base los criterios de aceptación previstos en un contrato para fabricar un software desarrollado a medida. Los criterios de aceptación deberán establecerse en el momento en que las partes aceptan contraer dicho contrato. Las pruebas de aceptación normativa toman como base cualquier normativa de obligado cumplimiento, tales como normativas gubernamentales, legales o de seguridad.

Pruebas alfa y beta (o de campo)

Las pruebas alfa se llevan a cabo en el emplazamiento de la organización de desarrollo, pero no las realiza el equipo de desarrollo. Las pruebas beta, o pruebas de campo, las realizan los clientes en sus propias instalaciones.

Las organizaciones también pueden emplear otros términos, tales como pruebas de aceptación en fábrica y pruebas de aceptación en emplazamiento, en el caso de sistemas probados antes y después de su traslado a las instalaciones del cliente.

En la validación del software generado fueron utilizadas las pruebas de aceptación de usuarios, para las que fueron elaborados 21 casos de prueba. En este epígrafe se muestran solo el caso de prueba relacionado con la HU definida en el epígrafe 2.5, el resto de los casos de prueba se pueden encontrar en los anexos.

Tabla 3 Diseño de casos de prueba: Crear nuevo proyecto

Descripción general: Permitir crear un nuevo proyecto para que el usuario pueda desarrollar en el mismo.							
Condiciones de ejecución: El campo nombre es obligatorio llenarlo. La imagen debe poseer formato .png y ser menor de 200kb.							
Nombre de la sección: SC1 Crear nuevo proyecto							
Escenarios	Descripción	Nombre del proyecto	Autor	Descripción	Imagen	Respuesta del sistema	Flujo central

EC 1.1 Opción de crear nuevo proyecto	Selecciona la opción de crear nuevo proyecto.					Muestra un formulario para registrar los datos del nuevo proyecto, permitiendo entrar los valores: - Nombre - Autor – Descripción – Imagen. Permite, además: - Guardar los datos introducidos - Cancelar la operación en cualquier momento.	Marco de trabajo/Crear nuevo proyecto
EC 1.2 Opción de aceptar.	Introduce los datos del proyecto y selecciona la opción aceptar.	V	V	V	V	Valida los datos. Muestra el listado de plantillas.	Marco de trabajo/Crear nuevo proyecto /Formulario/Aceptar
EC 1.3 Opción de cancelar	Selecciona la opción de cancelar.					Elimina los datos creados. Regresa a la vista inicial del sistema.	Marco de trabajo/Crear nuevo proyecto /Formulario/Cancelar
EC 1.4 Datos incompletos	Existen datos incompletos.	I	N/A	N/A	N/A	Muestra un indicador sobre el campo vacío. No se habilita la opción de aceptar.	Marco de trabajo/Crear nuevo proyecto /Formulario

EC 1.5 Datos incorrectos	Existen datos incorrectos.	I	V	V	V	Muestra un indicador sobre los campos incorrectos. No se habilita la opción de aceptar.	Marco de trabajo/Crear nuevo proyecto /Formulario
		V	I	V	V		
		V	V	I	V		
		V	V	V	I		
EC 1.6 Seleccionar plantilla	Selecciona una plantilla para la creación de la multimedia.					Entra al entorno. Muestra un mensaje de información. Permite, además: -Cancelar la operación en cualquier momento.	Marco de trabajo/Crear nuevo proyecto /Formulario/Aceptar /Listado de plantillas/Selecciona plantilla
EC 1.7 Opción de cancelar	Selecciona la opción de cancelar.					Elimina los datos creados. Regresa a la vista inicial del sistema.	Marco de trabajo/Crear nuevo proyecto /Formulario/Aceptar/ Listado de plantillas/Cancelar

Descripción de las variables

No	Nombre de campo	Clasificación	Valor nulo	Descripción
1	Nombre	Campo de texto	No	Campo de carácter obligatorio que representa el nombre de la multimedia. Admite caracteres alfanuméricos. Desde 3 hasta 30 caracteres. Ejemplo: Mi multimedia
2	Autor	Campo de texto	Si	Campo opcional que representa el autor de la multimedia. Permite caracteres alfanuméricos desde 3 hasta 30 caracteres.

				Ejemplo: Rosmelys
3	Descripción	Campo de texto	Si	Campo opcional que representa una descripción de la multimedia. Permite caracteres alfanuméricos de hasta 300 caracteres. Ejemplo: Esta es una multimedia de prueba
4	Imagen	Campo de selección	Si	Campo opcional que representa una imagen para acompañar a la multimedia , es un campo de selección.

3.3.2.1 Resultados de las pruebas de aceptación

Al concluir cada iteración planificada para el desarrollo de la propuesta, se realizaron las pruebas correspondientes a la misma con el objetivo de hacer la entrega planificada con la calidad correspondiente. A continuación, se muestra un gráfico con los resultados de las tres iteraciones de pruebas donde se obtuvo un total de 7 no conformidades (NC) significativas y 10 no significativas.

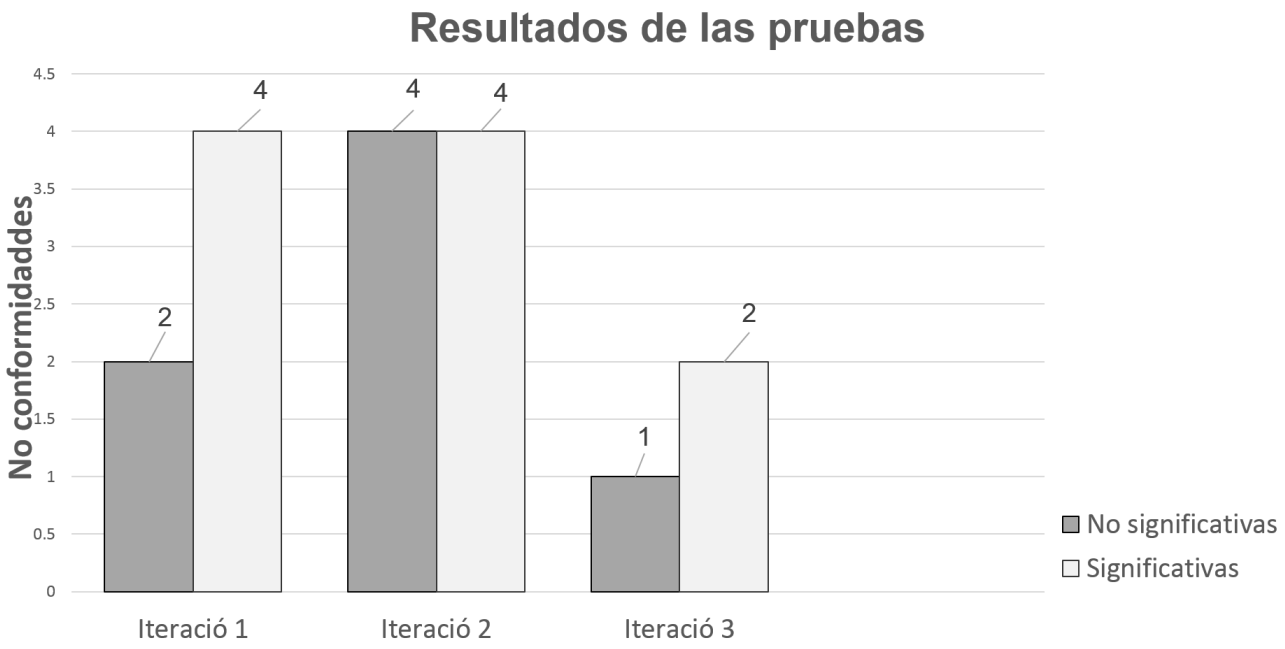


Figura 28 Resultados de las pruebas de aceptación

Las principales NC no significativas encontradas fueron errores ortográficos, tanto omisiones de tildes como cambio de mayúsculas por minúsculas. También se mostraban algunos mensajes innecesarios en pantalla. Las principales NC significativas encontradas fueron errores de integración con los componentes personalizables, validación, errores en cargar las páginas de cada multimedia desde el servidor. Después de concluida cada iteración se resolvieron las no conformidades arrojadas, para garantizar así la satisfacción del cliente.

3.4 Validación experimental de resultados

Para validar los resultados obtenidos teniendo en cuenta la hipótesis: la creación de un entorno de desarrollo para el Marco de trabajo de creación de multimedia con tecnologías libres, que integre y gestione los componentes existentes, permitirá disminuir el tiempo de desarrollo de estos productos; se llevó a cabo un experimento que se describe a continuación.

Fueron seleccionados dos grupos, el primero fue compuesto por dos estudiantes del grupo 4502 y el segundo por dos estudiantes del grupo 4503, pertenecientes a la UCI. A ambos grupos se les pidió que desarrollaran una multimedia integrada por 5 páginas hipervinculadas en las que estuvieran presentes los

componentes personalizables Menú, Galería de imágenes, Mapa y Animaciones, además de los componentes básicos Audio, Video, Texto e Imagen. Se debían secuenciar los componentes e incluirles animaciones de entrada y salida. Por último, se tendría que exportar el producto final como un proyecto HTML. El primer grupo debía desarrollar la multimedia con la herramienta Adobe Flash CS6 mientras que el segundo debía hacerlo con la herramienta implementada como propuesta de solución. Los grupos no poseían conocimientos previos sobre las herramientas, por lo que les fue impartida una capacitación a cada grupo por separado sobre la herramienta que les correspondía. Posteriormente fue posible llevar a cabo el desarrollo de las dos multimedia, arrojando los siguientes resultados:

Tabla 4 Resultado del experimento

Grupo	**Tiempo empleado en el desarrollo de la multimedia**
Grupo 1	21 horas
Grupo 2	17 horas

Una vez analizados los resultados es posible comprobar que la creación del entorno de desarrollo permite disminuir el tiempo de desarrollo de estos productos.

3.5 Conclusiones del capítulo

En este capítulo fueron desarrolladas las disciplinas implementación y pruebas, para lo que fueron generados los diagramas de componentes y despliegue. El primer artefacto permitió entender la relación de dependencia y uso del entorno de desarrollo, mientras que el segundo representó de forma gráfica la relación entre los componentes hardware y software de la propuesta de solución. Se muestra la estructura de carpetas determinada por el patrón arquitectónico MVC con la que fue desarrollada la herramienta y se especifica la estructura que deben presentar los proyectos para ser importados por esta. El uso de los estándares de codificación definidos por el documento “Estándares de codificación Proyecto Marco de trabajo para el desarrollo de multimedia con tecnologías libres" permitieron una mayor organización y claridad del código fuente. Se aplicaron pruebas unitarias, de aceptación al sistema permitiendo determinar y erradicar las deficiencias encontradas, obteniendo finalmente una solución con un alto nivel de calidad.

Además, se desarrolló un experimento mediante el cual fue posible validar la hipótesis planteada en la investigación.

CONCLUSIONES GENERALES

En la culminación de la presente investigación se puede comprobar el cumplimiento de los objetivos propuestos, arribando a las siguientes conclusiones:

- El producto desarrollado cumple con las características obtenidas a partir del estudio de herramientas similares, lo que lo coloca al mismo nivel de otras herramientas existentes.
- La utilización de una metodología ágil ofreció una rápida respuesta a cambios de requisitos a lo largo del desarrollo del proyecto gracias a su proceso iterativo. Para el desarrollo fueron usadas herramientas y tecnologías libres como son: Bower, Grunt, Yeoman, AngularJS, NodeJS, JavaScript, HTML5 y CSS3.
- La creación del entorno de desarrollo permitió integrar un grupo de componentes personalizables en una herramienta de fácil manejo, capaz de brindar un ambiente de desarrollo multiplataforma para la creación de multimedia con tecnologías libres.
- Las pruebas realizadas al software permitieron validar las funcionalidades atendiendo a los requisitos del cliente. El experimento realizado demostró la veracidad de la hipótesis.

RECOMENDACIONES

A partir del estudio realizado y los resultados o beneficios que proporciona este trabajo de diploma, se proponen las siguientes recomendaciones:

- ✓ La herramienta debería contar con un sistema de autentificación garantizando así el acceso a las multimedia solo del personal autorizado.
- ✓ Que sean incorporadas a la herramienta las siguientes funcionalidades:
    - Una vez que haya sido seleccionada la plantilla, se permita al usuario realizar modificaciones sobre la misma.
    - Se puedan añadir transiciones a las páginas.
    - Ordenar componentes.
    - Deshacer y Rehacer.
    - Edición de audio

REFERENCIAS BIBLIOGRÁFICAS

1. PERLACIA REAL V, MARTÍNEZ CARBALLO Y. Componentes básicos para el marco de trabajo de desarrollo de multimedia con tecnologías libres. [Habana, Cuba]: Universidad de las Ciencias Informáticas; 2015.

2. ASALE R-. Diccionario de la lengua española - Edición del Tricentenario [Internet]. Diccionario de la lengua española. [citado 5 de febrero de 2016]. Recuperado a partir de: http://dle.rae.es/?id=Q4K6XyV

3. RUÍZ SÁNCHEZ M, LEAL RUGAMA MI. Multimedia [Internet]. 2008 [citado 14 de diciembre de 2015]. Recuperado a partir de: http://www.binasss.sa.cr/revistas/enfermeria/v24n1/art7.htm

4. PINTO M. Alfamedia [Internet]. 2011 [citado 14 de diciembre de 2015]. Recuperado a partir de: http://www.mariapinto.es/alfamedia/cultura/entornos.htm

5. GALLEGO DJ, ALONSO CM. Sistemas Multimedia. En: Tecnología Educativa Nuevas Tecnologías aplicadas a la Educación. Alcoy. Alicante: Editorial Marfil S. A.; 1995. p. 165-86.

6. PRESSMAN, R. S. Ingenieria de software. Un enfoque práctico. Editorial Felix Varela;

7. DÍAZ PÉREZ P, CATENAZZI, Nadia, AEDO CUEVAS, Ignacio. De la multimedia a la hipermedia. Madrid: Editorial Rama; 1996. 291 p.

8. BELLOCH C. Aplicaciones Multimedia.

9. BIANCHINI A. Conceptos y definiciones de hipertexto [Internet]. 2000 [citado 14 de diciembre de 2015]. Recuperado a partir de: http://ldc.usb.ve/~abianc/hipertexto.html

10. REHMAN RU, PAUL C. The Linux Development Platform: Configuring, Using and Maintaining a Complete Programming Environment. 2002.

11. RAZQUIN ZAZPE P. Los sistemas de autor multimedia. Revista General de Información y Documentación. 2014;8(2).

12. Adobe [Internet]. 2015. Recuperado a partir de: http://www.escuelaslibres.org.ar/2010/07/ktoon-animacion-2d-en-linux/

13. D´ ÁVILA M. ToolBook x Director [Internet]. 2004 [citado 24 de febrero de 2016]. Recuperado a partir de: http://www.mhavila.com.br/topicos/mm/tbk_director.html

14. Soft&Apps [Internet]. 2013. Recuperado a partir de: http://www.softandapps.info/2013/05/06/powtoon-utilidad-web-para-crear-presentaciones-y-videos/

15. Meograph [Internet]. 2015. Recuperado a partir de: http://www.meograph.com/robertoalejandro/155485/new-mix-2?moment=614925#edit

16. Clases de Periodismo [Internet]. 2013. Recuperado a partir de: http://www.clasesdeperiodismo.com/2013/10/18/5-herramientas-para-crear-historias-multimedia/

17. BELLOCH ORTÍ C. Diseño y desarrollo de aplicaciones multimedia educativas [Internet]. Recuperado a partir de: http://www.uv.es/bellochc/pwedu6.htm

18. MARTIN J. Meograph, la nueva herramienta de ayuda al periodista para contar historias interactivas [Internet]. [citado 15 de marzo de 2016]. Recuperado a partir de: http://www.checkapps.net/2012/07/meograph-herramienta-ayuda-periodistas-historias-multimedia.html

19. ZUÑIGA A. Powtoon [Internet]. prezi.com. [citado 15 de marzo de 2016]. Recuperado a partir de: https://prezi.com/amx4yjom9lix/powtoon/

20. RODRÍGUEZ DE LA CRUZ A. FORTES_MTM_Plan_de_desarrollo_de_software.

21. RODRÍGUEZ SÁNCHEZ, Tamara. Metodología de desarrollo para la Actividad productiva de la UCI. 2014.

22. Visual Paradigm Company Overview [Internet]. [citado 3 de febrero de 2016]. Recuperado a partir de: http://www.visual-paradigm.com/aboutus/

23. WebStorm: The Smartest JavaScript IDE [Internet]. JetBrains. [citado 19 de abril de 2016]. Recuperado a partir de: https://www.jetbrains.com/webstorm/

24. LÓPEZ SALVADOR V. Yeoman, Grunt, Bower, ¡Oh My!!! [Internet]. RICE. 2015 [citado 2 de mayo de 2016]. Recuperado a partir de: https://rice.uci.cu/?p=3552

25. Lenguajes del lado servidor o cliente [Internet]. [citado 9 de diciembre de 2015]. Recuperado a partir de: http://www.adelat.org/media/docum/nuke_publico/lenguajes_del_lado_servidor_o_cliente.html

26. HTML | Codigoprogramacion [Internet]. [citado 9 de diciembre de 2015]. Recuperado a partir de: http://codigoprogramacion.com/category/cursos/html.

27. GAUCHAT JD. El gran libro de HTML5, CSS3 y Javascript. Marcombo. 2012.

28. NEHER R. JavaScript [Internet]. Mozilla Developer Network. 2015 [citado 7 de abril de 2016]. Recuperado a partir de: https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Introducci%C3%B3n#What_is_JavaScript

29. Mauricio. Desarrollo de Aplicaciones Web: 1.3 Lenguajes de Programación Web y DBMS [Internet]. Desarrollo de Aplicaciones Web. 2013 [citado 3 de febrero de 2016]. Recuperado a partir de: http://appsdelweb.blogspot.com/2013/02/13-lenguajes-de-programacion-web-y-dbms.html

30. Node.js javascript en el servidor [Internet]. [citado 6 de febrero de 2016]. Recuperado a partir de: http://hipertextual.com/archivo/2014/04/nodejs-javascript-servidor/

31. jquery.org jQuery F-. jQuery [Internet]. [citado 19 de abril de 2016]. Recuperado a partir de: https://jquery.com/

32. jquery.org jQuery F-. jQuery UI [Internet]. [citado 19 de abril de 2016]. Recuperado a partir de: https://jqueryui.com/

33. AngularJS. Framework JavaScript para Webapps «Los Tiempos Cambian [Internet]. [citado 3 de febrero de 2016]. Recuperado a partir de: http://www.lostiemposcambian.com/blog/javascript/angularjs-framework-javascript-para-webapps/

34. VALBUENA APONTE, Ángela María. Guía comparativa de Frameworks para los lenguajes HTML 5, CSS y JavaScript para el desarrollo de aplicaciones Web. Universidad Tecnológica de Pereira; 2014.

35. Node.js Introduction [Internet]. www.tutorialspoint.com. [citado 9 de diciembre de 2015]. Recuperado a partir de: http://www.tutorialspoint.com/nodejs/nodejs_introduction.htm

36. Express - Node.js web application framework [Internet]. [citado 19 de abril de 2016]. Recuperado a partir de: http://expressjs.com/

37. WANG R. NW.js Documentation [Internet]. [citado 17 de mayo de 2016]. Recuperado a partir de: http://docs.nwjs.io/en/latest/

38. JACOBSON I, BOOCH G, RUMBAUGH J. El proceso unificado de desarrollo de software. Madrid: Pearson Educación. S.A.; 2002. 464 p.

39. KOCH N, ESCALONA MJ. Ingeniería de Requisitos en Aplicaciones para la Web [Internet]. 2002 [citado 1 de marzo de 2016]. Recuperado a partir de: https://www.powtoon.com/about/

40. AGÜERO MIRABAL L. Requerimientos funcionales y no funcionales [Internet]. 16:04:34 UTC [citado 2 de marzo de 2016]. Recuperado a partir de: http://www.slideshare.net/Lismirabal/requerimientos-funcionales-y-no-funcionales

41. BASS L, CLEMENTS P, KAZMAN R. Software Architecture in Practice. 2da ed. Addison Wesley; 2003.

42. GAMMA E, HELM R, JOHNSON R, VLISSIDES J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison Wesley; 1994.

43. LARMAN C. UML y patrones: análisis y diseño orientado a objetos. México: Prentice Hall; 1999.

44. Patrones GRASP. Patrones GoF. Diferencia entre GRASP y GoF - TuxNots [Internet]. [citado 8 de marzo de 2016]. Recuperado a partir de: https://sites.google.com/site/tuxnots/materias-de-la-facu/metodologia-de-sistemas/patronesgrasppatronesgofdiferenciaentregraspygof

45. MARTÍNEZ JUAN FJ. Guía de construcción de software en java con patrones de diseño ESCUELA UNIVERSITARIA DE INGENIERÍA TÉCNICA EN INFORMÁTICA DE OVIEDO. Escuela universitaria de ingeniería técnica en informática de Oviedo;

46. OSMANY A. Learning JavaScript Design Patterns. 2012.

47. PRATT M. Inyección de dependencias. ¿Qué es y para qué sirve? [Internet]. Codecriticon. 2012 [citado 3 de mayo de 2016]. Recuperado a partir de: http://codecriticon.com/inyeccion-de-dependencias/

48. LINDESAY F. Promises [Internet]. [citado 3 de mayo de 2016]. Recuperado a partir de: https://www.promisejs.org/

49. PRESSMAN R. Software Engineering. A Practitioner's Approach. USA; 1999. 241 p.

50. PRESSMAN RS. Ingeniería del software, un enfoque práctico. 2005.

51. ESCRIBANO GF. Introducción a Extreme Programming. 2002.

52. RODRÍGUEZ DE LA CRUZ A. Estándares de codificación Proyecto Marco de trabajo para el desarrollo de multimedia con tecnologías libres.

53. NIETO J. Sistema web Aymi. Estándares de Programación. Versión 3.0. 2011.

54. PRESSMAN RS. Software Engineering: A Practitoner´s Approach. seventh. New York: McGraw Hill; 2010.

55. VIVAS WHITE P, LOPEZ ROMERO E. Informática: Cuerpo de Profesores de Enseñanza Secundaria. Temario. CEP; 2009. 638 p.

56. MÜLLER T, BEER A, KLONK M, VERMA R. Probador Certificado. Programa de estudio de nivel básico. 2010.

57. Karma [Internet]. GitHub. 2016 [citado 3 de mayo de 2016]. Recuperado a partir de: https://karma-runner.github.io/0.8/index.html

58. PIVOTAL LABS. Jasmine: Behavior-Driven JavaScript [Internet]. 2015 [citado 21 de abril de 2016]. Recuperado a partir de: http://jasmine.github.io/1.3/introduction.html

59. Minutes AQH · ST 13. Testing MEAN Applications - Mocha - Mastering MEAN [Internet]. Testing MEAN Applications - Mocha - Mastering MEAN. [citado 24 de mayo de 2016]. Recuperado a partir de: https://masteringmean.com/lessons/211-Testing-MEAN-Applications-Mocha

60. PÉREZ IC, PÉREZ GONZÁLEZ R, RODRÍGUEZ MARTÍN AD. METODOLOGIA DE DESARROLLO [Internet]. [citado 19 de mayo de 2016]. Recuperado a partir de: http://www.academia.edu/8187104/METODOLOGIA_DE_DESARROLLO