Universidad de las Ciencias Informáticas

Facultad 4

Biblioteca de componentes para el desarrollo de aplicaciones con tecnología multimedia en el centro FORTES de la UCI

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

Autores:

Miguel Pérez Fernández

Alejandro Rodríguez de la Cruz

Tutores:

Ing. Arlan Galvez Alonso

Ing. Nilber Barbán Góngora

La Habana, junio 2014

"Año 56 de la Revolución"

DECLARACIÓN DE AUTORÍA:

Por este medio declaramos ser los únicos autores del presente trabajo de diploma y autorizamos a la Facultad 4 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

_____	_____
Firma del Tesista	Firma del Tesista
Miguel Pérez Fernández	Alejandro Rodríguez de la Cruz
_____	_____
Firma del tutor	Firma del tutor
Ing. Arlan Galvez Alonso	Ing. Nilber Barbán Góngora

"La ciencia no es más que un refinamiento del pensamiento cotidiano."

Albert Einstein

Dedicatoria

De Miguel

A mis padres Niurka Fernández Alonso y Rolando Pérez Mestril que han sido la piedra angular que me ha permitido llegar hasta aquí.

A toda mi familia, que me ha apoyado y siempre ha estado ahí para mí.

De Alejandro

A mis padres Odalis de la Cruz Pérez y David F. Rodríguez Brito, que me han apoyado en todo momento durante estos cinco años para llegar hasta aquí.

A todos mis abuelos, que aportaron su granito de arena para forjar en mí los verdaderos valores humanos.

Agradecimientos

De Miguel

A mis padres, por estar siempre ahí para mí y por ser lo más grande que tengo.

A toda mi familia por apoyarme en los momentos difíciles.

A mi compañero de tesis por la preocupación y la entrega durante todo este proceso.

A Ely por el amor y cariño que me ha dado, por estar a mi lado y aguantarme en los momentos de estrés.

A mis tutores (sobre todo Arlan), por servirme de guía en este proceso tan complicado que es el trabajo de diploma.

A mis compañeros de grupo de estos 5 años, por haber compartido tan bellos momentos.

Y a todos aquellos que de una forma u otra han hecho posible que este aquí y que pueda cumplir este sueño.

De Alejandro

A mis padres que son lo más grande de mi vida.

A mis abuelos que no están y a mi abuela Arminda que sigue cuidando de mí.

A mi tía Muma y a Luis por soportarme todas las veces que los he molestado durante la carrera.

A mis primos Rachel y Lulo que son mis hermanos.

A las amistades que a lo largo de estos 5 años estuvieron compartiendo conmigo los momentos buenos y malos.

A Ana que con mucho amor y paciencia me ha apoyado y querido sin fin.

A los tutores en especial a Arlan que ha estado guiando la investigación del trabajo de diploma.

En fin a todos los que me ayudaron.

Resumen

En los últimos años, con el auge de las TIC, se hace difícil pensar que las herramientas para el almacenamiento y clasificación de los componentes para el desarrollo de aplicaciones con tecnología multimedia no puedan ser utilizadas a través de la red mediante un servidor web. La flexibilidad que ofrece la tecnología web debe aprovecharse para crear un entorno que apoye el desarrollo basado en la reutilización de componentes.

Actualmente en el centro FORTES los componentes elaborados en soluciones anteriores, no son aprovechados correctamente en la creación de nuevas aplicaciones con características similares. Ello dificulta la posibilidad de desarrollar aplicaciones a partir de estructuras ya definidas, perdiendo tiempo y recursos realizando búsquedas bibliográficas para lograr su reutilización. Para brindar solución a los problemas antes planteados se propone como objetivo general de la investigación: elaborar una biblioteca de componentes para el desarrollo de aplicaciones con tecnología multimedia en el centro FORTES, que permita organizar y centralizar la documentación y los componentes para el proceso de elaboración de aplicaciones con este tipo de tecnología, favoreciendo su reutilización. En su cumplimiento, se realizó un estudio del estado del arte sobre las bibliotecas de componentes y se seleccionaron las herramientas y tecnologías necesarias para su desarrollo.

El resultado de la investigación deja constancia documental de la metodología y herramientas empleadas, y en cumplimiento al objetivo propuesto se obtuvo una aplicación web que permite el almacenamiento y organización de los componentes de tecnología multimedia desarrollados en el Centro, favoreciendo así su reutilización.

Palabras claves: biblioteca, componentes, multimedia

Contenido

INTRODUCCIÓN 1
CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA 5
1.1 Introducción 5
1.2 Conceptos asociados al objeto de estudio 5
1.3 Sistemas homólogos a nivel mundial 10
1.3.1 Biblioteca de Componentes Visuales 10
1.3.2 jQuery UI 11
1.3.3 Diseño Asistido por Computador de Sistemas de Potencia 11
1.4 Sistemas homólogos en Cuba 12
1.5 Sistemas homólogos en la UCI 12
1.5.1 Biblioteca de componentes para la implementación de la interfaz web de eXcriba 12
1.5.2 Biblioteca de componentes de software 13
1.6 Tendencias 14
1.7 Metodología de desarrollo de software 17
1.7.1 Metodologías tradicionales 17
1.7.2 Metodologías ágiles 17
1.8 Herramientas y tecnologías 20
1.8.1 Sistema gestor de contenidos 20
1.8.2 Sistema gestor de base de datos 23
1.8.3 Servidor web 25
1.8.4 Lenguajes de programación 26
1.8.5 Entorno de desarrollo integrado 30
1.8.6 Marco de trabajo 31
1.8.7 Herramientas CASE 33
1.8.8 Herramientas de prototipado de interfaces de usuario 34
1.9 Conclusiones del capítulo 36
CAPÍTULO 2. ELABORACIÓN DE LA PROPUESTA DE SOLUCIÓN 37
2.1 Introducción 37
2.2 Descripción del flujo actual del proceso 37

Índice de contenido

2.3 Propuesta de solución 37
2.4 Definición de la audiencia 38
2.5 Resultados esperados 38
2.6 Usuarios relacionados con el sistema 38
2.7 Estructura definida para un componente 39
2.8 Fase de exploración 41
2.8.1 Aspectos funcionales del sistema 41
2.8.2 Aspectos no funcionales del sistema 42
2.8.3 Historias de usuario (HU) 44
2.9 Fase de Planificación 46
2.9.1 Estimación de esfuerzos por HU 46
2.9.2 Plan de iteraciones 47
2.9.3 Plan de duración de las iteraciones 48
2.9.4 Plan de entregas 49
2.10 Conclusiones del capítulo 51
CAPÍTULO 3. PRODUCCIÓN Y PRUEBAS 52
3.1 Introducción 52
3.2 Diseño del sistema 52
3.2.1 Metáfora 52
3.2.2 Tarjetas Clase Responsabilidad Colaboración (CRC) 52
3.2.3 Prototipo de interfaz 53
3.2.4 Estándares de codificación 54
3.3 Fase de producción 58
3.3.1 Arquitectura de software 59
3.3.2 Patrones de diseño 59
3.3.3 Módulos desarrollados para la biblioteca 61
3.3.4 Términos para publicar componentes en la biblioteca 63
3.3.5 Tareas de ingeniería 64
3.4 Fase de pruebas 65
3.4.1 Pruebas unitarias 66

Índice de contenido

3.4.2 Pruebas de aceptación 68
3.4.3 Resultados de las pruebas de aceptación 70
3.5 Conclusiones del capítulo 71
CONCLUSIONES GENERALES 72
RECOMENDACIONES 73
REFERENCIAS BIBLIOGRÁFICAS 74
BIBLIOGRAFÍA 82
GLOSARIO DE TÉRMINOS 83
ANEXO 1: Tendencias de los lenguajes HTML5, CSS3 y ActionScript3 85
ANEXO 2: Historias de usuario 93
ANEXO 3: Tarjetas CRC 110
ANEXO 4: Tareas de ingeniería de las HU 112
ANEXO 5: Casos de prueba de aceptación 130

Índice de figuras

Fig.1 Tendencia de los lenguajes HTML5, ActionScript3 y CSS3 16
Fig.2 Proceso de vida del software en la Metodología XP 19
Fig.3 Estructura del componente 40
Fig.4 Estructura incorrecta de un componente 41
Fig.5 Prototipo de interfaz de usuario de la página principal de la biblioteca de componentes multimedia 54
Fig.6 Uso del punto y coma en Drupal 55
Fig.7 Estructuras de control de Drupal 56
Fig.8 Llamadas a una función en Drupal 56
Fig.9 Asignaciones 57
Fig.10 Declaración de arreglos con Drupal 57
Fig.11 Arquitectura empleada por el CMS Drupal 59
Fig.12 Estructura del módulo bibliocom 62
Fig.13 Relaciones entre las tablas principales del sistema 63
Fig.14 Pruebas unitarias al sistema 66
Fig.15 Realización de pruebas unitarias al sistema 67
Fig.16 Resultado de las pruebas unitarias al sistema 67
Fig.17 Resultados de las pruebas de aceptación al sistema 71

Índice de tablas

Tabla 1 Roles del sistema....39
Tabla 2 HU- Crear cuenta de usuario....45
Tabla 3 HU- Personalizar componente....45
Tabla 4 Plan de estimación de esfuerzos por HU....46
Tabla 5 Plan de duración de las iteraciones....49
Tabla 6 Plan de entregas....49
Tabla 7 Tarjeta CRC Control de versiones y personalización de componentes....53
Tabla 8 Tarjeta CRC Compartir componente....53
Tabla 9 HU de la iteración 1....64
Tabla 10 Tarea de ingeniería de la HU1....64
Tabla 11 Tarea de ingeniería de la HU2....65
Tabla 12 Caso de prueba de aceptación HU1....68
Tabla 13 Caso de prueba aceptación HU2....69
Tabla 14 No conformidades por iteración....70

INTRODUCCIÓN

Décadas atrás la información era transmitida a través de textos, imágenes, u otra media. Con el auge de las Tecnologías de la Información y las Comunicaciones (TIC) se ha transformado el paradigma de la representación de los contenidos, agrupando los recursos audiovisuales y sumando un elemento primordial del aprendizaje como es la interactividad (1), logrando que la comunicación sea bilateral y no unilateral como hacían los medios tradicionales; esto es logrado gracias a los productos multimedia, donde son vinculados las imágenes, los textos, el audio y el video. Los productos multimedia con la influencia de las TIC han provocado una revolución, principalmente en la mercadotecnia y la educación. A nivel mundial el uso de estos productos en la educación se ha vuelto imprescindible en los métodos educativos a distancia para apoyar el aprendizaje activo y colaborativo.

En la actualidad el desarrollo tecnológico ha aumentado en el sector de los dispositivos móviles y la tendencia a la tecnología en la nube, imponiendo nuevas formas de representación de la información, apoyándose en la fabricación de productos con tecnología multimedia. Las nuevas tecnologías han cambiado el paradigma de los productos multimedia, al resaltar la necesidad de nuevas características como: la eficiencia y la adaptabilidad a los nuevos entornos, lo que implica la necesidad de disminuir el consumo de recursos y el tiempo de desarrollo de los mismos. El tiempo de desarrollo de un multimedia es clave porque con frecuencia se generan nuevos contenidos a representar. Las empresas buscan ser cada vez más competentes en los flujos del desarrollo de productos basados en estos criterios, optando por el uso de la reutilización de componentes. Surge la necesidad de producir estos productos más complejos y en plazos de desarrollo más cortos, empleando conceptos como: desarrollo basado en componentes y Líneas de Productos de Software (LPS). Una LPS consiste en un conjunto de elementos clave para producir sistemas de software que comparten características comunes o similitudes, pero al mismo tiempo mantienen características propias (2). Algunos de los grandes productores de multimedia (*VisualDat*, Comunidades de Castilla-La Mancha) utilizan las LPS para la reutilización de componentes buscando la eficiencia en el proceso de producción. La implementación del uso de componentes les beneficia considerablemente en el desarrollo de nuevos productos con mayor calidad y con menos recursos humanos.

En la Universidad de las Ciencias Informáticas (UCI), se confeccionan productos multimedia de diferentes temáticas. Dichos productos están enfocados principalmente al entorno educativo, donde muchos tienen el

fin de apoyar el proceso de enseñanza-aprendizaje de los planes de estudios por los que se rige la institución, además se crean otros para soportes informativos y promocionales. La mayoría de estos productos se desarrollan en el Centro de Tecnologías para la Formación (FORTES) que a su vez cuenta con un departamento de desarrollo de componentes, desglosado en una línea de producción multimedia encargada de desarrollar componentes reutilizables, para el ensamblaje de dichos productos. Actualmente los componentes elaborados en soluciones anteriores, no son aprovechados correctamente en la creación de nuevas aplicaciones con características similares. Ello dificulta la posibilidad de desarrollar aplicaciones a partir de estructuras ya definidas. Además se pierde tiempo y recursos realizando búsquedas bibliográficas para lograr la reutilización de los mismos. La documentación técnica del proceso de desarrollo de productos de este tipo es escasa y está desactualizada.

A partir de la situación problémica antes descrita, se plantea como **problema a resolver** ¿Cómo contribuir al desarrollo de aplicaciones con tecnología multimedia en el centro FORTES? Se define como **objeto de estudio** las bibliotecas de componentes, mientras que el **campo de acción** se enmarca en las bibliotecas de componentes para el desarrollo de aplicaciones con tecnología multimedia.

El **objetivo general** de la investigación es elaborar una biblioteca de componentes para el desarrollo de aplicaciones con tecnología multimedia en el centro FORTES, que permita organizar y centralizar la documentación y los componentes para el proceso de elaboración de aplicaciones con este tipo de tecnología, favoreciendo su reutilización. En relación con lo anterior se plantean los siguientes **objetivos específicos**:

- Formular el marco teórico de la investigación relacionado con las bibliotecas de componentes multimedia.
- Elaborar la propuesta de la solución.
- Evaluar la calidad del sistema propuesto.

Para dar cumplimiento a cada uno de los objetivos planteados se han definido las **tareas de investigación** siguientes:

- Elaboración de los diseños teóricos y metodológicos de la investigación.
- Revisión bibliográfica acerca de bibliotecas para la reutilización de componentes con tecnología multimedia.

- Descripción de las diferentes metodologías de desarrollo de software y selección de la que guiará la investigación.
- Investigación de las tendencias, herramientas y tecnologías actuales para llevar a cabo la solución.
- Identificación y descripción de las necesidades del cliente.
- Diseño de la interfaz de usuario.
- Especificación de las estrategias de codificación, los estándares y patrones a utilizar en el proceso de desarrollo.
- Implementación de las funcionalidades identificadas para el software.
- Realización de las pruebas al sistema.

Como **idea a defender** se define: con la elaboración de una biblioteca de componentes para el desarrollo de aplicaciones con tecnología multimedia en el centro FORTES, se organizará y centralizará los componentes y la documentación referente a los mismos, favoreciendo su reutilización.

Métodos teóricos

Analítico-sintético: Para desarrollar la investigación fue necesario descomponer en partes el problema, para analizar desde diferentes aristas los conceptos asociados al objeto de estudio y los sistemas similares. Lo que facilita la comprensión del problema planteado y permite que una vez unidas las partes nuevamente, se sinteticen las ideas y permitan la descripción de las relaciones entre ellas y sus características.

Análisis histórico-lógico: Para estudiar la evolución histórica y desarrollo de las bibliotecas de componentes, así como las tendencias de lenguajes y metodologías empleados en su desarrollo. Proporcionando un estudio del funcionamiento de los servicios prestados en las bibliotecas analizadas y de las tendencias actuales en el proceso de desarrollo de aplicaciones con este tipo de tecnología.

Métodos empíricos

Revisión documental: Para recopilar y analizar la información referente a las bibliotecas de componentes. Ello permitió la elaboración del marco teórico conceptual basado en el desarrollo coherente y contextualizado de las ideas asociadas al objeto de estudio; estableciendo además, las relaciones y diferencias entre las mismas.

El presente documento consta de una introducción, tres capítulos, conclusiones y recomendaciones, los capítulos quedan estructurados de la siguiente forma:

Capítulo 1. Fundamentación teórica. En este capítulo se realiza un estudio de los elementos teóricos para el desarrollo de la biblioteca de componentes multimedia. Se analizan sistemas con características similares y las tendencias actuales para el desarrollo de productos con tecnología multimedia. Luego se analiza y selecciona la metodología de desarrollo de software, las herramientas y tecnologías adecuadas basado en las ventajas que tiene para el desarrollo de la biblioteca de componentes.

Capítulo 2. Elaboración de la propuesta de solución. Se describen las características de la biblioteca de componentes. Se explica detalladamente las fases de Exploración y Planificación de la metodología XP, generando sus respectivos artefactos dentro de los que se encuentran las Historias de usuario (HU), el plan de iteraciones así como el plan de entrega.

Capítulo 3. Producción y pruebas. Este capítulo recoge aspectos de las fases de XP Producción y Pruebas, generando sus respectivos artefactos dentro de los que se encuentran las tarjetas Clase-Responsabilidad-Colaboración (CRC) y las pruebas de aceptación. Se detalla además cómo se implementa el sistema, se reflejan los estándares de codificación y los resultados obtenidos en el proceso de pruebas al mismo.

Como **posibles resultados** de la investigación se espera obtener una biblioteca de componentes para el desarrollo de aplicaciones con tecnología multimedia en el centro FORTES, que permita organizar y centralizar la documentación y los componentes para el proceso de elaboración de aplicaciones con este tipo de tecnología, favoreciendo su reutilización. Además el sistema permitirá la personalización de dichos componentes a través del código, el control de diferentes versiones de los mismos y brindará la posibilidad de contar con una biblioteca personal, donde los usuarios serán capaces de guardar todos aquellos componentes que le resulten de interés.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se expone una visión general de los aspectos relacionados con las bibliotecas de componentes, los conceptos necesarios para el estudio de las mismas y las tendencias del desarrollo de productos con dicha tecnología. Se analizan diferentes bibliotecas de componentes empleadas a nivel mundial, en Cuba y en la UCI y se enfocan los principales aspectos de las metodologías de desarrollo de software. Por último se describen las herramientas, tecnologías y lenguajes de programación para llevar a cabo el proceso de desarrollo de la biblioteca de componentes para el desarrollo de aplicaciones con tecnología multimedia.

1.2 Conceptos asociados al objeto de estudio

Multimedia

Es aquello que se expresa, transmite o percibe a través de varios medios (3). Además el Diccionario en línea de Cambridge, la define como el uso de una combinación de imágenes estáticas y móviles, sonido, música y palabras, especialmente en ordenadores o entretenimiento (4).

Por otra parte (5) explica que: “multimedia es el uso del ordenador para presentar y combinar: texto, gráficos, audio y video con enlaces que permitan al usuario navegar, interactuar, crear y comunicarse”.

Reutilización

Reutilización es definida como la acción y efecto de reutilizar (6). También se debe considerar como reutilizable que el recurso debe ser modular para servir como base o componente de otro recurso (7). La reutilización de software es el proceso de implementar o actualizar sistemas de software usando activos de software existentes.

Componentes

Un componente es una pieza de código pre-elaborado que encapsula alguna funcionalidad expuesta a través de interfaces estándar (8). Además, es considerado como aquello que forma parte de la composición de un todo. Se trata de elementos que, a través de algún tipo de asociación o contigüidad, dan lugar a un conjunto uniforme (9).

Un componente es una parte no trivial, casi independiente y reemplazable de un sistema que llena claramente una funcionalidad dentro de un contexto en una arquitectura bien definida. Se conforma y provee la realización física por medio de un conjunto de interfaces (9).

Tradicionalmente se toma por componente aquella parte de un todo que puede incluirse en otros elementos con características similares. Dentro de las características fundamentales se encuentran (10)(11):

- Identificable – Un componente debe tener una identificación clara y consistente que facilite su catalogación y búsqueda en repositorios de componentes.
- Auto-contenido – Un componente no debe requerir de la utilización de otros para ejecutar la función para la cual fue diseñado.
- Genérico – Sus servicios deben servir para varias aplicaciones.
- Mantenido – Un componente debe estar inmerso en un proceso de mejoramiento continuo que le garantice al integrador nuevas versiones que incluyan correctivos, optimizaciones y nuevas características.
- Accesible solo a través de su interfaz – El componente debe exponer al público únicamente el conjunto de operaciones que lo caracteriza (interfaz) y ocultar sus detalles de implementación. Esta característica permite que un componente sea reemplazado por otro que implemente la misma interfaz.
- Sus servicios son invariantes – Las operaciones que ofrece un componente, a través de su interfaz, no deben variar. La implementación de estos servicios puede ser modificada, pero no debe afectar la interfaz.
- Documentado – Un componente debe tener una documentación adecuada que facilite su búsqueda en repositorios de componentes, adaptación a nuevos entornos e integración con otros componentes.
- Certificado – El componente puede ser certificado por una agencia de software independiente o mediante la aplicación de modelos de auto-certificación que le permiten al comprador del componente determinar la calidad del software adquirido.

Reutilización de componentes

La reutilización de componentes de software es un proceso inspirado en la manera en que se producen y ensamblan componentes en la ingeniería de sistemas físicos. Es el proceso de reutilización a través de la

producción de componentes genéricos, fáciles de integrar, distribuidos e independientes de las plataformas de desarrollo (11).

Otros autores lo definen como un conjunto de elementos pre-ensamblados en un marco de aplicación específico (12), o conceptualizado como “el proceso de crear sistemas de software a partir de software existente, en lugar de desarrollarlo desde el comienzo” (13).

Por lo que se abordará en la investigación como aquellos componentes que pueden ser empleados para la confección de nuevos productos de software.

Línea de productos de software

Una línea de productos de software es un conjunto de sistemas basados en software y productos similares producidos a partir de un conjunto de activos de software compartido mediante un medio común de producción (14).

En el Instituto de Ingeniería de Software (SEI) de la Universidad Carnegie Mellon, se define la línea de productos de software como un conjunto de software de uso intensivo que gestionan un conjunto de características que satisfacen las necesidades específicas de un segmento especial del mercado o misión y que se desarrollan a partir de un conjunto común de activos esenciales de una manera prescrita (15).

Estas líneas productivas actualmente están enfocadas a las técnicas de ingeniería y administración del proceso de creación, evolución y mantenimiento de una línea de producción por lo se puede encontrar como Línea de Productos de Ingeniería para sistemas y software.

Línea de productos de software multimedia

Al analizar los conceptos anteriores de línea de productos de software, se entiende por línea de productos de software multimedia a todas las líneas de productos de software de una factoría que están destinadas a la creación de productos de tecnología multimedia.

Repositorio

En el diccionario de la Real Academia de la Lengua Española se define al repositorio como: “el lugar donde se guarda algo” (16).

Además deben cumplir funciones de preservación, gestión del acceso y aumento de la visibilidad de los datos (17).

Repositorio institucional

"Un repositorio institucional es un conjunto de servicios que una universidad ofrece a los miembros de su comunidad para la gestión y diseminación de los materiales digitales creados por la institución y sus miembros. Destacable la consideración de servicio que se da al repositorio, como algo dinámico al servicio de la comunidad, más allá de un depósito pasivo de documentos digitales" (18).

Se considera como aquellos servicios prestados por las universidades, al conjunto de la comunidad, para recopilar, administrar, difundir y preservar la producción documental digital generada en la institución, cualquiera que sea su tipología, a través de la creación de una colección digital organizada (19).

Biblioteca

Una biblioteca puede definirse, desde un punto de vista estrictamente etimológico, como el lugar donde se guardan libros, sin embargo en la actualidad […] puede referirse tanto a las colecciones bibliográficas y audiovisuales (20). Además es aquella "institución cuya finalidad consiste en la adquisición, conservación, estudio y exposición de libros y documentos" (21).

En el Diccionario de Términos de Informática se dice, es la institución que realiza la recopilación, procesamiento y almacenamiento de obras impresas y su entrega a los lectores, divulga los conocimientos y dirige la lectura (22).

Biblioteca digital

La Federación de Bibliotecas Digitales considera que son: "Organizaciones que proveen recursos, incluyendo el personal especializado, para seleccionar, estructurar, ofrecer acceso intelectual, interpretar, distribuir, preservar, la integridad y asegurar la persistencia a lo largo del tiempo de obras digitales, de tal forma que estén disponibles para su uso por parte de una comunidad definida o conjunto de comunidades" (23).

Es además, una biblioteca accesible a través de redes electrónicas, cuyos textos son estructurados y por tanto pueden ser organizados en bases de datos, a fin de buscar información rápidamente, copiado y pegados, desplegados en hipertextos (22).

Esta forma de biblioteca les permite a los usuarios el fácil acceso a la información en línea, además de ser una nueva vía de obtención de información.

Biblioteca virtual

Es aquella que además de ofrecer materiales en formato digital, ofrece servicios bibliotecarios en línea o virtual, de ambas formas se aprovecha la respuesta del usuario para organizar y brindarle los materiales que este necesita o para ofrecerles las herramientas necesarias con el fin de obtener información (23).

Una biblioteca virtual es la que puede prestar sus servicios desde cualquier lugar sin necesidad de desplazamientos físicos del usuario, y esto atañe no sólo a las colecciones o a la información que se gestiona, sino a la interacción con el usuario (22).

Biblioteca electrónica

En un artículo titulado “Los prototipos y la terminología”, se dice que la biblioteca electrónica es la que se encuentra ricamente dotada de equipos micro-electrónicos y de instalaciones de telecomunicaciones, que permitirán acceder a la información en formato electrónico *‘in situ’* o a larga distancia; y en las colecciones de estas bibliotecas convivirán todo tipo de materiales y formatos (22).

A pesar de la diversidad de criterios, varios autores consideran que los términos bibliotecas digitales, virtuales o electrónicas pueden ser manejados como un único concepto, a pesar de que a veces se discute sobre que término posee mejor acogida (22). Pues las bibliotecas digitales y virtuales no poseen diferencias sino que son un conjunto controlado de materiales multimedia en formato digital, concebido para el beneficio de sus usuarios, estructurado de forma que facilite el acceso a sus contenidos y equipado con ayudas a la navegación en la red mundial (22).

Estos conceptos enmarcan diferentes formas de manejar la información; pero eventualmente todos se relacionan entre sí, por el hecho de que todos hacen un uso extensivo de las tecnologías y les permiten a sus usuarios una vía de acceso a la información.

Biblioteca de componentes

Según los conceptos antes expuestos se puede entender, entonces, como una biblioteca de componentes a toda colección organizada de componentes que sirvan de soporte para la confección de nuevos productos o componentes por parte de sus usuarios.

Al analizar estas formas de desarrollo de software, se denota la existencia de un problema, que sería el cómo almacenar cada uno de esos componentes creados, para así poder reutilizarlos cuando fuese necesario; para dar solución a este problema surgen las bibliotecas como método para almacenar y catalogar estos elementos. Dentro de las bibliotecas existen variadas opciones como: las bibliotecas digitales, las electrónicas y las virtuales; aunque algunos autores coinciden en que todos representan el mismo sistema de archivos en línea. Dentro de estas variedades de bibliotecas se encuentran entonces, las bibliotecas de componentes que sirven como soporte a las metodologías de software basado en componentes pues son muy empleadas en la actualidad para recuperación y almacenamiento de componentes de software, lo cual da paso además, a la ejecución de búsquedas especializadas que favorecen al usuario final y el proceso de creación de nuevas aplicaciones.

1.3 Sistemas homólogos a nivel mundial

En el mundo existen bibliotecas de componentes enfocadas diferentes temáticas, por lo que es necesario realizar un estudio de las mismas, con el objetivo de analizar dichos sistemas y comprender la relación de similitud que puede o no tener con la propuesta de solución a presentar.

1.3.1 Biblioteca de Componentes Visuales

La Biblioteca de Componentes Visuales o VCL (por sus siglas del inglés *Visual Components Library*) es un marco de trabajo visual y orientado a objetos cuyo objetivo final es crear clases que representan a componentes; aunque algunas clases no hagan referencia a componentes concretos sino que realizan tareas de gestión interna y se emplean como clases bases para derivar mediante herencia otras clases. Esencialmente está estructurado por clases (24). El contenido de los componentes aparece en una lista estructurada y ordenada por páginas, además de las propiedades, métodos y eventos de cada componente, lo que permite editar cada componente a través de sus propiedades.

Dentro de sus limitaciones se encuentra que cada componente debe crearse de forma dinámica y no permite la herencia múltiple. Está destinada al trabajo con componentes de programación por lo que se desestima como modelo a seguir para la confección de la biblioteca de componentes multimedia.

1.3.2 jQuery UI

jQuery UI es una biblioteca de componentes para el *framework jQuery*. La biblioteca de componentes de *jQuery* UI es una colección de componentes muy completa que permite la creación de diferentes *widgets*, efectos, interacciones. Dentro de las ventajas de utilizar la biblioteca de *jQuery* UI se encuentra la visualización del funcionamiento de cada componente, lo que significa gran ayuda para los usuarios finales, pues una vez visualizado el funcionamiento, puede seleccionar el componente más a fin con sus necesidades. También posibilita la visualización del código HTML perteneciente a cada componente, por lo que un usuario puede gestionar el funcionamiento de dicho componente a través de personalizaciones al código del mismo, de acuerdo con sus necesidades (25).

La principal desventaja de este sistema radica en que para agregar nuevos componentes hay que esperar que una nueva versión sea liberada por los creadores, tampoco permite almacenar los componentes creados por los usuarios.

1.3.3 Diseño Asistido por Computador de Sistemas de Potencia

PSCAD (Diseño Asistido por Computador de Sistemas de Potencia o *Power System* CAD por su siglas en inglés) es una librería que cuenta con más de 280 componentes flexibles y validados, permite que los usuarios construyan avanzados modelos no lineales del sistema de potencia al combinar generación, transporte, electrónica de potencia, distribución, y los más importantes circuitos de control de en uno o varios modelos de gran tamaño (26). Este software está organizado a partir de los módulos *File Manager*, *Draft*, *T-Line*, *Cables*, *Runtime* y *Multiplot*. El trabajo con estos módulos permite la simulación de sistemas de potencia y al mismo tiempo posibilita graficar, dibujar, ordenar y calcular los datos obtenidos en la simulación (27). El empleo de PSCAD permite el uso de componentes tales como (26):

- Resistencias (R), inductancias (L), y condensadores (C) de valor fijo y variable.
- Fuentes de corriente y tensión controlables.
- Conmutadores, interruptores y faltas programables.
- Convertidores electrónicos de potencia y circuitos de control.

- Búsqueda del mejor caso en optimización de múltiples variables de forma no lineal.
- Componentes de secuencia para el control de operaciones de simulación, etc.

El alcance de esta biblioteca se limita a componentes de software relacionados con la electrónica, por lo que no contribuye con el objetivo de la actual investigación.

1.4 Sistemas homólogos en Cuba

En Cuba, existen pocos sistemas basados en el trabajo de las bibliotecas de componentes. A pesar de esto, se puede establecer una comparación con la Biblioteca de componentes de interfaz de usuario para el Sistema de Identificación Cubano, que tiene por objetivo el almacenamiento de los componentes comunes para la construcción de las interfaces de usuario del Sistema de Identificación, Inmigración y Extranjería de la República de Cuba, que garantice la agilización del proceso de desarrollo, así como la uniformidad y la calidad en la presentación de las mismas. Para el desarrollo del mismo se empleó la metodología de desarrollo *MSF for CMMI Process Improvement,* además *Altova UModel 2009 Enterprise Edition* como herramienta de modelado con UML por lenguaje, el sistema está desarrollado en .NET (28).

Atendiendo lo anteriormente planteado se determina que el sistema no contribuye al objetivo actual de la investigación.

1.5 Sistemas homólogos en la UCI

En la Universidad de las Ciencias Informáticas (UCI) se ha analizado el tema mediante las investigaciones desarrolladas como Trabajos de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas. Existen varios sistemas desarrollados que responden a esta temática.

1.5.1 Biblioteca de componentes para la implementación de la interfaz web de eXcriba

La Biblioteca de componentes para la implementación de la interfaz web del Gestor de Documentos Administrativos eXcriba, es una biblioteca de componentes relacionada al proceso de desarrollo de interfaces web; la misma posee componentes tales como (10):

- Explorador (*explorer*).
- Ventana emergente.
- Acordeón.
- Paginador.

- Formulario.
- Tablas.
- Vista de árbol (*treeview*).
- Examinar.
- Región de búsqueda.

El sistema se encuentra estructurado a partir de módulos tales como: *excriba-core, x-acordeón*, *x-audit, x-basic-operation*, *x-breadcrumb*. Permite gestionar los componentes genéricos (10).

Dentro de las desventajas que posee este sistema se encuentra que no cuenta con soporte para las tendencias más actuales de la tecnología, por lo que no contribuye al objetivo de la presente investigación.

1.5.2 Biblioteca de componentes de software

La Biblioteca de componentes de software, es una biblioteca con funcionalidades que permiten la gestión (adición, eliminación, búsqueda y descarga) de cada uno de los componentes de software que estén presentes en la misma, además de recopilar información sensible de los mismos. El acceso a la misma se realiza a partir de roles con que cuenta cada usuario. El sistema permitirá modificar los componentes que se encuentren almacenados y a su vez enviará una notificación al creador del mismo para que tenga constancia de que su componente fue modificado; además se podrán gestionar componentes de software que pertenezcan a diferentes tecnologías (29).

A pesar de que este sistema puede soportar la gestión de componentes con diferentes tecnologías se encuentra centrado en el proceso de desarrollo de software de forma general y no en el de aplicaciones con tecnología multimedia. Esto supone que habría que rediseñar todo el sistema y cambiar parte de su estructura interna. Por otra parte, el análisis desarrollado permitió que el equipo de desarrollo tuviera una idea clara de cómo funciona el proceso de gestión de componentes en una biblioteca de este tipo.

Análisis de los sistemas estudiados

Luego de un análisis de las características principales de varias bibliotecas de componentes de diferentes temáticas, se determinó que algunos de dichos sistemas, no están acordes a las tendencias actuales del uso de las tecnologías. Además, los que responden a este problema, carecen de funcionalidades que permitan la personalización de los componentes que en ellos se encuentran y los que cuentan con esta

funcionalidad, necesitan de una nueva versión del sistema para poder incluir los nuevos elementos. De todas las bibliotecas estudiadas ninguna hace referencia al trabajo con componentes multimedia y su re-adaptación para el desarrollo de esta línea, elevaría el costo tan alto que se hace imposible su aplicación. Tomando lo anterior expuesto se llegó a la conclusión que dichos sistemas no satisfacen la necesidades del Centro, pues no cuentan con los requerimientos específicos que se necesitan. Sin embargo, es apreciable su aporte en cuanto a la estructura y el trabajo con los componentes que se desarrollan.

La Biblioteca de componentes para el desarrollo de aplicaciones con tecnología multimedia del centro FORTES, permitirá organizar y centralizar la documentación y los componentes para el proceso de elaboración de aplicaciones con este tipo de tecnología y no se limita a una biblioteca de componentes como los sistemas anteriormente estudiados sino que posibilitará además la personalización de los componentes almacenados en el sistema, el control de las versiones de cada uno de dichos componentes, además de la opción de una biblioteca personal donde el usuario almacene los componentes de su interés.

1.6 Tendencias

Años atrás, el entorno multimedia requería de dispositivos analógicos, pero la evolución de la tecnología digital ha permitido una amplia difusión de nuevos tipos de dispositivos, así como del uso de aplicaciones diseñadas para los mismos. La tendencia del desarrollo de productos con tecnología multimedia, está en desarrollarlos con diseños adaptativos (web *responsive*), es decir, para todos los tipos de dispositivos, que van desde las computadoras hasta los dispositivos móviles (Tabletas, teléfonos celulares). La adaptación de los productos multimedia a dispositivos móviles implica que tengan que consumir menos recursos de hardware y se puedan visualizar en cualquier dispositivo independientemente de la marca, modelo o generación.

La implementación de productos multimedia con el lenguaje *ActionScript3.0*, utilizado por el entorno de *Adobe* (dígase *Macromedia Flash*, *Adobe Flash CS6*), imposibilita su reproducción en los dispositivos móviles debido a su alto consumo de memoria RAM que necesita para ejecutar las animaciones. Otra de las limitantes que presenta el uso de Flash, es que no todos los navegadores y sistemas operativos - como las distribuciones de *Linux* y *Apple* - soportan este tipo de tecnología. Por lo que, desde que comenzó a distribuir sus productos en la web, *Flash* creó complementos como el *Flash Player* para suplir esta

deficiencia. En el 2013 es dada la noticia de que *Adobe* iba a dejar de darle soporte al complemento *Flash Player*.

Emergiendo y quedando como reemplazo de *Flash*, se presentan en unión inseparable, el Lenguaje de Marcado de Hipertexto en su versión 5 (HTML5 por sus siglas en inglés) y las Hojas de Estilo en Cascada en su versión 3 (CSS3 por sus siglas en inglés). Estas tecnologías web hacen posible que las aplicaciones puedan ser visualizadas en una gran variedad de dispositivos móviles, sistemas operativos y navegadores. Además resuelven los problemas que tenía Flash de ser altamente consumidor de los recursos de hardware del dispositivo.

Analizando en la herramienta *Google Trends*, se pudo realizar un seguimiento a las tendencias de empleo de estos lenguajes (Ver análisis en ANEXO 1), teniendo en cuenta los criterios de publicaciones sobre estos temas, la evolución en el tiempo del interés por cada lenguaje y las visitas a los sitios que abordan las temáticas relacionados con estos para la creación de productos multimedia. En la gráfica que se muestra a continuación en la figura 1, se puede apreciar una comparación de las variaciones que han sufrido cada uno de estos lenguajes. Las tendencias del uso del lenguaje *ActionScript3* en diciembre del 2013 fueron en descenso hasta llegar casi a 0, dando un grado de la baja usabilidad de este lenguaje. *Adobe Flash* presenta una situación similar, aunque no es mostrado en el gráfico, pues su empleo viene en retroceso desde el 2005. Por el contrario, el uso del lenguaje HTML5 ha aumentado desde la segunda mitad del 2009. Al analizar de manera general el gráfico con los 3 lenguajes (*ActionScript3*, HTML5 y CSS3) se puede llegar a la conclusión parcial que una de las tendencias actuales es al uso de estos lenguajes (HTML5 y CSS3) para el desarrollo de aplicaciones orientadas a la web y a la diversidad de formas de visualizarlas en los dispositivos tecnológicos que están en el mercado.

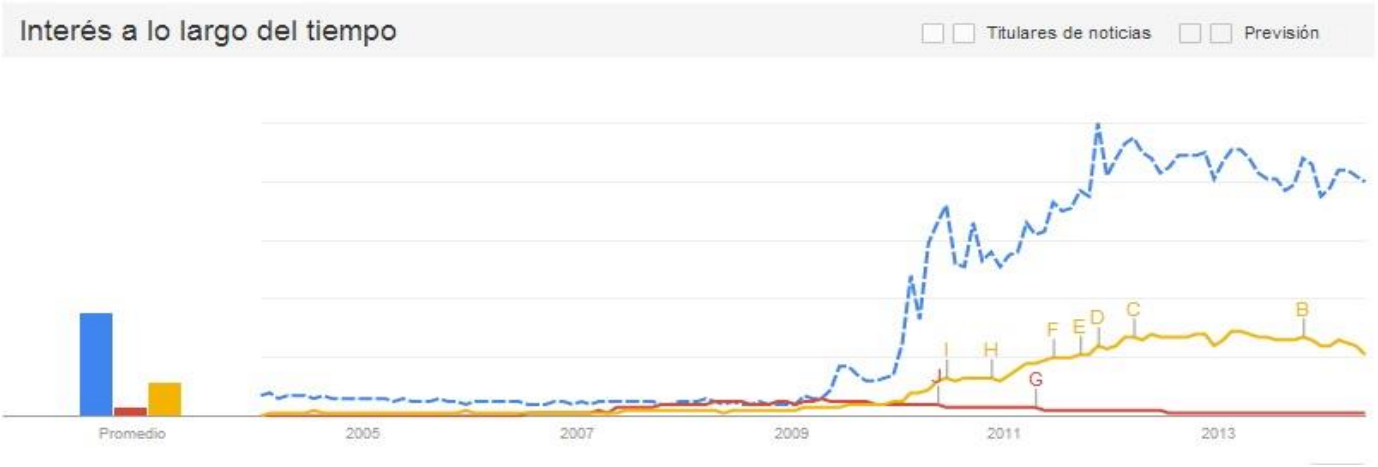


Fig.1 Tendencia de los lenguajes HTML5 (azul), ActionScript3 (rojo) y CSS3 (amarillo) (30)

Según el sitio web de Hewlett Packard (HP) para América Central, algunas de las principales tendencias de empleo de contenido multimedia son (31):

*"*Los consumidores quieren acceder al contenido en cualquier dispositivo, en cualquier lugar. El público está adoptando un estilo de vida de consumo de medios multiplataforma donde esperan tener la misma experiencia de usuario de alta calidad, sin importar la forma de acceder a los contenidos. En este entorno, cada vez más personas acceden a estos contenidos en dispositivos móviles. Hoy en día se está generando más contenido en alta definición, con resoluciones cada vez mayores y velocidades de cuadro más altas. Esto aumenta considerablemente el volumen de datos que se está creando".

Actualmente el desarrollo de software está encaminado a la reutilización de componentes, por las ventajas que representa para las industrias de software. La programación basada en componentes hace que las aplicaciones sean fáciles de extender, reduce los costos y el tiempo de producción, siendo punto clave en la mayoría de las empresas dedicadas al software (31).

Las bibliotecas de componentes permiten a gran escala la reutilización de componentes para los procesos productivos de las empresas, se hace difícil pensar que las herramientas para el almacenamiento, clasificación y recuperación de componentes no puedan ser utilizadas a través de la red Internet mediante un servidor Web. La utilización de la flexibilidad y el poder que ofrece la tecnología web debe aprovecharse para crear un entorno que de soporte al desarrollo con y para la reutilización de forma distribuida (32).

1.7 Metodología de desarrollo de software

Las metodologías de desarrollo de software no son más que aquella filosofía, procedimientos o estándares y técnicas que se van a emplear en el proceso de confección de un software. Las metodologías conocidas en la actualidad son clasificadas en dos grandes grupos: las metodologías tradicionales y las metodologías ágiles; las primeras pensadas sobre el uso exhaustivo de la documentación sobre el ciclo de desarrollo del producto y las otras, centradas en la capacidad de resolver las necesidades del cliente atendiendo a cambios continuos en los requisitos (23).

1.7.1 Metodologías tradicionales

Se centran en la definición detallada de los procesos y tareas a realizar, herramientas a utilizar, y requiere una extensa documentación, ya que pretende prever todo de antemano. Este tipo de metodologías son más eficaces y necesarias cuanto mayor es el proyecto que se pretende realizar respecto a tiempo y recursos que son necesarios emplear, donde es requerida una gran organización (33). Dentro de las metodologías tradicionales se encuentran *Rational Unified Procces* (RUP), *Microsoft Solution Framework* (MSF), entre otras. Este tipo de metodología se limita ante los cambios continuos que se puedan producir durante el proyecto; necesita un equipo relativamente grande para el desarrollo de software y este generalmente está distribuido en áreas separadas, el uso de estas metodologías implica la generación de mayor cantidad de artefactos, además se requiere gran cantidad de roles (34).

Por las características antes expuestas se rechaza este tipo de metodología para el desarrollo de la biblioteca de componentes multimedia.

1.7.2 Metodologías ágiles

Principales ideas de la metodología ágil (33)**:**

- Se encarga de valorar al individuo y las iteraciones del equipo más que a las herramientas o los procesos utilizados.
- Se hace mucho más importante crear un producto software que funcione que escribir mucha documentación.
- El cliente está en todo momento colaborando en el proyecto.
- Es más importante la capacidad de respuesta ante un cambio realizado que el seguimiento estricto de un plan.

SCRUM

Define un marco para la gestión de proyectos. Está especialmente indicada para proyectos con un rápido cambio de requisitos. El desarrollo de software se realiza mediante iteraciones, denominadas *sprints*, con una duración de 30 días. El resultado de cada *sprint* es un incremento ejecutable que se muestra al cliente (35). Otra característica importante son las reuniones a lo largo del proyecto, entre ellas destaca la reunión diaria de quince minutos del equipo de desarrollo para coordinación e integración, lo que la convierte en una metodología de desarrollo simple que se basa en su adaptabilidad a los diferentes cambios que puede experimentar el proceso de evolución del proyecto (36).

SCRUM es una metodología ágil, y como tal (36):

- Es un modo de desarrollo de carácter adaptable más que predictivo.
- Orientado a las personas más que a los procesos.
- Emplea la estructura de desarrollo ágil incremental basada en iteraciones y revisiones.

Crystal Methodologies

Se trata de un conjunto de metodologías para el desarrollo de software caracterizadas por estar centradas en las personas que componen el equipo (de ellas depende el éxito del proyecto) y la reducción al máximo del número de artefactos producidos. Han sido desarrolladas por Alistair Cockburn. El desarrollo de software se considera un juego cooperativo de invención y comunicación, limitado por los recursos a utilizar. El equipo de desarrollo es un factor clave, por lo que se deben invertir esfuerzos en mejorar sus habilidades y destrezas, así como tener políticas de trabajo en equipo definidas. Estas políticas dependerán del tamaño del equipo, estableciéndose una clasificación por colores, por ejemplo *Crystal Clear* (3 a 8 miembros) y *Crystal Orange* (25 a 50 miembros) (35).

XP (eXtreme Programming)

Es la metodología ágil que se encuentra centrada en potenciar las relaciones interpersonales como clave fundamental para el éxito en el desarrollo del software, promoviendo especialmente el trabajo en equipo y la preocupación por el aprendizaje de los desarrolladores. La programación extrema se basa en la simplicidad, la comunicación y el reciclado continuo de código. Es basada también en la retroalimentación continua entre el cliente y el equipo de desarrollo, una comunicación fluida entre todos los participantes en el desarrollo, además de la simplicidad en las soluciones implementadas. Esta metodología se define

especialmente para proyectos con requisitos imprecisos y cambiantes pues alienta a los desarrolladores a responder a dichos cambios aún en fases tardías del ciclo de vida del desarrollo (35). XP pretende conseguir una curva de coste del cambio con crecimiento leve, que en un inicio es más costosa, pero que a lo largo del proyecto permite tomar decisiones de desarrollo lo más tarde posible sin que esa nueva decisión de cambio implique un alto coste en el proyecto. La metodología define cuatro variables para cualquier proyecto de software: costo, tiempo, calidad y alcance; de estas, sólo tres podrán ser fijadas por el cliente y/o por el jefe de proyectos, la cuarta es responsabilidad del equipo de desarrollo y se establecerá en función de las otras tres (37). Cuenta con una abundante documentación referente a las fases que traza esta metodología.

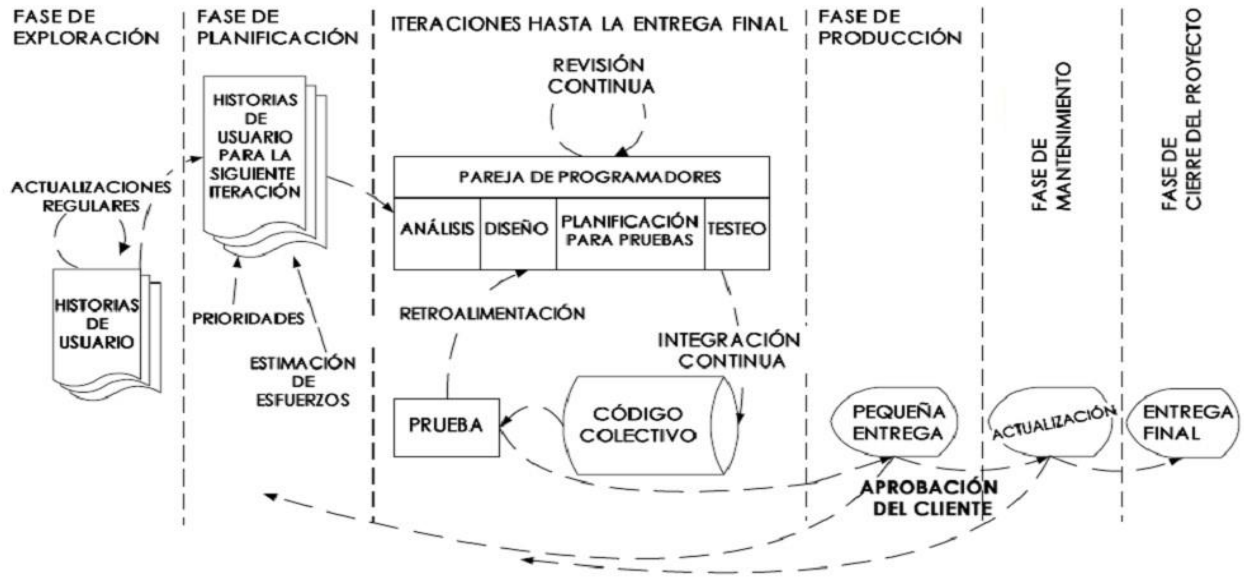


Fig.2 Proceso de vida del software en la Metodología XP

Selección de la metodología de desarrollo de software a utilizar

Después de analizadas las metodología de desarrollo de software, se decide utilizar *eXtreme Programming* como metodología que guiará el ciclo de desarrollo de la biblioteca de componentes, porque se emplea para proyectos de corto plazo como el de la presente investigación. Presenta una estructura de roles adaptable al equipo de desarrollo y a requisitos cambiantes. Fomenta la continua retroalimentación entre los programadores y el cliente, este último se encuentra disponible en el proyecto, planificando las entregas de cada iteración, guiando constantemente el trabajo, estableciendo la prioridad de las HU a implementar y

determinando que HU están terminadas. Utiliza las tareas de ingeniería para que la implementación de la biblioteca no se detenga al ausentarse uno de los miembros del equipo, pues las funcionalidades están bien descritas y pueden ser ejecutadas por otro miembro del equipo. XP es ideal para pequeños equipos de desarrollo, en el caso de esta investigación es un equipo de 2 programadores adaptándose bien con una de las prácticas de XP que es la programación en parejas. Es una metodología bien documentada y dentro de las metodologías ágiles es de las más empleadas en la Universidad. Como parte de la presente investigación, el equipo de desarrollo en conjunto con el cliente, determinó que se abarcarán para el desarrollo de la misma solamente las fases de Exploración, Planificación, Producción y Pruebas.

Se rechaza *Crystal* por presentar limitaciones como el desarrollo de proyectos con un alto componente crítico, es decir proyectos con altos factores de riesgo como requisitos cambiantes y limitaciones con el tiempo. SCRUM es desechada porque tiene el rol llamado *SCRUM Master* para resolver problemas, no para manifestar que se debe cumplir con una tarea o guiar un proceso específico. Se requiere de un experto en la metodología que monitorice su cumplimiento. A veces es necesario complementarlo con otros procesos de XP.

1.8 Herramientas y tecnologías

En el desarrollo de aplicaciones se necesitan un conjunto de herramientas y tecnologías que apoyen las fases de la metodología seleccionada. En los subepígrafes siguientes se hace un análisis de varias de estas herramientas y tecnologías, además se seleccionan las que serán utilizadas en el desarrollo de la biblioteca de componentes.

1.8.1 Sistema gestor de contenidos

Un sistema gestor de contenidos (*Content Management System*, abreviado CMS) es una herramienta que se emplea en la creación, administración y publicación de sitios web. Estos cubren el ciclo de vida de las páginas de un sitio, permitiendo manejar su estructura, en aspectos relevantes como la creación de contenidos, actualización, administración de la información y la navegación de los usuarios (38).

Joomla 2.5

Joomla es un sistema gestor de contenidos (CMS) dinámicos que permite crear sitios web de alta

interactividad, profesionalidad y eficiencia. Su administración está enteramente basada en la gestión *online* (en línea) de contenidos (39).

Joomla se caracteriza por (39):

- Extenso trabajo en estandarización de estilo y consistencia del código.
- Incorpora *Bootstrap* en el paquete *jui*.
- Posee *driver* para *PostgreSQL* y PHP *Memcached*.
- Instalación de paquetes de idioma directamente desde el gestor de extensiones.
- *TinyMCE* actualizado a la versión 3.5.6.
- *Tests* de sistema en el CMS actualizado.

WordPress 3.3.1

WordPress es una avanzada plataforma semántica de publicación personal orientada a la estética, los estándares web y la usabilidad (40). En sus inicios fue orientado al desarrollo de *blogs*, pero ha evolucionado hacia uno de los CMS de uso general más completos. Es de los CMS más conocidos y utilizados, debido a la sencillez de su uso e implantación, dirigido fundamentalmente a sitios web de comunicación y donde se requiere una presencia básica en Internet. Se encuentra liberado bajo una licencia GPL y utiliza PHP como lenguaje de programación, *MySQL* como motor de base de datos y Apache o *Nginx* como servidor web (41).

Dicho CMS contiene además otras características(41)(42):

- Generación de contenidos extremadamente fácil.
- Extensa librería de *plugins*.
- Protección de contenidos por contraseña, filtros *antispam* o controles de comentarios.
- Administrado por múltiples autores, con una interfaz fácil y amigable para el usuario.
- Posibilita el desarrollo web con aspecto y características absolutamente profesionales.
- Fácil de adaptar para casi cualquier proyecto web, ya que puedes crear un blog o incluso portafolios.
- Dispone de buen soporte a través de abundante documentación y foros, es una de las comunidades más dinámicas en el contexto de los Sistemas de Gestión de Portales Web.

Drupal 7.24

Drupal es un CMS que se distribuye como software libre bajo licencia GNU GPL (*General Public License*) versión 2 o superior. El software está desarrollado con el lenguaje de programación PHP y utiliza las bases de datos *MySQL*, *SQLite*, *MaríaDB*, *PostgreSQL*. Permite desarrollar cualquier tipo de portal o aplicación web. Además de las funcionalidades básicas que vienen integradas en el software, es posible añadir nuevas funcionalidades a través de módulos (43).

Dicho CMS presenta potencialidades que hacen de este, uno de los más utilizados a nivel mundial (44):

- Comunidad de desarrollo: *Drupal* cuenta con una amplia comunidad de usuarios, los cuales se mantienen implementando nuevas versiones y módulos para lograr el crecimiento del CMS.
- Carácter modular: basado en la inclusión de módulos elaborados por la comunidad de desarrolladores que proporcionan las más disímiles funcionalidades.
- Personalización: un robusto entorno de personalización está implementado en el núcleo de *Drupal*.
- Tanto el contenido como la presentación pueden ser individualizados de acuerdo a las preferencias definidas por el usuario.
- Control de versiones: el sistema de control de versiones de *Drupal* permite seguir y auditar totalmente las sucesivas actualizaciones del contenido.
- Plantillas (*Templates*): el sistema de temas de *Drupal* separa el contenido de la presentación permitiendo controlar o cambiar fácilmente el aspecto del sitio web. Se pueden crear plantillas con HTML y/o con PHP.
- Sindicación del contenido: *Drupal* exporta el contenido en formato RDF/RSS para ser utilizado por otros sitios web.
- Sistema para el Control de Asistencia integrado al CMS *Drupal*.
- Drupal incorpora una “capa de abstracción de base de datos”.
- Multiplataforma: *Drupal* ha sido diseñado desde el principio para ser multiplataforma. Puede funcionar con Apache o *Microsoft* IIS como servidor web y en sistemas como *GNU&Linux*, BSD, *Solaris*, *Windows* y *Mac OS X*. Por otro lado, al estar implementado en PHP, es totalmente portable.
- Administración vía web: la administración y configuración del sistema se puede realizar enteramente con un navegador y no precisa de ningún software adicional.

Selección del sistema gestor de contenido a utilizar

Joomla tiene la desventaja que es solo para la creación de sitios web, no permite la implementación de módulos, lo que limita el desarrollo de productos personalizados. Además, no cuenta con una comunidad de desarrollo en la UCI. Recién incorpora en la versión 3 un driver de *PostgreSQL*. *Wordpress* por ser el CMS más utilizado tiene la desventaja de estar constantemente bajo ataques de piratas informáticos, debilitando la seguridad del gestor de contenido. Cada vez que se publican nuevas actualizaciones para arreglar agujeros de seguridad, los *plugins* antiguos son incompatibles. En cambio, *Drupal* es un CMS que va más allá de las potencialidades de un gestor de contenidos, permite estructurar una aplicación web, al incluir adaptabilidad, eficiencia y seguridad en sus aplicaciones, no limita su área de aplicación. Una de sus grandes facilidades es que incorpora una "capa de abstracción de base de datos" permitiendo que sus módulos puedan ser instalados independientemente del gestor base de datos. Administra de forma fácil y sencilla los contenidos que almacena. El equipo tiene un amplio dominio en el desarrollo de aplicaciones con dicho CMS. La UCI cuenta con un portal para la comunidad de *Drupal*, el cual ofrece una comunicación con personas que tienen experiencia en el uso de este CMS. Por las ventajas que tiene *Drupal* se utilizará la versión 7.24 del mismo para el desarrollo de la biblioteca de componentes.

1.8.2 Sistema gestor de base de datos

Un sistema gestor de base de datos (SGBD) es un sistema de software que permite la definición de bases de datos; así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación (34). A continuación se analizan algunos sistemas gestores de base de datos.

MySQL 5.6.11

MySQL es un sistema de gestión de bases de datos relacional, es privativo aunque tiene una versión limitada bajo la licencia GPL de la GNU. Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. Se ha enfocado tradicionalmente en aplicaciones web de lectura, usualmente escritas en PHP, donde la principal preocupación es la optimización de consultas sencillas. Su característica fundamental es su velocidad a la hora de realizar las operaciones, lo que le hace uno de los gestores que ofrecen mayor rendimiento. Tiene una probabilidad muy reducida de corromper los datos, incluso en los casos en los que los errores no se produzcan en el propio gestor, sino en el sistema en el que está (45).

PostgreSQL 9.1

*PostgreSQ*L es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado. Utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando (46).

Características de *PostgreSQL* (47)(48):

- Instalación ilimitada. No hay que pagar licencia para su uso como otras herramientas.
- Soporte. Además de las ofertas de soporte, se cuenta con una comunidad importante de profesionales y entusiastas de *PostgreSQL* de los que su compañía puede obtener beneficios.
- Ahorros considerables en costos de operación. *PostgreSQL* ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que otros productos, conservando todas las características, estabilidad y rendimiento.
- Extensible. El código fuente está disponible para todos sin costo. Si el equipo de desarrollo necesita extender o personalizar *PostgreSQ*L de alguna manera, pueden hacerlo con un mínimo esfuerzo, sin costos adicionales.
- Multiplataforma. *PostgreSQL* está disponible en la mayoría de plataformas como *Windows* y *Linux* fundamentalmente.
- Diseñado para ambientes de alto volumen. *PostgreSQL* usa una estrategia de almacenamiento de filas llamada MVCC para conseguir una mejor respuesta en ambientes de grandes volúmenes.

Selección del sistema gestor de base de datos a utilizar

Después de haber analizado los gestores de base de datos *MySQL* y *PostgreSQL* se decide utilizar este último por las características antes mencionada, además de que ha sido utilizado en el desarrollo de aplicaciones anteriores por los autores de la presente investigación; se considera que presenta funcionalidades esenciales para el desarrollo de la biblioteca de componentes como los *triggers*, donde estos implican una respuesta más rápida por parte de las bases de datos; se cuenta además con la presencia de la Comunidad Técnica Cubana de *PostgreSQL* y un centro especializado en dicho gestor dentro de la UCI. Se utilizará *PostgreSQL* en su versión 9.2 y el cliente *PgAdmin* en su versión 1.16.1.

1.8.3 Servidor web

Un servidor web es un programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten usando el protocolo *Hypertext Transfer Protocol* (HTTP) o el protocolo HTTPS (la versión cifrada y autenticada) (34).

Internet Information Services (IIS) 6.0

Es una aplicación de servidor web y un conjunto de módulos de extensión de funciones creadas por *Microsoft* para su uso con *Microsoft Windows*. Brinda servicios de software que admiten la creación, configuración y administración de sitios Web, además de otras funciones de Internet. Soporta HTTP, HTTPS, SMTP y NNTP (49).

Características de IIS (50):

- Las características agregadas en seguridad se aprovechan de las últimas tecnologías de cifrado y métodos de autenticación mediante certificados de cliente y servidor.
- IIS tiene la forma de asegurar los datos es mediante SSL (*Secure Sockets Layer*). Esto proporciona un método para transferir datos entre el cliente y el servidor de forma segura, permitiendo también que el servidor pueda comprobar al cliente antes de que inicie una sesión de usuario.
- La autenticación implícita permite a los administradores autenticar a los usuarios de forma segura a través de servidores de seguridad y *proxy*.
- IIS también es capaz de impedir que aquellos usuarios con direcciones IP conocidas obtengan acceso no autorizado al servidor, permitiendo especificar la información apropiada en una lista de restricciones.
- En cuanto a la seguridad, IIS tiene integrado el protocolo Kerberos v5.

Apache 2.2

Apache 2.2 es un servidor web de software libre desarrollado por la Apache Software Foundation (Fundación Apache Software) cuyo objetivo es servir o suministrar páginas web (en general, hipertextos) a los clientes web o navegadores que las solicitan (49). Está diseñado para ser un servidor web potente y flexible que pueda funcionar en la más amplia variedad de plataformas y entornos.

Entre estas características se encuentran (51):

- Soporta PHP5.
- Es altamente configurable, rápido, flexible, eficiente y adaptado a los nuevos protocolos web.
- Su condición de ser multiplataforma permite mantener este servidor web en todos los sistemas operativos en los que se despliegue el software.
- Brinda todo el código fuente y tiene una licencia libre de restricciones.
- Es un servidor seguro que permite protección de ficheros.
- Permite la configuración de mensajes de errores personalizados.

Selección del servidor web a utilizar

Después de enunciar las características de los servidores web IIS y Apache se decide utilizar Apache en su versión 2.2 por las características antes mencionadas, tiene soporte para el protocolo de seguridad HTTPS, es muy utilizado en el desarrollo web, actualmente los paquetes de servidores (WAMP, XAMPP) lo traen incorporado como servidor web, incluye comunicación con el protocolo IPv6, se puede configurar un *hosting* virtual, permite la integración con varios gestores de base de datos. Se desecha la posibilidad de usar el IIS porque no se puede ejecutar en Linux al no ser multiplataforma, debe ser usado pagando una licencia y está dirigido mayormente a la tecnología de *Microsoft* como los son los sitios basados en ASP.

1.8.4 Lenguajes de programación

Un lenguaje de programación es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes (52).

Lenguajes de programación del lado del cliente

HTML5

HTML5 no es solo la última versión de la especificación HTML, también es un término general que describe un conjunto de tecnologías relacionadas que se utilizan para hacer moderno y rico el contenido web. La especificación principal HTML5 define el uso de elementos para etiquetar el contenido, indicando su importancia. Atiende las necesidades de las aplicaciones web. Posibilita utilizar, manipular y almacenar una imagen de la cámara del ordenador así como comunicarse con el servidor de formas nuevas e innovadoras; permite a las páginas web almacenar datos localmente en el lado del cliente y operar fuera de línea de

manera más eficiente, además de combinar audio y video en la web abierta. Proporciona una mayor optimización de la velocidad y un mejor uso del hardware del equipo. Tiene la ventaja de estar mejor estructurado que sus versiones anteriores. Utiliza la etiqueta canvas para la representación gráfica vectores, muy importante para el dibujo en 2D y 3D (53).

Se utilizará dicha versión, para implementar los componentes con tecnología multimedia que serán añadidos a la biblioteca y para maquetar la información que se mostrará en la misma.

CSS3

Hojas de Estilo en Cascada (***C****ascading* ***S****tyle* ***S****heets*) es el lenguaje utilizado para describir la presentación de documentos HTML o XML, esto incluye varios lenguajes basados en XML como son XHTML o SVG. Además, es usado para modificar la interfaz de usuario de aplicaciones y programas, este es el caso de los productos basados en XUL como son *Firefox*, *Seamonkey* o *Thunderbird* (54). El propósito básico de CSS es permitir al diseñador definir detalles de estilo, declaraciones de formato como fuentes, tamaños de elementos y colores. Luego aplicar esos estilos a partes seleccionadas de páginas HTML usando selectores: referencias a un elemento o grupo de elementos a los que se aplica el estilo (55).

Se utilizará la versión 3 (CSS3) para darle estilo a los contenidos de la biblioteca así como a los componentes multimedia a desarrollar, ya que permite agregar efectos animados a objetos y cambiar la apariencia atendiendo a la personalización realizada por el usuario.

JavaScript

Lenguaje de programación interpretado. No es un lenguaje de programación orientada a objetos (POO) propiamente dicho como java o c/c++, pero sí es un lenguaje de programación basado en objetos (56). La gran mayoría de los sitios web modernos usan *JavaScript*. Los navegadores web recientes, las consolas de videojuegos, tabletas y teléfonos inteligentes incluyen intérpretes de *JavaScript*. Este lenguaje forma parte de la tríada de tecnologías que todos los desarrolladores web deben aprender: HTML para especificar el contenido de las páginas web, CSS para especificar la presentación de las páginas web y *JavaScript* para especificar el comportamiento de las páginas web. Además *JavaScript* es un lenguaje de alto nivel de programación dinámico que se adapta bien. Deriva su sintaxis de *Java*, superando desde hace mucho tiempo sus raíces de secuencias de comandos del idioma para convertirse en un lenguaje de propósito

general, robusto y eficiente (57). También permite el manejo de objetos dentro de la página web y sobre ese objeto se puede definir diferentes eventos. Dichos objetos facilitan la programación de páginas interactivas, a la vez que se evita la posibilidad de ejecutar comandos que puedan ser peligrosos para la máquina del usuario, tales como el formateo de unidades y la modificación de archivos (58).

En el desarrollo de la biblioteca se utilizará este lenguaje para las validaciones de datos, las animaciones y la ejecución de acciones en el lado del cliente, para no sobrecargar el servidor con peticiones innecesarias.

AJAX

AJAX (*Asynchronous JavaScript And XML* por sus siglas en inglés), no es un lenguaje de programación sino un conjunto de tecnologías tales como HTML, JavaScript, CSS, DHTML (*Dynamic* HTML por sus siglas en inglés), PHP, ASP.NET (*Active Server Pages* por sus siglas en inglés), JSP y XML que permiten hacer páginas de Internet más interactivas. La característica fundamental de AJAX es que permite actualizar parte de una página con información que se encuentra en el servidor sin tener que refrescar completamente la página. De modo similar su puede enviar información al servidor (59).

Ventajas de AJAX (59):

- Utiliza tecnologías ya existentes.
- Soportada por la mayoría de los navegadores modernos.
- Interactividad. El usuario no tiene que esperar hasta que lleguen los datos del servidor.
- Portabilidad. No requiere *plugins* como *Flash* y *Applet* de *Java*.
- Mayor velocidad debido a que no hay que retornar toda la página nuevamente.
- La página se asemeja a una aplicación de escritorio.

Esta tecnología se empleará para la escritura de archivos, el envío de variables del cliente al servidor y el mejoramiento de la velocidad del sitio, al no tener que recargar la página visualizada sino el área que se va a modificar.

Lenguajes de programación del lado del servidor

PHP 5.3

PHP (acrónimo de *Hypertext Preprocessor*) puede ser incluido con facilidad dentro del código HTML. Su código puede ser escrito en todos los sistemas operativos gratuitos del tipo *Unix*, como *Linux* y *FreeBSD*, versiones comerciales de *Unix*, como Solaris e IRIX o en las diferentes versiones de *Microsoft Windows* (60).

Entre sus características fundamentales están (60):

- Posee gran número de funciones predefinidas. A diferencia de otros lenguajes de programación. PHP está dotado de un gran número de funciones que simplifican enormemente tareas habituales, como descargar documentos, enviar correos, trabajar con cookies y sesiones, entre otras funciones.
- Dispone de una amplia gama de librerías, esto permite que PHP pueda ser utilizado en muchas áreas diferentes, tales como encriptado, gráficos, XML y otras; entre las que se encuentra la poderosa librería *PHPExcel*, la misma brinda gran cantidad de funcionalidades sobre el manejo con documentos *Excel*.
- Es un lenguaje de secuencia de comandos de servidor, diseñado específicamente para la Web. Dentro de una página Web puede incrustar código PHP que se ejecutará cada vez que se visite una página. El código PHP es interpretado en el servidor Web y genera código HTML y otro contenido que el visitante verá.
- Es un producto de código abierto, lo que quiere decir que se puede acceder a su código. Puede ser utilizado, modificado y redistribuido sin coste alguno.
- Dispone de una conexión propia a todos los sistemas de base de datos.

PHP cuenta con una extensa comunidad de desarrolladores a nivel mundial, lo cual facilita la colaboración y la gestión de soluciones a problemas ocurridos. Además en la UCI existe la comunidad de PHP que mantiene actualizada la abundante documentación que dispone. Por las características de este lenguaje y dado que *Drupal* está basado en PHP y sus módulos se desarrollan con este lenguaje, se utilizará el mismo en su versión 5.3 para la implementación de la biblioteca.

1.8.5 Entorno de desarrollo integrado

***I**ntegrated **D**evelopment **E**nvironment* (entorno de desarrollo integrado o IDE), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien poder utilizarse para varios. Un IDE puede denominarse como un entorno de programación que ha sido tratado como un programa aplicación. Esto significa que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (61).

PHPStorm 6.0.1

PhpStorm es un IDE ligero e inteligente enfocado en la productividad del desarrollador. Provee completamiento inteligente de código, navegación rápida y chequeo de errores al momento. Siempre está listo para dar forma al código, ejecutar *unit* o proveer *debugging* visual. Entre los principales beneficios de su uso se puede mencionar que es un editor PHP inteligente con (62):

- Completamiento de código PHP.
- Detector de código duplicado.
- Mezcla lenguajes (*JavaScript*, SQL, XML).

NetBeans PHP 7.3

NetBeans es un IDE desarrollado por *Sun Microsystems*, de código abierto y multiplataforma, hecho principalmente para el lenguaje de programación Java. Permite diseñar aplicaciones de forma fácil con solo arrastrar objetos a la interfaz de un formulario. Es una plataforma pensada para escribir, compilar, depurar y ejecutar programas. *NetBeans* permite a los desarrolladores crear rápidamente aplicaciones web, empresariales, de escritorio y aplicaciones móviles al utilizar la plataforma *Java*, así como *Ajax*, *Groovy* y *Grails*, y C / C + + (63).

La versión 7.3 del *NetBeans* PHP es un entorno dedicado a la codificación con dicho lenguaje (soporta hasta PHP 5.4), que permite la integración completa con los estándares web y brinda soporte completo para el desarrollo con HTML5, *JavaScript*, CSS3 y algunos de los *frameworks* de desarrollo más populares, como: Zend Framework y Synfony, entre otros (64).

Ventajas de *NetBeans* (63):

- Ofrece un mejor soporte a las últimas tecnologías de *Java* y de desarrollo web.

- Rápida edición de código.
- Administración de proyectos de forma fácil y eficiente.
- Tiene compilador de lenguajes y estructuras de sistemas.

Selección del entorno integrado de desarrollo a utilizar

La principal ventaja por la que se escoge el *Netbeans* PHP en la versión 7.3 para el desarrollo de sistema es que tiene soporte para funciones específicas del CMS *Drupal*, *Symfony* y otros *frameworks*, además el equipo de desarrollo de la presente investigación está familiarizado con dicha herramienta. El *PHPStorm* tiene la desventaja de ser alto consumidor de recursos de hardware (RAM), hasta ahora no se encuentra disponible en la universidad una versión para linux, además de que no es libre, pues hay que pagar licencia por su uso.

1.8.6 Marco de trabajo

Un marco de trabajo o *framework* es una estructura conceptual y tecnológica de soporte definida normalmente con artefactos o módulos de software específicos. Sobre su base puede ser organizado y desarrollado cualquier otro proyecto de software. Algunos pueden incluir soportes de programas, bibliotecas y lenguajes que sean interpretados dentro de otros programas, que puede ayudar a desarrollar y unir los diferentes componentes de un determinado proyecto (65).

ExtJS 3.0

Ext JS es un *framework JavaScript* ligero y de alto rendimiento, compatible con la mayoría de navegadores para crear páginas web y aplicaciones dinámicas. Cuenta con un amplio repositorio de componentes, una suite completa de documentación; proporciona las herramientas necesarias para construir aplicaciones de escritorio robustas. Trae un paquete de datos que permite a los desarrolladores utilizar el patrón modelo-vista-controlador (MVC) en la construcción de su aplicación. Es multiplataforma tiene soporte para diferentes versiones de navegadores (66).

jQuery 1.7

jQuery es un *framework* basado en el software libre y el código abierto que posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2 que le permite ser empleada tanto por software privativo como libre. Permite simplificar la manera de interactuar con los documentos HTML (Lenguaje de

Marcado de Hipertexto), manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web. Además ofrece una serie de funcionalidades basadas en *JavaScript* que, de otra manera requerirían de mucho más código; es decir, con las funciones propias de esta librería se logran grandes resultados en menos tiempo y espacio (67). Tiene la ventaja de que es un producto serio, estable, bien documentado y con un gran equipo de desarrolladores a cargo de la mejora y actualización del marco de trabajo. Tiene una amplia comunidad de creadores de *plugins* o componentes, lo que hace fácil encontrar soluciones ya creadas en *jQuery* para implementar interfaces de usuario, galerías, votaciones o diversos efectos (68).

Bootstrap 2.0

Bootstrap es un *framework* que simplifica el proceso de creación de diseños web combinando CSS y *JavaScript*. Permite crear interfaces con la particularidad de adaptar las mismas al tamaño del dispositivo en que se visualice; como los diferentes navegadores, tabletas y teléfonos móviles, a distintas escalas y resoluciones (69).

Symfony 2

Symfony2 es la versión más reciente de *Symfony*. Ha sido ideado para exprimir al límite todas las nuevas características de PHP 5.3 y por eso es uno de los marcos de trabajo PHP con mejor rendimiento. Su arquitectura interna está completamente desacoplada, lo que permite reemplazar o eliminar fácilmente aquellas partes que no encajan en el proyecto (70). Está basado en el clásico patrón de diseño web conocido como arquitectura MVC (Modelo-Vista-Controlador). Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Desarrollado completamente con PHP 5, ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. *Symfony* es compatible con la mayoría de gestores de bases de datos como *MySQL*, *PostgreSQL*, *Oracle* y *SQL Server* de *Microsoft*. Se puede ejecutar tanto en plataformas **nix* (*Unix*, *Linux*, etc.) como en plataformas *Windows* (71).

Selección de los marcos de trabajo a utilizar

La principal desventaja de *ExtJS* es que no está diseñado para pequeños proyectos. No presenta versiones compiladas, además se ha introducido recientemente en el desarrollo de aplicaciones con tecnología multimedia con HTML 5. Se selecciona *jQuery* en su versión 1.7 para trabajar los efectos de animación de

los componentes multimedia, este marco de trabajo permite que sea utilizado en software libre, es continuamente desarrollado por una amplia comunidad que genera *plugins* para aumentar sus funcionalidades. *Bootstrap* en su versión 2.0 se utilizará como marco de trabajo de diseño, especialmente para que los componentes multimedia se adapten a los dispositivos en que se visualizarán y para mejorar el diseño de las interfaces de dichos componentes. *Symfony* sin embargo presenta una curva de aprendizaje muy pronunciada lo que dificulta que en poco tiempo se logre un conocimiento medio de su filosofía y forma de trabajo, además de que la gestión de la seguridad resulta compleja para gran cantidad de desarrolladores.

1.8.7 Herramientas CASE

Según (72) las herramientas de Ingeniería de Software Asistida por Computadora (CASE) ayudan a los gestores y practicantes de la ingeniería de software en todas las actividades asociadas a los proyectos de software. Automatizan las actividades de gestión de proyectos, gestionan todos los productos de los trabajos elaborados a través del proceso y ayudan a los ingenieros en el trabajo de análisis, diseño y codificación.

Rational Rose Enterprise 7.0

Rational Rose Enterprise es una potente herramienta de modelado visual para ayudar en el análisis y diseño de software orientado a objetos. *Rational Rose*, proporciona un lenguaje común de modelado para el equipo, que facilita la creación de software de calidad más rápidamente (60).

Posee características adicionales incluidas (60):

- Basada en modelos que se integra con las bases de datos y los IDE de las principales plataformas del sector.
- Soporta el análisis de patrones ANSI C++, *Rose* J y *Visual* C++ basado en *"Design Patterns: Elements of Reusable Object-Oriented Software".*
- Integración con cualquier sistema de control de versiones compatible con SCC, como IBM *Rational ClearCase*.
- Admite la integración con otras herramientas de desarrollo.
- Posee un *plugin* para conectarse con *MS Excel.*
- Capacidad de crear definiciones de tipo de documento XML.

Visual Paradigm for UML 8.0

Visual Paradigm es una herramienta CASE que utiliza "UML" como lenguaje de modelado. Soporta el ciclo de vida completo del desarrollo de software: Análisis y Diseño orientados a objetos, Construcción, Pruebas y Despliegue. En esta herramienta, el diseño está centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad. Es una herramienta diseñada para una amplia gama de usuarios, incluyendo el analista del sistema, ingenieros de software, entre otros (60).

Visual Paradigm for UML se caracteriza por (60):

- Soporta todos los diagramas de *SysML* y diagramas de entidad-relación.
- Ofrece amplias características de modelado de casos de uso incluyendo la función completa de UML, diagrama de casos de uso y editor de flujo de eventos.
- Genera *Java*, C#, C++, PHP y lenguaje de definición de datos (DDL) para todas las bases de datos populares.
- Proporciona abundantes tutoriales del lenguaje de modelado, demostraciones interactivas de UML y proyectos UML.
- Se integra con herramientas *Java*, como son: *Eclipse*/IBM, *NetBeans* IDE, entre otras.

Selección de la herramienta CASE a utilizar

Para el modelado se selecciona *Visual Paradigm for UML* por ser una herramienta completa, facilita la construcción del modelo de software a los miembros del equipo en tiempo real. Es multiplataforma, se puede conectar a los diferentes gestores de base de datos, es fácil de usar pues sus componentes se encuentran relacionados facilitando la realización de diagramas. Se utilizará la versión 8.0. Mientras, el *Rational Rose Enterprise* no es multiplataforma y posee una licencia de comercialización de carácter privativo, con un costo adicional al proceso de desarrollo.

1.8.8 Herramientas de prototipado de interfaces de usuario

Son herramientas que por su facilidad de uso permiten desarrollar los esquemas básicos de todo el sitio web y poder realizar ajustes sin tener que rectificar ni una sola línea de código (73).

Pencil 2.0

Herramienta libre y de código abierto para crear prototipos y diagramas de interfaz gráfica de usuario (GUI)

de forma rápida y sencilla. Es una extensión para *Firefox* que actúa como una herramienta de dibujo. Es liberado bajo la versión 2 de la GPL y está disponible para prácticamente todas las plataformas que en *Firefox* se puede ejecutar. Se ejecuta dentro de la ventana de *Firefox* permitiendo graficar con total libertad, diagramar, utilizar botones, así como editar (74).

La herramienta *Pencil* ofrece las siguientes características (74):

- Edición en pantalla de los elementos de texto.
- Multiplataforma.
- Plantillas incorporadas para realizar diagramas y prototipos.
- Instalación de plantillas definidas por el usuario.
- Operaciones estándar de dibujo: alineación, rotación y escala.
- Tramado PNG.
- Exportación a HTML, PNG, documento *Openoffice.org*, documento de *Word* y PDF.

Balsamiq Mockup 2.1.13

Es una de las herramientas de prototipado más populares entre diseñadores. Permite al diseñador organizar los *widgets* pre-construidos con un editor *WYSIWYG* (lo que vez es lo que obtienes) de arrastrar y soltar. La aplicación se ofrece en una versión de escritorio, así como un *plugin* para *Google Drive*, *Confluence* y JIRA. Por otra parte ofrece la misma velocidad y la sensación como cuando se dibuja con un lápiz o la experiencia de dibujar en una pizarra, pero incorporando los medio digitales como arrastrar, soltar y redimensionar (75).

Selección de la herramienta de prototipado de interfaces a utilizar

Por las características de flexibilidad y eficiencia se escoge el *Balsamiq Mockup* como herramienta para el diseño de los prototipos de interfaces de usuario. El equipo de desarrollo de esta investigación ha trabajado en otras ocasiones con la herramienta. Se empleará la versión 2.1.13. Por otro lado, *Pencil* a pesar de poseer ventajas para el diseño de HU tiene la limitante de incompatibilidad con versiones superiores del navegador *Firefox* 18.0.

1.9 Conclusiones del capítulo

En el presente capítulo se definieron las bases para el inicio del desarrollo del sistema. El análisis de las tendencias demostró la importancia de la reutilización de componentes para las líneas de producción de software y el cambio de lenguaje de programación para el desarrollo de productos multimedia, es decir, el cambio de *ActionScript3* por el uso de HTML5, CSS3 y *JavaScript*. Fue seleccionada la metodología XP como rectora del proceso de desarrollo. Además fueron seleccionadas las herramientas y tecnologías requeridas para la implementación del sistema, donde se determinó que se empleará como IDE para el desarrollo *NetBeans* PHP en su versión 7.3, como lenguajes de programación PHP en su versión 5.3, para la implementación de la biblioteca, mientras que HTML5, CSS3 y *JavaScript* para el desarrollo de los componentes multimedia, además se empleará como sistema gestor de base de datos *PostgreSQL* en su versión 9.1, como gestor de contenido *Drupal* en su versión 7.24, se utilizarán los *frameworks jQuery* en su versión 1.7 y *Boostrap* en su versión 2.0; al igual que AJAX como tecnología.

CAPÍTULO 2. ELABORACIÓN DE LA PROPUESTA DE SOLUCIÓN

2.1 Introducción

En el presente capítulo se describe la propuesta del sistema y las características que posee el mismo para la informatización del proceso de desarrollo de aplicaciones con tecnología multimedia. El mismo tiene por objetivo presentar propuesta de solución, así como sus principales funcionalidades. Se especifican los artefactos generados de acuerdo a la metodología seleccionada, dentro de los que sobresalen las Historias de usuario (HU), los planes de iteraciones y de entrega respectivamente.

2.2 Descripción del flujo actual del proceso

El flujo actual del proceso se desarrolla de la siguiente manera: Primero, es asignado un nuevo proyecto a desarrollar. Luego se determina la arquitectura y se busca información en diferentes bibliografías. Se comienza la implementación sin una base de proyectos realizados anteriormente con funcionalidades similares. Se libera luego de las pruebas si no presenta errores, en caso de que presente errores, se notifican los mismos a los desarrolladores, se corrigen y comienza el proceso de revisión nuevamente hasta ser liberado para la entrega al cliente.

2.3 Propuesta de solución

Una vez planteado el objetivo e identificadas las necesidades que posee el centro FORTES de informatizar el proceso de desarrollo de productos con tecnología multimedia, se propone el desarrollo de una aplicación web que permita organizar y centralizar la documentación y los componentes para el proceso de elaboración de aplicaciones con este tipo de tecnología, favoreciendo su reutilización. Dichos productos estarán enfocados a la utilización de tecnologías libres y su visualización en la web, por lo que está encaminado al empleo de tecnologías como HTML, CSS y *JavaScript* principalmente. Incluye además la gestión de documentos relacionados con el proceso productivo de los componentes, como son: manuales, tutoriales y/o ejemplos para el uso de patrones. El sistema ofrece la posibilidad de personalizar los componentes existentes en la aplicación. Se tiene en cuenta que el componente personalizado por el usuario pueda ser una nueva versión de ese componente, este será revisado por los especialistas en el tema y, en caso positivo, liberar la nueva versión del componente para su utilización por parte de los desarrolladores del Centro. La aplicación brinda la posibilidad de compartir los componentes con otros usuarios, lo cual fomenta

el desarrollo en grupo para dar solución a problemas similares; además permite el control de las versiones de componentes y la gestión de una biblioteca personal, donde cada usuario podrá contar con los componentes de su interés. Teniendo en cuenta las diferentes funcionalidades, se establecen cuatro roles principales (administrador, editor, registrado y anónimo) a través de los que se gestiona el acceso al sistema.

2.4 Definición de la audiencia

La audiencia de la biblioteca de componentes multimedia es el público que denota el cumplimiento de los objetivos del sistema, es decir, hacia el grupo de personas al que está dirigida la aplicación. A partir de lo expuesto anteriormente, se deduce que la audiencia son los profesores y estudiantes del centro FORTES. Es importante señalar que aunque se haya definido la audiencia para las personas del centro FORTES, no quedan excluidas todas aquellas que se sientan motivadas por el desarrollo de aplicaciones con estas tecnologías.

2.5 Resultados esperados

Los resultados que se esperan alcanzar con el desarrollo del sistema son:

- Facilitar el proceso productivo de productos con tecnología multimedia del centro FORTES.
- Centralizar la documentación y componentes existentes sobre tecnología multimedia.
- No repetir esfuerzos en la producción de un componente.
- Disminuir el tiempo de desarrollo de productos con tecnología multimedia mediante la reutilización de los componentes asociados a la misma.
- Dar seguimiento, control y organización a los componentes que se producen en el centro.
- Consolidar el uso de componentes reutilizables.
- Establecer una comunidad activa para el perfeccionamiento y desarrollo de componentes reutilizables para productos con tecnología multimedia.

2.6 Usuarios relacionados con el sistema

Como parte del proceso de interacción con el sistema se han definido cuatro roles como se especifican a continuación:

Tabla 1 Roles del sistema

Usuario	Descripción
Administrador	Persona autorizada para la gestión del sistema, encargado de adicionar, actualizar y eliminar toda la información de la biblioteca. Cuenta con los permisos requeridos para llevar a cabo todas las funciones administrativas del sistema, con el fin de darle soporte y mantenimiento a la aplicación, así como velar por la seguridad. Este usuario hereda los permisos que tiene el usuario registrado.
Editor	Es la persona con privilegios que le permiten adicionar, actualizar, eliminar o aprobar la publicación de componentes y versiones de los mismos en la biblioteca. Este usuario hereda los permisos que tiene el usuario registrado.
Registrado	Usuario que se encuentra autenticado en el sistema, tendrá acceso a los diferentes módulos y servicios que brinda la aplicación y tiene en cuenta el rol que cumple dentro de la misma. Cuenta con privilegios como insertar y personalizar componentes, crear versiones y añadir componentes a la biblioteca personal.
Anónimo	Es aquella persona que accede a la aplicación sin haber creado antes una cuenta de usuario en el sistema, está limitado a buscar visualizar y descargar componentes y documentación.

2.7 Estructura definida para un componente

A la hora de almacenar un componente en la solución propuesta, es necesario que cumpla con una estructura previamente definida para su correcta organización y visualización dentro del sistema, así como para su reutilización por parte de los desarrolladores. El componente debe tener una estructura como se muestra en la figura siguiente.

Fig.3 Estructura del componente

Descripción de los archivos contenidos en el componente:

- LEEME.txt es el archivo donde se describe cómo funciona el componente, las dependencias que tiene el mismo, las librerías utilizadas y cualquier otro elemento que aclare cómo se puede reutilizar.
- El archivo HTML es el que visualizará el componente. Debe tener el nombre *index* por defecto para su correcta visualización dentro del sistema (ej. *index.html*). El sistema carga de forma automática el archivo con ese nombre, para la vista previa del componente.
- Los archivos CSS deben estar dentro de la carpeta css, independientemente del nombre que estos tengan.
- Los archivos JS deben estar dentro de su respectiva carpeta js, sin importar su nombre.
- Las medias (imágenes, videos) u otros recursos utilizados en el componente deben estar agrupados en la carpeta recursos.
- En caso de utilizar un archivo XML, este se encontrará al mismo nivel del archivo index.html.
- Las carpetas css, js y recursos son necesarias si el componente contiene este tipo de elementos.
- El componente debe estar comprimido en formato .zip con la estructura antes descrita, además a la hora de comprimir los elementos del mismo, estos no deben estar contenidos dentro de una carpeta.

La figura siguiente muestra un ejemplo de estructura incorrecta al comprimir el componente.

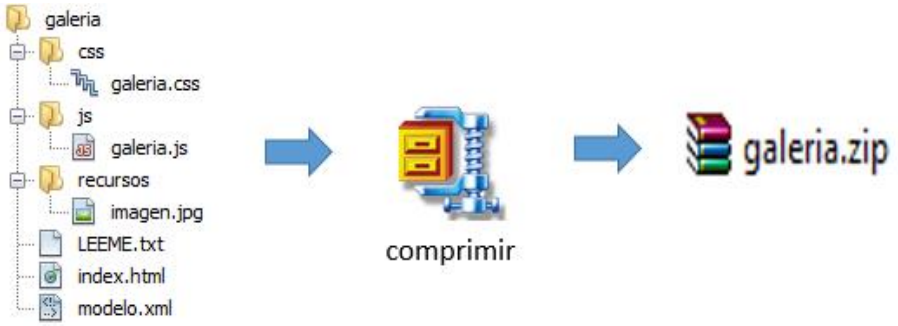


Fig.4 Estructura incorrecta de un componente

Es imprescindible que el componente contenga el archivo LEEME.txt, para que otros usuarios conozcan el funcionamiento del mismo y puedan reutilizarlo según sus necesidades.

2.8 Fase de exploración

La primera fase de la metodología de desarrollo del software XP es la exploración, donde se define el alcance del proyecto (76), además son generadas las HU por parte del cliente, donde se recoge brevemente las características o funcionalidades del sistema. Al mismo tiempo los desarrolladores se familiarizan con las herramientas y las tecnologías para trabajar en el desarrollo del sistema.

2.8.1 Aspectos funcionales del sistema

Para darle cumplimiento a los objetivos trazados, se plantean los aspectos funcionales con los que deberá cumplir el sistema.

1. **Gestionar usuarios**.
    1.1 Crear usuario.
    1.2 Modificar usuario.
    1.3 Eliminar usuario.
2. **Autenticar usuario.**
3. **Gestionar componentes.**
    3.1 Adicionar componente.
    3.2 Editar componente.
    3.3 Eliminar componente.
4. **Mostrar componente.**

5. **Filtrar componentes por categoría.**
6. **Publicar componente.**
7. **Buscar componentes y documentos.**
8. **Descargar componente.**
9. **Personalizar componente.**
10. **Mostrar ficha técnica del componente.**
11. **Mostrar últimos componentes.**
12. **Gestionar documentación.**
    - 12.1 Adicionar documentación.
    - 12.2 Editar documentación.
    - 12.3 Eliminar documentación.
13. **Mostrar documentación.**
14. **Filtrar documentación por tipo.**
15. **Descargar documentación.**
16. **Enviar notificaciones por correo.**
17. **Gestionar versión de componente.**
    - 17.1 Crear versión de componente.
    - 17.2 Eliminar versión de componente.
18. **Subir versión de componente.**
19. **Compartir componente.**
20. **Gestionar biblioteca personal.**
    - 20.1 Añadir componente a la biblioteca personal.
    - 20.2 Eliminar componente de la biblioteca personal.
21. **Mostrar biblioteca personal.**

2.8.2 Aspectos no funcionales del sistema

Los siguientes aspectos no funcionales identificados, son propiedades o cualidades que debe tener el sistema:

Interfaz.

El diseño de la interfaz visual debe ser minimalista, sin uso excesivo de animaciones e imágenes de alta calidad o tamaño que perjudiquen el tiempo de respuesta de las peticiones que se realicen a la aplicación. La interfaz debe ser diseñada de forma tal que permita una sencilla navegación y el fácil entendimiento de las funcionalidades que brinda la biblioteca, además de poseer colores refrescantes para una mejor interacción entre el usuario y la aplicación.

Usabilidad.

La aplicación debe ser diseñada de forma que los usuarios que hagan uso de la misma obtengan los conocimientos necesarios en el menor tiempo posible, para evitar el entrenamiento de los usuarios para trabajar con el sistema y facilitar una mejor explotación de sus funcionalidades.

Software.

Para instalar el sistema se debe contar:

Máquina servidor:

Sistema Operativo: *Windows* o *Linux*.

Servidor web: Apache 2.

Lenguaje PHP versión 5 o superior.

Gestor de base de datos: *PostgreSQL* 9.1.

Máquina cliente:

Sistema operativo: *Windows* XP o superior, cualquier distribución de Linux.

Navegador web: *Firefox* versión 4.0 o superior, *Google Chrome* 27.0 o superior*, Internet Explorer* 7.0 o superior.

Hardware.

La máquina servidor deberá contar con un microprocesador *Dual Core*, 2GB de RAM, un mínimo de 3 GB de disco duro, se debe considerar la posible expansión del volumen de datos.

Rendimiento.

Rapidez y eficiencia tanto en los tiempos de respuesta como en la velocidad de procesamiento. Garantizar

velocidad estable de navegación a nivel de aplicación. Proporcionar tiempos de respuesta mínimos que no excedan de los 3 segundos, para los procesos del sistema.

Seguridad.

Acceder a la información autorizada de acuerdo al rol de cada usuario en el sistema. Evitar que la información en el sistema sea modificada por usuarios anónimos.

Disponibilidad.

Cada usuario con facultades debe tener acceso pleno al uso de la aplicación 7 días a la semana, 365 días al año.

Portabilidad.

El sistema debe ser capaz de ejecutarse en diferentes entornos sin sufrir cambios en su estructura.

Adaptabilidad.

El sistema debe ser adaptable a la resolución de la pantalla de los diferentes dispositivos que utilice el cliente.

Capacidad.

Considerar características técnicas mínimas para la ejecución en clientes. El sistema debe permitir aproximadamente 120 conexiones de manera simultánea.

2.8.3 Historias de usuario (HU)

Las HU contemplan de forma sencilla lo que el cliente desea en la aplicación. Estas plantillas tienen la misma finalidad que los casos de uso, aunque es importante destacar que existen diferencias teniendo en cuenta la forma de representar ambos artefactos. Las HU son usadas para estimar tiempos de desarrollo de la parte de la aplicación que describen. También se utilizan en la fase de pruebas, para verificar si el sistema cumple con lo que especifica la HU. A continuación se muestran algunas de las HU propuestas (Ver todas las HU en ANEXO 2):

Tabla 2 HU- Crear cuenta de usuario

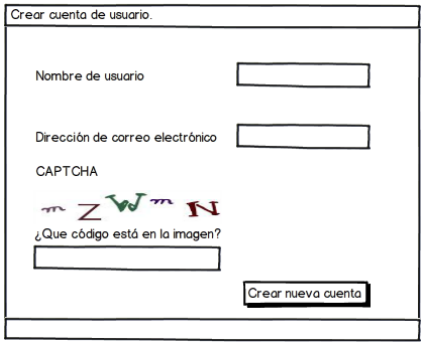
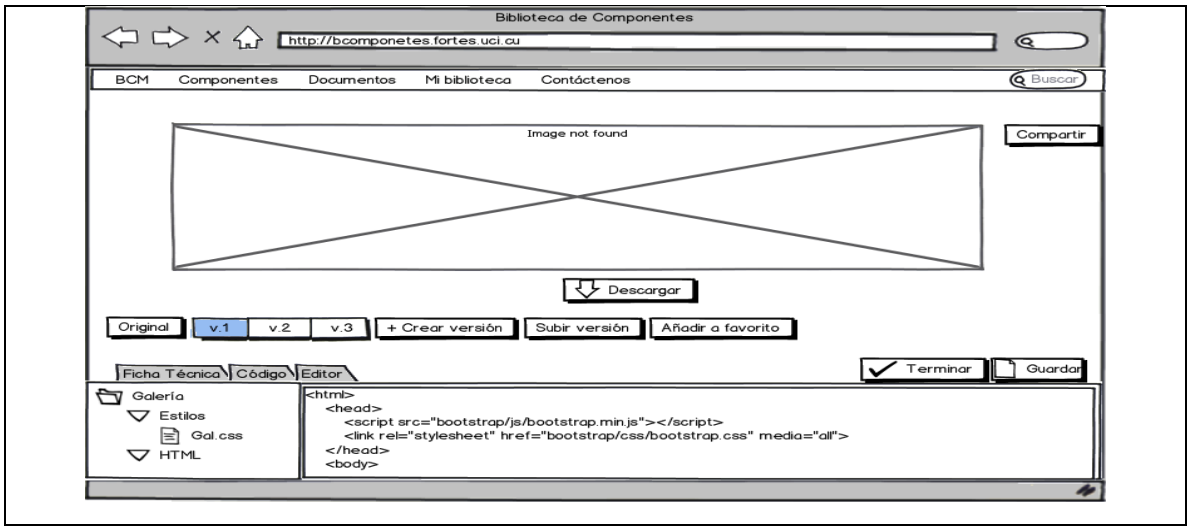
Historia de usuario	
No.: 1	**Nombre:** Crear cuenta de usuario
Usuario: Anónimo, administrador	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3
Riesgo en desarrollo: Alta	**Iteración:** 1
Descripción: Permite al usuario crearse un cuenta en el sistema. El usuario debe llenar los datos nombre de usuario y dirección de correo electrónico.	
Prototipo de interfaz: 	

Tabla 3 HU- Personalizar componente

Historia de usuario	
No.: 13	**Nombre:** Personalizar componente
Usuario: Registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 2.6
Riesgo en desarrollo: Alta	**Iteración:** 3
Descripción: El usuario registrado puede personalizar el componente a través del código.	
Prototipo de interfaz:	



2.9 Fase de Planificación

En esta fase el usuario es el encargado de definir la prioridad que tiene cada una de las HU, a partir de esta, el equipo de desarrollo hace una estimación de esfuerzo para el cumplimiento de las mismas y se llega al acuerdo de cuáles de las HU estarán incluidas en la primera entrega (37).

2.9.1 Estimación de esfuerzos por HU

Para el desarrollo del sistema propuesto, y basado en el análisis de cada uno de los aspectos funcionales del mismo, se realizó una estimación de esfuerzo para la implementación de cada una de las HU, posibilitando que se posea una medida real del progreso alcanzado por el sistema. Los resultados de la estimación se muestran a continuación:

Tabla 4 Plan de estimación de esfuerzos por HU

No.HU	Historia de usuario	Puntos de estimación
1	Crear cuenta de usuario	0.3
2	Modificar cuenta de usuario	0.3
3	Eliminar cuenta de usuario	0.3
4	Autenticar usuario	0.3
5	Agregar componente	0.3
6	Editar componente	0.3

7	Eliminar componente	0.3
8	Mostrar componente	0.3
9	Filtrar componente por categoría	0.3
10	Publicar componente	0.3
11	Buscar componentes y documentación	0.3
12	Descargar componente	0.3
13	Personalizar componente	2.6
14	Mostrar ficha técnica del componente	0.4
15	Mostrar últimos componentes	0.3
16	Agregar documento	0.3
17	Modificar documento	0.3
18	Eliminar documento	0.3
19	Mostrar documentos	0.3
20	Filtrar documentos por tipo	0.3
21	Descargar documentación	0.3
22	Enviar notificaciones por correo	0.3
23	Crear versión de componente	0.4
24	Eliminar versión de componente	0.4
25	Subir versión de componente	0.4
26	Compartir componente	0.4
27	Añadir componente a la biblioteca personal	0.4
28	Eliminar componente de la biblioteca personal	0.4
29	Mostrar biblioteca personal	0.4

2.9.2 Plan de iteraciones

Una vez que se han descrito las HU y se ha estimado el esfuerzo por los desarrolladores para la realización de las mismas, se procede a realizar la planificación de la etapa de implementación del sistema. En este plan se agrupan las HU, donde se especifica en la iteración que será desarrollada cada una. Basado en

este análisis se ha determinado implementar la aplicación en cuatro iteraciones, las cuales se describen a continuación:

Iteración 1: En la primera iteración se persigue el objetivo de darle cumplimiento a la implementación de las HU No.1, 2, 3, 4, 5, 6, 7, 8, 9 y 11; siendo estos los aspectos funcionales relacionados con el proceso de gestión (adicionar, modificar y eliminar) de los usuarios en la aplicación y los componentes que en ella se encuentran, se abarca también las funciones de autenticación de usuarios así como las de mostrar componentes, filtrarlos por categoría y la búsqueda tanto de componentes como de la documentación presente en el sistema.

Iteración 2: Esta iteración está encaminada a dar cumplimiento a las HU No.10, 12, 15, 16, 17, 18, 19, 20, 21 y 22. En esta iteración se engloba el proceso de implementación de varias de las funciones necesarias para el trabajo con los componentes y la gestión de documentos; dentro de las que sobresalen la de búsqueda y descarga de componentes y documentación, así como las de agregar, modificar, eliminar, mostrar, filtrar por tipo y descargar documentos. Además se desarrollan las funcionalidades de envío de notificaciones por correo y la de mostrar los últimos componentes añadidos al sistema.

Iteración 3: En la tercera iteración se da paso al cumplimiento de las HU No.13 y 14. Esta iteración está relacionada con la implementación de las funcionalidades necesarias para la personalización de los componentes y la de mostrar la ficha técnica de cada uno de ellos.

Iteración 4: En la cuarta iteración se desarrolla lo concerniente a las HU No.23, 24, 25, 26, 27, 28 y 29. Esta iteración está marcada por la implementación de las funcionalidades que se encargarán de la gestión de las versiones de los componentes y de la biblioteca personal; entre ellas sobresalen las de crear, eliminar y subir versiones de los componentes, además se encuentran las que permiten compartir un componente con otros usuarios, añadir y eliminar elementos a la biblioteca personal y mostrar esta última.

2.9.3 Plan de duración de las iteraciones

Como parte del ciclo de vida de un proyecto al utilizar la metodología XP, es necesario crear un plan de duración de cada una de las iteraciones. Este plan tiene como finalidad mostrar la duración estimada de cada una de las iteraciones. A continuación se presenta la tabla con las estimaciones determinadas para cada una de las iteraciones.

Tabla 5 Plan de duración de las iteraciones

No. de Iteración	Historia de usuario por iteración	Tiempo de duración (semanas)
1	HU1, HU2, HU3, HU4, HU5, HU6, HU7, HU8, HU9, HU11	3
2	HU10, HU12, HU15, HU16, HU17, HU18, HU19, HU20, HU21, HU22	3
3	HU13, HU14	3
4	HU23, HU24, HU25, HU26, HU27, HU28, HU29	2.8
Total	29	11.8

El plan de duración de las iteraciones está basado en los puntos de esfuerzo que determina la metodología XP en la cual un punto de esfuerzo equivale a 40 horas de trabajo semanal (77). Un día de trabajo posee una estimación asociada de 0.2.

2.9.4 Plan de entregas

El plan de entregas está orientado a especificar el número de entregables que tendrá el sistema antes de su despliegue y las iteraciones comprendidas en cada uno de estos. El cliente es el encargado de decidir cuáles de las HU deben estar implicadas en cada entregable de acuerdo a sus prioridades o a las prioridades del negocio, para de esta forma aumentar el valor de las mismas y justificar las siguientes.

Tabla 6 Plan de entregas

Historias de usuario	Del 17 de febrero al 7 de marzo.	Del 10 al 28 de marzo.	Del 31 de marzo al 25 de abril.	Del 28 de abril al 16 de mayo.
Crear cuenta de usuario.	V 1.0	Finalizado	Finalizado	Finalizado
Modificar cuenta de usuario.	V 1.0	Finalizado	Finalizado	Finalizado
Eliminar cuenta de usuario.	V 1.0	Finalizado	Finalizado	Finalizado
Autenticar usuario.	V 1.0	Finalizado	Finalizado	Finalizado
Agregar componente.	V 1.0	Finalizado	Finalizado	Finalizado

Editar componente.	V 1.0	Finalizado	Finalizado	Finalizado
Eliminar componente.	V 1.0	Finalizado	Finalizado	Finalizado
Mostrar componente.	V 1.0	Finalizado	Finalizado	Finalizado
Filtrar componente por categoría.	V 1.0	Finalizado	Finalizado	Finalizado
Buscar componentes y documentación.	V 1.0	Finalizado	Finalizado	Finalizado
Publicar componente.		V 1.0	Finalizado	Finalizado
Descargar componentes.		V 1.0	Finalizado	Finalizado
Mostrar últimos componentes.		V 1.0	Finalizado	Finalizado
Agregar documento.		V 1.0	Finalizado	Finalizado
Modificar documento.		V 1.0	Finalizado	Finalizado
Eliminar documento.		V 1.0	Finalizado	Finalizado
Mostrar documentos.		V 1.0	Finalizado	Finalizado
Descargar documentos.		V 1.0	Finalizado	Finalizado
Filtrar documentos por tipo.		V 1.0	Finalizado	Finalizado
Enviar notificaciones por correo.		V 1.0	Finalizado	Finalizado
Personalizar componentes.			V 1.0	Finalizado
Mostrar ficha técnica del componente.			V 1.0	Finalizado
Crear versión de componente.				V 1.0
Eliminar versión de componente.				V 1.0
Subir versión de componente.				V 1.0
Compartir componentes.				V 1.0
Añadir componente a la biblioteca personal.				V 1.0
Eliminar componente de la biblioteca personal.				V 1.0
Mostrar biblioteca personal.				V 1.0

2.10 Conclusiones del capítulo

En el presente capítulo se propone la aplicación a desarrollar. La misma permitirá la gestión de los componentes de software con tecnología multimedia desarrollados en el departamento, cumpliendo de esta manera el objetivo general planteado. Se definió la audiencia del sistema, así como los roles que desempeñará cada usuario, los cuales son: administrador, editor, registrado y anónimo. En la fase de exploración se escribieron veintinueve HU de aspectos funcionales y diez de aspectos no funcionales. Las HU fueron agrupadas en cuatro iteraciones donde las de mayor prioridad para el cliente y el negocio fueron recogidas en las primeras iteraciones. Cada una de las iteraciones posee un tiempo estimado promedio de tres semanas para su desarrollo. El plan de entregas arrojó que el tiempo de desarrollo del sistema será aproximadamente de tres meses.

CAPÍTULO 3. PRODUCCIÓN Y PRUEBAS

3.1 Introducción

Antes de comenzar el desarrollo de un sistema, es necesario que dicho proceso sea guiado por un conjunto de estándares y patrones con el objetivo de lograr una línea base, para que todos los desarrolladores que trabajen con el sistema puedan entender el funcionamiento del mismo. En el presente capítulo se detallan las fases de Producción y Pruebas según se establece en la metodología XP. Además se refleja el diseño del sistema con la metáfora, las tarjetas CRC, así como los estándares de codificación para hacer el código legible y entendible a los miembros del equipo y futuros desarrolladores. Por último se generan las tareas de investigación y se realizan las pruebas al software para verificar la calidad del mismo.

3.2 Diseño del sistema

3.2.1 Metáfora

La metodología XP se basa en 12 principios básicos, uno de ellos es la metáfora. Según (77) la metáfora del sistema es algo que todos los clientes y programadores entienden sobre cómo funciona el sistema. Las metáforas ayudan con la abstracción y modelado del sistema.

Se establece como metáfora para la solución, la siguiente idea: “es una aplicación web donde los usuarios pueden encontrar componentes para reutilizarlos y consultar bibliografía referente a los mismos”.

3.2.2 Tarjetas Clase Responsabilidad Colaboración (CRC)

La metodología XP a diferencia de otras metodologías no dispone de un diagrama de clases utilizando notación UML. En su lugar XP utiliza las tarjetas CRC, estas modelan las relaciones entre las clases. *Drupal* en su funcionamiento o lógica interna no hace uso de conceptos como clases, sin embargo, utiliza otros conceptos de la POO como son: la abstracción, el polimorfismo, encapsulación y herencia. Se hace necesario realizar modificaciones a las tarjetas CRC según las características de *Drupal*. Se define la estructura de las tarjetas CRC de la siguiente manera (Ver todas las tarjetas CRC en ANEXO 3):

Clases: Módulo que ejecuta la tarjeta CRC.
Responsabilidades: Se define como el conjunto de funcionalidades que realiza el módulo.

Colaboraciones: Son los módulos de los que depende la clase para que funcionen las responsabilidades del módulo.

Tabla 7 Tarjeta CRC Control de versiones y personalización de componentes

Módulo *bibliocom*	
Responsabilidades	**Colaboraciones**
Personalizar componentes	*node*
Mostrar datos del componente	*user*
Gestionar versiones	*Drupal API*
Subir versión	*tar*
Control de versiones	*zip*
Gestionar biblioteca personal	*xcompress*
Mostrar biblioteca personal	*features*
Descargar componentes	

Tabla 8 Tarjeta CRC Compartir componente

Módulo m_compartir	
Responsabilidades	**Colaboraciones**
Compartir componentes	*node* *user* *m_notificacion*

3.2.3 Prototipo de interfaz

La interfaz principal del sistema en cumplimiento de la HU de apariencia se diseña de la siguiente forma:

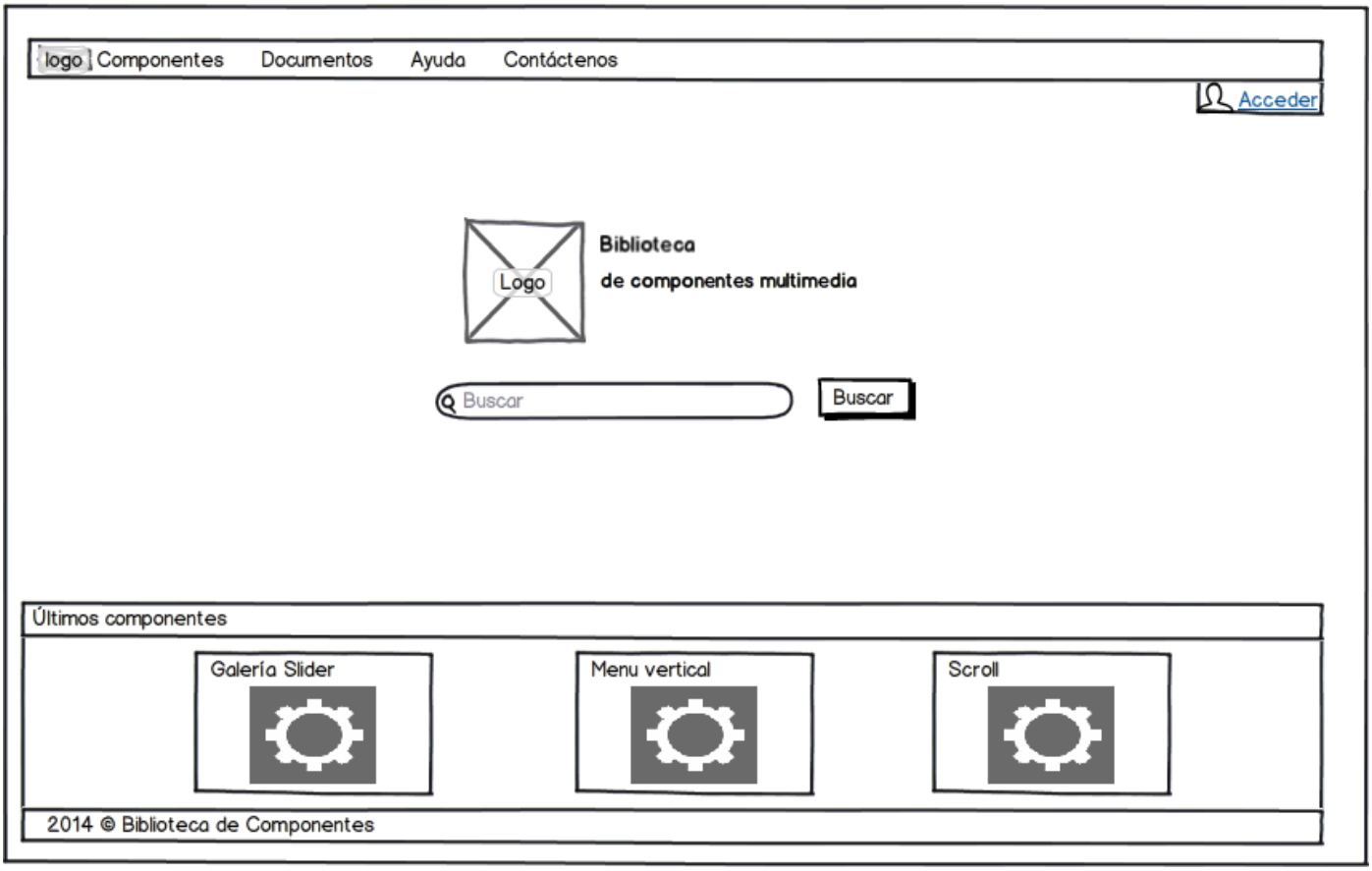


Fig.5 Prototipo de interfaz de usuario de la página principal de la biblioteca de componentes multimedia

3.2.4 Estándares de codificación

Los estándares de codificación son patrones para la generación de código. *Drupal* define sus propios estándares de programación o codificación para los desarrolladores de módulos. A fin de desarrollar los módulos *bibliocom*, m_compartir y m_notificacion se toman como estándares de codificación los siguientes estándares ya definidos para *Drupal* y publicados en el libro *Experto en Drupal. Creación y gestión de portales web con Drupal 7*.

Etiquetas de apertura y cierre de PHP

Se debe escribir las etiquetas <?php y ?>, y en ningún caso la versión corta <? y ?>. En los archivos .module y .inc., se omite la etiqueta de cierre de PHP (?>) al final de los archivos. Por tanto, la etiqueta de cierre final

del archivo (?>) es opcional en *Drupal*. En los archivos .tpl.php o en cada fragmento de PHP debe llevar sus correspondientes etiquetas de apertura y cierre para diferenciarlo del código HTML (43).

Operadores

Los operadores binarios, que se utilizan entre dos valores deben separarse de estos valores, a ambos lados del operador, por un espacio. Por ejemplo, $numero = 3, en lugar de $numero=3. Esto se aplica a operadores como +, -, *, /, =, ==, !=, >, <, (concatenación de cadenas), .=, +=, -=, etc.
Los operadores unarios como ++, -- no deben tener separación. Por ejemplo, $numero++ (43).

Uso de comillas

Se pueden usar tanto las comillas simples ('cadena') como las comillas dobles ("cadena") para delimitar las cadenas de caracteres. Las comillas dobles son necesarias si se desean incluir variables dentro de las cadenas de texto. Por ejemplo, "<h1>*$title*</h1>". También se recomienda el uso de comillas dobles cuando el texto puede incluir alguna comilla simple (43).

Uso de punto y coma (;)

Aunque PHP permite escribir líneas de código individuales sin el terminador de línea (;), como por ejemplo <?*php print $title*?>. En *Drupal* es siempre obligatorio: <?*php print $title*;?> (43).

- Correcto: <?php print $title; ?>
- Incorrecto: <?php print $title ?>

Fig.6 Uso del punto y coma en Drupal (43)

Estructuras de control

Con respecto a las estructuras de control, hay que tener en cuenta las siguientes normas (43):
- Debe haber un espacio entre el comando que define la estructura (*if*, *while*, *for*, etc.) y el paréntesis de apertura. Esto es así para no confundir las estructuras de control con la nomenclatura de las funciones.
- La llave de apertura {, se situará en la misma línea que la definición de la estructura, separada por un espacio.
- Se recomienda usar siempre las llaves {} aun en los casos en que no sea obligatorio su uso (una sola "línea" de código dentro de la estructura de control).
- Las estructuras *else* y *elseif* se escribirán en la línea siguiente al cierre de la sentencia anterior.

```
//sentencia if
if (condicion1 || condicion2) {
  accion1;
}
elseif (condicion3 && condicion4) {
  accion2;
}
else {
  acciones_por_defecto;
}

//sentencia switch
switch (condicion) {
  case 1:
    accion1;
    break;

  case 2:
    accion2;
    break;

  default:
    acciones_por_defecto;
}

//sentencia for
for ($i = 0; $i < 5; $i++) {
  acciones;
}
```

Fig.7 Estructuras de control de Drupal (43)

Funciones

Los nombres de las funciones deben estar escritos en minúsculas y las palabras separadas por guion bajo. Además, se debe incluir siempre como prefijo el nombre del módulo, tema, etc., para evitar así duplicidad de funciones (43).

En su declaración, después del nombre de la función, el paréntesis de inicio de **los argumentos debe ir sin espacio**. Cada argumento debe ir separado por un espacio, después de la coma del argumento anterior (43). En la llamada a la función se aplican las mismas reglas anteriores con respecto a los parámetros, como se muestra en el siguiente ejemplo:

```
$field = field_info_instance('node', 'taxonomy_forums', $node->type);
```

Fig.8 Llamadas a una función en Drupal (43)

Como excepción, es posible usar más de un espacio antes de una asignación (=) para mejorar la presentación, cuando se realicen varias asignaciones en bloque:

```
$numero1      = foo($a, $type);
$primer_valor = foo2($b);
$i            = foo3();
```

Fig.9 Asignaciones (43)

Arrays

Los valores dentro de un array (o matriz) se deben separar por un espacio (después de la coma que los separa). El operador => debe separarse por un espacio a ambos lados (43).

Cuando la línea de declaración del array supera los 80 caracteres, cada elemento se debe escribir en una única línea, indentándolo una vez (2 espacios). En este último caso, la coma de separación del último elemento también se escribirá, aunque no existan más elementos. De esta forma se evitan errores al añadir nuevos elementos al vector (43).

```
$vector1 = array(1, 2, 'clave' => 'valor');

$vector2 = array(
  'forum' => 'foro1',
  'template' => 'forums',
  'arguments' => array('tid' => NULL, 'topics' => NULL),
  'size' => 128,
);
```

Fig.10 Declaración de arreglos con Drupal (43)

Constantes

Los nombres de las constantes deben escribirse en mayúsculas, con guiones bajos para separar palabras. Al igual que ocurre con las funciones, los nombres de las constantes deben tener como prefijo el nombre del módulo (o tema) en el que se utilizan, para evitar errores de duplicidad de constantes. Este prefijo también se escribirá en mayúsculas (43).

Nombres de módulos

Como norma general, el nombre de un módulo nunca debería incluir guiones bajos, aunque se componga de varias palabras. De esta forma será más fácil identificar el módulo al que pertenece una función, ya que

el prefijo o nombre del módulo será todo aquello que esté antes del primer guion bajo. Por ejemplo, es aconsejable utilizar mimodulo en lugar de mi_modulo (43). Esta regla no es obligatoria, y es muy común encontrar, entre los módulos contribuidos, nombres conteniendo guiones bajos.

3.3 Fase de producción

En esta fase, se realiza un análisis de cada una de las HU en conjunto con el plan de iteraciones y se realizan modificaciones en caso de ser necesario. Luego las HU se descomponen en tareas de ingeniería y son asignadas a un grupo de desarrollo o a una persona como responsable de su implementación (76). Dichas tareas, son asignadas a los programadores por lo que son escritas en lenguaje técnico y no en un lenguaje entendible por el cliente.

La implementación del sistema parte de la estructura base de *Drupal* donde se destacan cinco conceptos claves para comprender el funcionamiento del sistema implementado. A continuación se describen los conceptos básicos que permiten la flexibilidad y organización de *Drupal*:

1- Módulos

Los módulos permiten añadir funcionalidades adicionales al núcleo de *Drupal*. Los módulos se estructuran con tres archivos importantes, el .info que guarda toda información referente al módulo como son las dependencias, la versión para la cual funcionará, el nombre y una descripción. El .install es el que le permite al módulo cuando se instala ejecutar acciones importantes para su funcionamiento. Un ejemplo la creación de las tablas en la base de datos. Por último el .module que es donde se implementan las funcionalidades que serán añadidas al sistema.

2- Temas

Los temas definen un diseño de interfaz específico para el sistema. Partiendo de este término es que *Drupal* puede separar los contenidos del diseño, posibilitando transformar el aspecto del sistema con solo cambiar o modificar el tema.

3- Nodos

Los nodos son la estructura que permite gestionar los contenidos en *Drupal*, es decir, almacenar la información que se maneja en el sistema.

4- Bloques y menús

Los bloques y menús se encargan de la configuración y la visualización de los contenidos. Definen dónde y cómo se presentaran los contenidos en el sistema.

5- Roles de usuario

La seguridad y control de los usuarios se garantiza a través de la capa de permisos de usuarios. Estos permisos son otorgados a través de los roles que le son asignados a un usuario. Los roles son implementados para dar acceso a los contenidos en el sistema.

3.3.1 Arquitectura de software

El modelo-vista-controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Como se puede observar en la figura 11, la utilización que *Drupal* hace del patrón arquitectónico MVC no existe interacción entre la vista y el modelo, esto se debe a que dichas interacciones siempre se realizan a través de la lógica de negocio, el controlador.

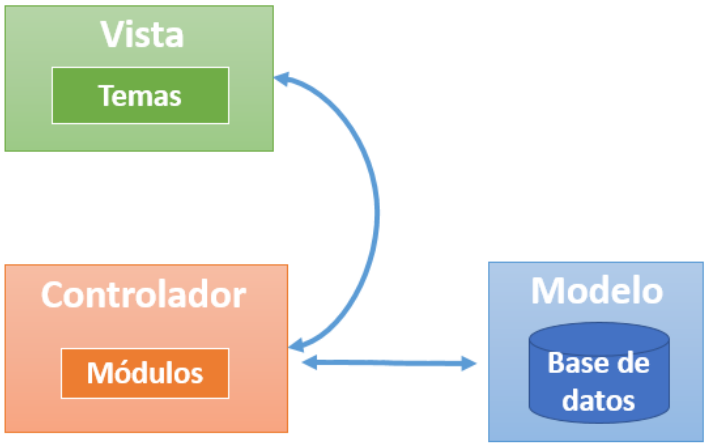


Fig.11 Arquitectura empleada por el CMS Drupal

3.3.2 Patrones de diseño

Los patrones de diseño son soluciones acertadas a un problema general, establecen pautas para definir estructuras de diseño en el desarrollo de un software y las soluciones de estos patrones pueden ser

reutilizadas en muchas situaciones. Los patrones de diseño están más enfocados a los diseños orientados a objetos pero según (78) dichos patrones a menudo tienen en cuenta características de los objetos tales como la herencia y el polimorfismo que proporcionan generalidad.

Drupal a pesar de no hacer exhaustivamente uso de las características de la POO, deja ver en su código base y en la API la utilización de estas características y el diseño orientado a objetos.

Al desarrollarse el sistema utilizando *Drupal* como base, se adoptan los patrones de diseño propiamente usados por el CMS, específicamente los patrones GOF (*Gang of Four*). A continuación se detalla cómo se emplean estos patrones en la solución:

Singleton (Instancia única)

El objetivo de este patrón es asegurarse de que de una clase solo existe una instancia y que esta es accesible, o mejor dicho, ofrecer un punto de acceso a ella (79).

Analizando los módulos y los temas de *Drupal* como objetos, los cuales pueden ser pensados como una clase con una instancia única, permite identificar el patrón *singleton*. Estos objetos no encapsulan datos, lo que separa a un módulo de otro es el conjunto de funciones que contiene, garantizando así la existencia de una única instancia para un módulo.

Decorator (Decorador)

Este patrón permite añadir funcionalidades o responsabilidades a un objeto de forma dinámica (80).

Ello se refleja en el uso de los ganchos (*hooks*) utilizados por los nodos, ejemplo de estos ganchos son *hook_node_load*(), *hook_node_view*(), *node_invoke*(), permitiendo a otros módulos acciones arbitrarias para extender su comportamiento, ampliando la variedad de comportamientos para ser añadidos a los nodos.

Bridge (Puente)

El objetivo de este patrón es la separación de la abstracción de la implementación, de tal forma que ambas puedan ser modificadas independientemente sin afectar la una a la otra (80).

En la capa de abstracción de datos de *Drupal* se implementa de manera similar al patrón puente. No se realizan llamadas directas a la base de datos, sino que se hacen a través de funciones genéricas (*db_select*, *db_update*, *db_delete*, *db_insert*) definidas por la capa de abstracción de datos, que realiza la función de puente. La utilización de este diseño permite que los módulos puedan utilizarse independientemente del gestor de base de datos utilizado.

Chain of Responsibility (Cadena de responsabilidades)

Proporciona a más de un objetos la capacidad de atender una petición, para así evitar el acoplamiento con el objeto que hace la petición. Se forma entre los objetos una cadena, en la cual cada objeto o satisface la petición o la pasa a la siguiente (80).

La cadena de responsabilidades se observa en los sistemas de menú de *Drupal*. El sistema, cuando un usuario visita una funcionalidad, debe determinar partiendo de la url, cuál módulo es el encargado de su gestión y presentación. Luego se sigue un flujo en cadena que comprueba en todos los módulos activos, a quién pertenece cada solicitud de la página, lográndose esto a través de la implementación del *hook_menu*(). Una vez encontrado el módulo, este le facilitará al sistema una función de retorno o *callback*, que es la encargada de generar el contenido de la página.

Command (Comando)

Este patrón permite solicitar una operación a un objeto sin conocer realmente el contenido de esta operación, ni el receptor real de la misma. Para ello se encapsula la petición como un objeto (81).

Este patrón de diseño es utilizado en *Drupal* para permitir que los módulos no tengan que definir cada gancho sino únicamente los que van a utilizar. Se pone en práctica la implementación de *callback*, en el *hook_menu*() cuando se crea un *ítem* de tipo *MENU_CALLBACK*.

3.3.3 Módulos desarrollados para la biblioteca

En la implementación del sistema que es objeto de esta investigación se desarrollaron tres módulos. El primero llamado *bibliocom* permite la gestión de las versiones, la personalización de componentes y gestión de la biblioteca personal de los usuarios. En el módulo se implementaron los tres archivos anteriormente descritos (.info, .install y .module) y se desarrollaron cuatro temas para presentar la información con diferentes formatos. El tema *node—bibliocom.tpl.php* visualiza las funcionalidades de personalización de

componentes, mientras que el *bpersonal.tpl.php* muestra la biblioteca personal de cada usuario. El *documento.tpl.php* renderiza el nodo de tipo documentos del sistema, modificando la forma en que el tema base presenta los nodos en el sistema. La estructura del módulo *bibliocom* es la siguiente:

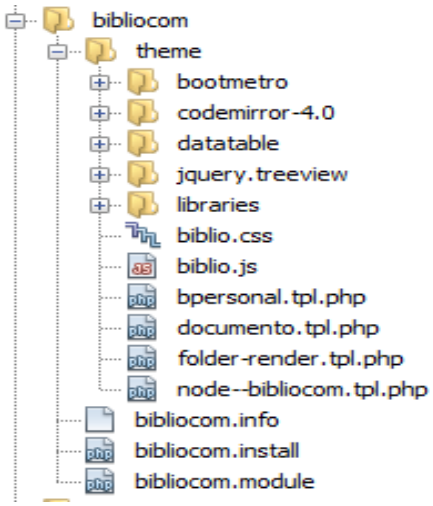


Fig.12 Estructura del módulo bibliocom

El módulo m_compartir define la estructura del nodo para compartir los componentes a otros usuarios. Este módulo posee una dependencia del m_notificacion que es para enviar correos con formato HTML. Los módulos implementados se encuentran en la dirección del proyecto */sites/all/modules/*. *Drupal* como buena práctica propone separar los módulos del núcleo de los módulos contribuidos[1].

Al instalar el módulo *bibliocom* se crean dos tablas en la base de datos, la tabla c_versiones y la tabla c_bpersonal. La primera tabla almacena los datos referentes a las versiones que se crean en el sistema. En tanto la segunda, guarda los datos de los componentes favoritos de cada usuario. El siguiente diagrama muestra las relaciones entre las tablas principales del sistema y las añadidas por el módulo *bibliocom*.

[1] Son módulos que no contiene el núcleo de Drupal y que son desarrollados por otras personas.

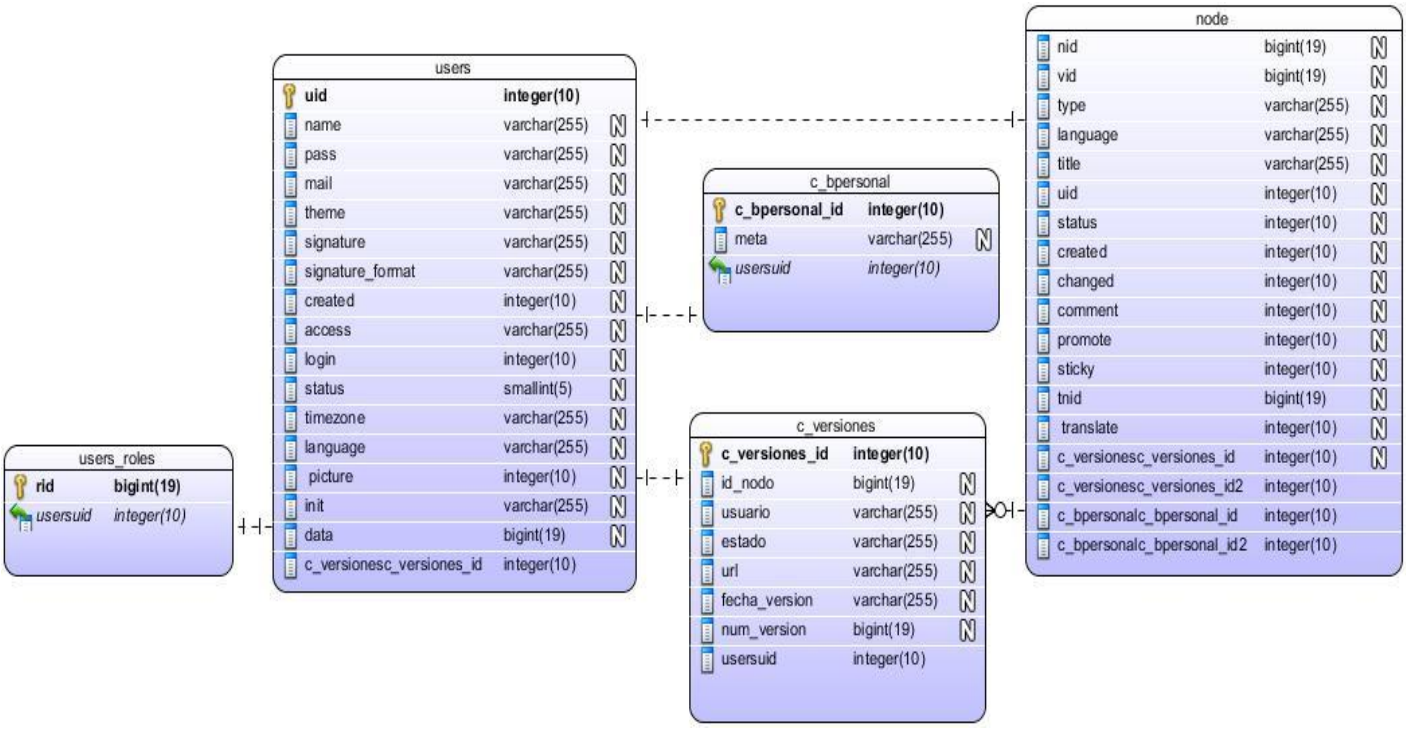


Fig.13 Relaciones entre las tablas principales del sistema

3.3.4 Términos para publicar componentes en la biblioteca

Como la solución propuesta está enfocada al uso de las tecnologías libres y el desarrollo de productos para la web, los componentes que se almacenan deben cumplir con estos requerimientos. El editor, para aprobar un componente, debe verificar que el mismo cumpla con los siguientes aspectos:

- Debe ser funcional.
- Cumplir con la estructura definida para el componente.
- Realizado con tecnologías libres.
- Deben ser componentes multimedia enfocados para la web.

Estos constituyen los requerimientos mínimos que debe tener un componente para que sea aceptado y publicado en la solución propuesta. Si el proyecto decide incluir otros aspectos a tener en cuenta para la publicación, estos deben ser notificados al editor.

3.3.5 Tareas de ingeniería

Las tareas de ingeniería son definidas a partir de las HU entregadas por el cliente. En las mismas es detallado con lenguaje técnico el proceso a partir del cual se implementará la solución a la HU en cuestión. Estas tareas son de gran ayuda a los desarrolladores, pues les brinda un modelo a seguir durante el proceso de implementación.

A continuación son mostradas las HU desarrolladas en la iteración 1, así como las tareas de ingeniería a desarrollar en las mismas (Ver todas las tareas de ingeniería en Anexo 4):

Tabla 9 HU de la iteración 1

Módulo	Historias de usuario (HU)	Tiempo de implementación	
		Estimado	Real
user	1	0.3	0.2
user	2	0.3	0.2
user	3	0.3	0.2
user	4	0.3	0.2
node	5	0.3	0.2
node	6	0.3	0.2
node	7	0.3	0.2
node	8	0.3	0.2
node	9	0.3	0.2
search	11	0.3	0.2

Tabla 10 Tarea de ingeniería de la HU1

Tarea	
Número de tarea: 1	**Número de HU:** 1
Nombre de la tarea: Creación de cuentas de usuarios	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 17/02/2014	**Fecha de terminación:** 17/02/2014
Encargado: Miguel Pérez Fernández	

Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite la creación de cuentas de usuarios. Se debe comprobar mediante la práctica el resultado del estudio.

Tabla 11 Tarea de ingeniería de la HU2

Tarea	
Número de tarea: 1	**Número de HU:** 2
Nombre de la tarea: Modificación de cuentas de usuarios	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 18/02/2014	**Fecha de terminación:** 18/02/2014
Encargado: Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite la modificación de cuentas de usuarios. Se debe comprobar mediante la práctica el resultado del estudio.	

3.4 Fase de pruebas

Uno de los pilares fundamentales de XP es el proceso de pruebas, el cual anima a los desarrolladores a probar constantemente tanto como sea posible. Mediante esta filosofía se reduce el número de errores no detectados, así como el tiempo entre la introducción de éste en el sistema y su detección (82). La metodología seleccionada propone el desarrollo de las siguientes pruebas para verificar la calidad del producto desarrollado(83):

- **Pruebas unitarias:** La prueba de unidad se centra en el módulo, usando la descripción del diseño detallado como guía, se prueban los caminos de control importantes con el fin de descubrir errores dentro del ámbito del módulo.
- **Pruebas de integración:** Las pruebas de integración se llevan a cabo durante la construcción del sistema, involucran a un número creciente de módulos y terminan probando el sistema como conjunto.
- **Pruebas de aceptación:** Las pruebas de aceptación, son las que se hacen con los clientes y define su aceptación del sistema. Son básicamente pruebas funcionales, sobre el sistema completo, y buscan una cobertura de la especificación de requisitos y del manual del usuario.

3.4.1 Pruebas unitarias

Las pruebas unitarias son establecidas antes de escribir el código y son ejecutadas constantemente ante cada modificación del sistema (35). Las pruebas unitarias o pruebas de unidad deben ser construidas antes que el código, permitiéndole a los programadores tener máxima claridad de lo que van a programar antes de hacerlo, así como conocer cada uno de los casos de prueba que deberán pasar, lo que optimizará el trabajo y el código será de mayor calidad (84).

El CMS *Drupal* ofrece la posibilidad de realizar estas pruebas con el módulo *simpleTest*. Dicho módulo es similar a *JUnit/PHPUnit*. Proporciona pruebas por defectos a los módulos del núcleo de *Drupal* pero además se le pueden crear pruebas a los módulos contribuidos. Este tipo de pruebas se diferencia de las demás debido a que no se accede a las base de datos para su ejecución.

En la realización de las pruebas unitarias al sistema se controlaron los módulos implementados bibliocom, m_compartir y m_notificacion y los que trae por defecto el núcleo (menú, *user*, *node*, *views*, *search*) que son los que poseen mayor cantidad de colaboraciones en el sistema.

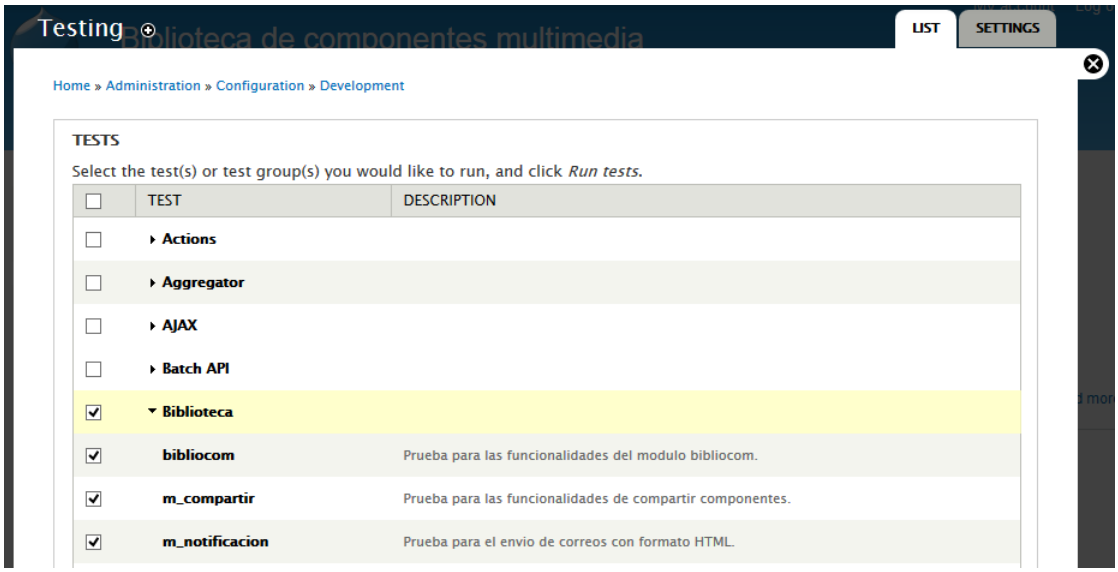


Fig.14 Pruebas unitarias al sistema

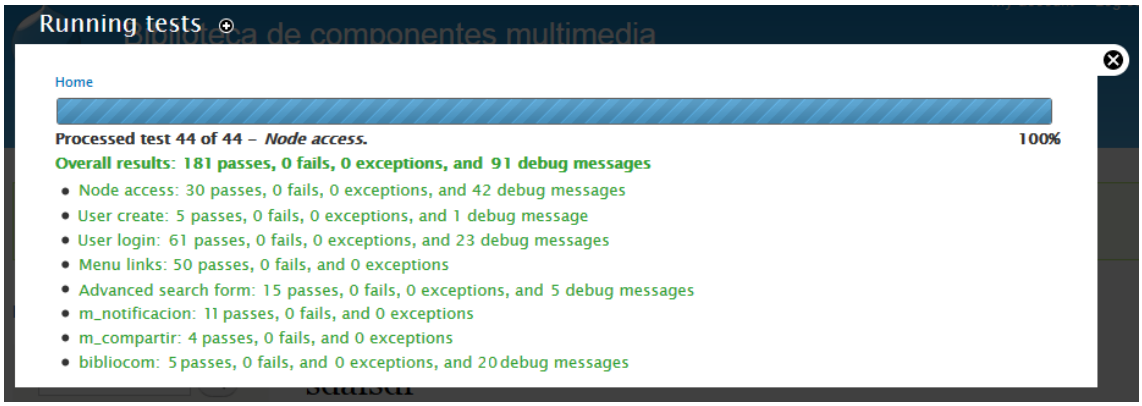


Fig.15 Realización de pruebas unitarias al sistema

MESSAGE	GROUP	FILENAME	LINE	FUNCTION	STATUS
Enabled modules: *bibliocom*	Other	bibliocom.test	29	BibliocomTestCase->setUp()	✓
Created role of name: kqb7AMwK, id: 4	Role	bibliocom.test	30	BibliocomTestCase->setUp()	✓
Created role of name: jOT9HFpP, id: 4	Role	bibliocom.test	30	BibliocomTestCase->setUp()	✓
Created permissions: administer blocks	Role	bibliocom.test	30	BibliocomTestCase->setUp()	✓
User created with name *InkusCsv* and pass *i3qZcrAFbG*	User login	bibliocom.test	30	BibliocomTestCase->setUp()	✓
El modulo define 3 permisos	Other	bibliocom.test	55	BibliocomTestCase->testPermission()	✓
Enabled modules: *bibliocom*	Other	bibliocom.test	29	BibliocomTestCase->setUp()	✓
Created role of name: zXCrjb5b, id: 4	Role	bibliocom.test	30	BibliocomTestCase->setUp()	✓
Created permissions: administer blocks	Role	bibliocom.test	30	BibliocomTestCase->setUp()	✓
User created with name *d3YGl9sV* and pass *DY69mAfNca*	User login	bibliocom.test	30	BibliocomTestCase->setUp()	✓
El modulo define 10 items en el hook menu	Other	bibliocom.test	60	BibliocomTestCase->testhookmenu()	✓
Enabled modules: *bibliocom*	Other	bibliocom.test	29	BibliocomTestCase->setUp()	✓
Valid HTML found on "http://localhost/biblioteca/administracion%20del%20bibliocom"	Browser	bibliocom.test	70	BibliocomTestCase->testCrearVersionesUnprivilegedUser()	✓
GET http://localhost/biblioteca/administracion%20del%20bibliocom returned 404 (7.24 KB).	Browser	bibliocom.test	70	BibliocomTestCase->testCrearVersionesUnprivilegedUser()	✓
Username field found.	Logout	bibliocom.test	74	BibliocomTestCase->testCrearVersionesUnprivilegedUser()	✓
Password field found.	Logout	bibliocom.test	74	BibliocomTestCase->testCrearVersionesUnprivilegedUser()	✓

Fig.16 Resultado de las pruebas unitarias al sistema

Resultados de las pruebas unitarias

Se realizaron las pruebas unitarias para los módulos *menú, user, node, views, search bibliocom, m_compartir y m_notificacion.* En 2 horas y 15 minutos, se obtuvo como resultado 44 pruebas realizadas, 181 métodos probados satisfactoriamente y 91 mensajes de depuración. En el caso del módulo *bibliocom* se probó el código para funcionalidades tales como instalar el módulo, los accesos a los *hook_menu*, los permisos, crear versiones, los roles, la seguridad del sistema. Al utilizar este tipo de prueba automática se ahorró al equipo de desarrollo estar probando los métodos de los módulos seleccionados de forma manual.

3.4.2 Pruebas de aceptación

Las pruebas de aceptación están destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente (82).

Durante las iteraciones, las HU son seleccionadas para su desarrollo de acuerdo a la planificación de cada una de las iteraciones, de esta forma cada HU se convierte en una prueba de aceptación. A continuación, se muestran las pruebas de aceptación propuestas (Ver casos de prueba de aceptación en ANEXO 5):

Tabla 12 Caso de prueba de aceptación HU1

Caso de prueba aceptación	
Código: HU1_P1	**Historia de usuario:** 1
Nombre: Crear cuenta de usuario	
Descripción: Prueba para la funcionalidad que permite crear una cuenta de un usuario en el sistema.	
Condiciones de ejecución: El usuario debe estar anónimo (1) o registrado como administrador (2) en la aplicación. No debe tener creada una cuenta en el sistema.	

Entrada/Pasos de ejecución: (1) El usuario debe dar clic en la opción de *Acceder,* luego *Crear nueva cuenta.* Llena los campos del formulario que aparece (Nombre de usuario, Dirección de correo electrónico y ¿Qué código aparece en la imagen?). En el último campo que es el *captcha* donde debe escribir lo que aparece en la imagen. Finaliza dando clic en el botón *Crear nueva cuenta.* (2) Acceder al menú de administración *Personas*, dar clic en *Añadir usuario,* luego, llenar los campos (Nombre de usuario, Dirección de correo electrónico, Contraseña, Confirmar contraseña, Estado, Roles) y selecciona si desea que se le envíe notificación acerca de su nueva cuenta. Finaliza pulsando el botón *Crear nueva cuenta.*
Resultado esperado: (1) Envía un correo al usuario para que cambie la contraseña. Se crea la cuenta. (2) Si no se seleccionó la opción *notifica por correo al usuario*, el sistema muestra el mensaje *Se creó una nueva cuenta de usuario para "usuario".* No se envía el mensaje de correo. En otro caso muestra el mensaje: *Un mensaje de bienvenida con más instrucciones se le ha enviado por correo electrónico al nuevo usuario.* Se crea la cuenta.
Evaluación de la prueba: Satisfactoria

Tabla 13 Caso de prueba aceptación HU2

Caso de prueba aceptación	
Código: HU2_P1	**Historia de usuario:** 2
Nombre: Modificar cuenta de usuario	
Descripción: Prueba para la funcionalidad que permite al administrador modificar la cuenta de usuario.	
Condiciones de ejecución: El usuario debe estar registrado como administrador. La cuenta a modificar debe existir.	

Entrada/Pasos de ejecución: El usuario accede al menú de administración *Personas*, selecciona la cuenta de usuario a editar, da clic en *Editar.* Aparece un formulario con los campos (Nombre de usuario, Dirección de correo electrónico, Contraseña, Confirmar contraseña, Estado, Roles, Opciones de idioma, Opciones de regionalización). Modifica el campo que va a editar y finaliza pulsando el botón *Guardar.*
Resultado esperado: Se modifica la cuenta. Se re-direcciona hacia el listado de persona del menú de administración del sistema.
Evaluación de la prueba: Satisfactoria

3.4.3 Resultados de las pruebas de aceptación

Se realizaron las pruebas a las funcionalidades en cuatro iteraciones, utilizando los casos de prueba de aceptación. Las no conformidades detectadas se clasificaron en dos grupos, las significativas, que incluyen las que son de funcionalidad y validación, mientras que las no significativas agrupan las de interfaz de usuario (errores ortográficos).

En la primera iteración se encontraron diez no conformidades significativas (de color verde) y siete no conformidades no significativas (de color azul). En la segunda iteración siete significativas y cuatro no significativas, para la tercera iteración tres y cuatro respectivamente. En la última iteración fueron encontradas solamente dos NC significativas. El resultado de las pruebas de aceptación por iteración se muestra en la tabla a continuación.

Tabla 14 No conformidades por iteración

	Iteración 1	Iteración 2	Iteración 3	Iteración 4	TOTAL
Interfaz de usuario	7	4	4	0	15
Funcionalidad	4	2	1	0	7
Validación	6	5	2	2	15
TOTAL	10/7	7/4	3/4	2/0	37

El gráfico que se observa a continuación muestra la relación entre las NC significativas y no significativas

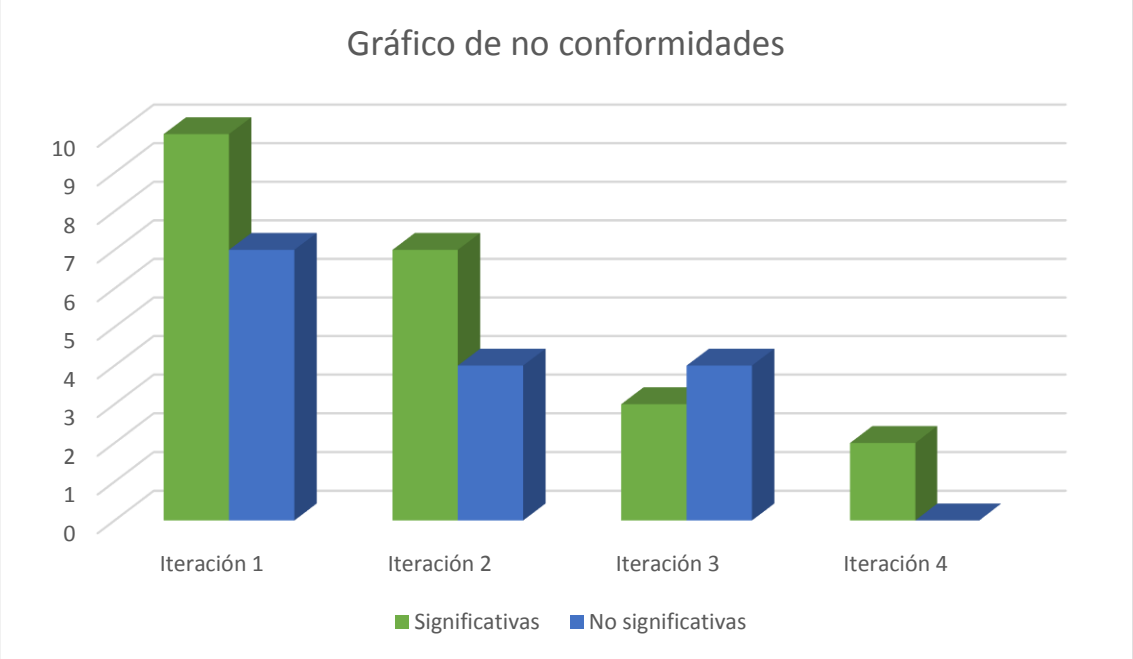


Fig.17 Resultados de las pruebas de aceptación al sistema

3.5 Conclusiones del capítulo

En el presente capítulo fueron reflejados los estándares y patrones de diseño empleados en el desarrollo del sistema. Se presentaron los artefactos correspondientes a las fases de Producción y Pruebas de la metodología de desarrollo utilizada, se abordaron las tareas de ingeniería correspondientes a las HU y se proporcionó una descripción de las tarjetas CRC. Una vez concluida la fase de producción se procedió a la realización de pruebas para comprobar la calidad del sistema, resultando satisfactorias las pruebas unitarias hechas al código y las pruebas de aceptación concebidas a las HU.

CONCLUSIONES GENERALES

Como parte del proceso de investigación y desarrollo de la aplicación, se le dio cumplimiento a todas las tareas y objetivos trazados inicialmente. Por lo que una vez concluida la investigación, se arriba a las siguientes conclusiones:

1- Con la elaboración del marco teórico-metodológico se determinaron los principales conceptos relacionados a las bibliotecas de componentes; además, el estudio realizado permitió un amplio análisis de las herramientas y tecnologías más empleadas en dicha línea de producción.
2- Atendiendo a las características del sistema a desarrollar y las necesidades del cliente, se determinó el uso de la metodología XP para regir el proceso de desarrollo del software.
3- Con los elementos teóricos y prácticos referentes al desarrollo de aplicaciones con tecnología multimedia, se elaboró una biblioteca de componentes para el desarrollo de aplicaciones con tecnología multimedia en el centro FORTES, que permitió organizar y centralizar la documentación y los componentes para el proceso de elaboración de aplicaciones con este tipo de tecnología, favoreciendo su reutilización.
4- Una vez concluida la implementación del sistema, se procedió a la realización de pruebas de aceptación a cada una de las HU y pruebas unitarias al código, obteniendo por resultado la calidad esperada del mismo.

RECOMENDACIONES

Durante el desarrollo de la aplicación se tuvieron en cuenta las necesidades solicitadas por el cliente, pero surgieron ideas que son recomendables para el seguimiento de este producto en aras de ampliar sus funcionalidades:

- Continuar el desarrollo del producto, agregándole nuevas funcionalidades y mejoras a las ya implementadas.
- Permitir que la edición de un componente no sea solamente a través de código, sino que se pueda hacer a través de una interfaz visual.
- Relacionar la aplicación con sistemas de control de versiones como por ejemplo el SVN.
- Migrar paulatinamente la producción de componentes hacia tecnologías libres (HTML5 y CSS3)

REFERENCIAS BIBLIOGRÁFICAS

1. RIZZO, César Labañino y RODRÍGUEZ, Mario del Toro. *Producción de multimedias educativas para la escuela cubana* [En línea]. [Consultado el 11 Febrero 2014]. S.l.: Centro Virtual Cervantes. Disponible en: http://cvc.cervantes.es/ensenanza/formacion_virtual/edicion_digital/toro.htm.

2. NORTHROP, Linda. Software Product Lines Essentials. En: [En línea]. [Consultado el 11 Febrero 2014]. S.l. 2008. Disponible en: http://resources.sei.cmu.edu/asset_files/Presentation/2008_017_001_24246.pdf.

3. SALAVERRÍA, Dr. Ramón. *Aproximación al concepto de multimedia desde los planos comunicativo e instrumental* [En línea]. [Consultado el 11 Febrero 2014]. 2001. S.l.: s.n. Disponible en: http://revistas.ucm.es/index.php/ESMP/article/download/ESMP0101110383A/12866.

4. Multimedia. En: *Cambridge Dictionaries Online* [En línea]. [Consultado el 11 Febrero 2014]. Disponible en: http://dictionary.cambridge.org/dictionary/english-spanish/multimedia?q=multimedia.

5. HOFFSTETTER, Fred. Aplicaciones Multimedia. En: *Universidad de Valencia* [En línea]. [Consultado el 11 Febrero 2014]. Disponible en: http://www.uv.es/~bellochc/pwedu4.htm.

6. *Reutilización* [En línea]. [Consultado el 11 Febrero 2014]. 22. S.l.: Real Academia Española, 2001. Disponible en: http://lema.rae.es/drae/?val=reutilizaci%C3%B3n.

7. NIETO MESA, Msc. Marco Oscar. Importancia de los Objetos de Aprendizaje en la Educación Virtual. En: S.l. 2011.

8. CASAL TERREROS, Julio. Desarrollo de Software basado en Componentes. En: *Microsoft Developer Network* [En línea]. Disponible en: http://msdn.microsoft.com/es-es/library/bb972268.aspx.

9. VALDÉZ SUÁREZ, Michel. *Repositorio de componentes para la línea de Soluciones para Intranets y Portales* [En línea]. La Habana: Universidad de las Ciencias Informáticas, 2012. Disponible en: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_05216_12/1/TD_05216_12.pdf.

10. Alberto del Toro Fuentes. *Biblioteca de componentes para la implementación de la interfaz web del Gestor de Documentos Administrativos eXcriba.* La Habana: Universidad de las Ciencias Informáticas, 2012.

11. MONTILVA, Jonás A., ARAPÉ, Nelson y COLMENARES, Juan Andrés. *Desarrollo de Software basado en componentes* [En línea]. [Consultado el 17 Febrero 2014]. 2003. S.l.: s.n. Disponible en: http://webdelprofesor.ula.ve/ingenieria/jonas/Productos/Publicaciones/Congresos/CAC03%20Desarrollo%20de%20componentes.pdf.

12. GAETAN, Gabriela, BUCCELLA, Agustina y CECHICH, Alejandra. *Un Esquema de Clasificación Facetado para Publicación de Catálogos de Componentes SIG* [En línea]. [Consultado el 17 Febrero 2014]. October 2008. S.l.: SEDICI Repositorio Institucional de la Universidad Nacional de la Plata.

Disponible en: http://sedici.unlp.edu.ar/bitstream/handle/10915/21819/Documento_completo.pdf?sequence=1.

13. MERINO SALVIA, Raul Ernesto. *Propuesta de procedimientos para la gestión organizacional dentro de las Líneas de Producción de Software* [En línea]. [Consultado el 17 Febrero 2014]. La Habana: Universidad de las Ciencias Informáticas, 2009. Disponible en: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_2482_09/1/TD_2482_09.pdf.

14. Software Product Lines. En: *Software Product Lines* [En línea]. [Consultado el 11 Febrero 2014]. Disponible en: http://www.softwareproductlines.com/.

15. Software Product Line. En: *Software Engineering Institute Carnegie Mellon University* [En línea]. [Consultado el 10 Febrero 2014]. Disponible en: http://www.sei.cmu.edu/productlines/.

16. *Repositorio* [En línea]. [Consultado el 17 Febrero 2014]. 22. S.l.: Real Academia Española, 2001. Disponible en: http://lema.rae.es/drae/?val=repositorio.

17. CÓRDOBA GONZÁLEZ, Saray. Concepto de REPOSITORIOS INSTITUCIONALES. En: Universidad de Costa Rica.

18. LYNCH, Clifford A. *Institutional repositories: essential infrastructure for scholarship in the digital age.* [En línea]. [Consultado el 17 Febrero 2014]. 2003. S.l.: Libraries and the Academy. Disponible en: http://muse.jhu.edu/journals/pla/summary/v003/3.2lynch.html.

19. BARRUECO, José Manuel y GARCÍA TESTAL, Cristina. *Repositorios institucionales universitarios: evolución y perspectivas.* [En línea]. [Consultado el 17 Febrero 2014]. 2009. S.l.: s.n. Disponible en: http://www.fesabid.org/zaragoza2009/actas-fesabid-2009/99-107.pdf.

20. Biblioteca. En: *UCPSE Secretaría Educación Honduras* [En línea]. [Consultado el 13 Febrero 2014]. Disponible en: http://www.edu.ucpse.org/index.php/es/tutorials/2013-07-27-19-25-56/46-bibliotecas.

21. *Biblioteca* [En línea]. [Consultado el 17 Febrero 2014]. 22. S.l.: Real Academia Española, 2001. Disponible en: http://lema.rae.es/drae/?val=biblioteca.

22. SÁNCHEZ DÍAZ, Lic. Marlery y VEGA VALDÉS, Dr. Juan Carlos. Bibliotecas electrónicas, digitales y virtuales: tres entidades por definir. En: *SciELO* [En línea]. [Consultado el 17 Febrero 2014]. 2002. Disponible en: http://scielo.sld.cu/scielo.php?pid=S1024-94352002000600005&script=sci_arttext.

23. Leanet de la C. Bello Fabelo. *Portal web para la biblioteca de la Universidad de las Ciencias Informáticas.* La Habana: Universidad de las Ciencias Informáticas, 2012.

24. Biblioteca de componentes visuales. En: *Curso de C++ Builder* [En línea]. [Consultado el 11 Febrero 2014]. Disponible en: http://elvex.ugr.es/decsai/builder/intro/4.html.

25. jQuery UI. En: *jQuery* [En línea]. [Consultado el 12 Febrero 2014]. Disponible en: https://jqueryui.com/.

Bibliografía

26. PSCAD. En: *INDIELEC Ingeniería de Diseño Electrotécnico* [En línea]. [Consultado el 11 Febrero 2014]. Disponible en: http://www.indielec.com/presentacion-cms-4-50-60/.

27. LUJAN, J. y CASARAVILLA, G. *PSCAD-EMTDC: Experiencias de su utilización en la Universidad de la República* [En línea]. [Consultado el 20 Febrero 2014]. S.l.: 5º Encuentro de Potencia, Instrumentación y Medidas. Disponible en: http://iie.fing.edu.uy/~gcp/a20.pdf.

28. LÓPEZ PALMA, Julio Amado. *Biblioteca de componentes de interfaz de usuario para el Sistema de Identificación, Inmigración y Extranjería de la República de Cuba*. La Habana: Universidad de las Ciencias Informáticas, 2011.

29. NIEBLA RODRÍGUEZ, Reynaldo. *Biblioteca de componentes de software* [En línea]. La Habana: Universidad de las Ciencias Informáticas, 2012. [Consultado el 18 Febrero 2014]. Disponible en: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_05810_12/1/TD_05810_12.pdf.

30. Comparación entre ActionScript3.0, HTML5 y CSS3. En: *Google Trends* [En línea]. [Consultado el 20 Febrero 2014]. Disponible en: http://trends.google.com/trends/explore#q=%2Fm%2F02pd26x%2C%20ActionScript%203.0%2C%20CSS3&cmpt=q.

31. Las tendencias de contenido multimedia impulsan soluciones de TI innovadoras. En: [En línea]. [Consultado el 9 Febrero 2014]. Disponible en: http://h30458.www3.hp.com/lar_es/esa/ent/1269519.html.

32. Reynaldo Niebla Rodríguez. *Biblioteca de componentes* [En línea]. [Consultado el 20 Febrero 2014]. S.l.: Universidad de las Ciencias Informáticas, 2012. Disponible en: http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_05810_12.

33. CARVAJAL RIOLA, José Carlos. *Metodologías ágiles: herramientas y modelo de desarrollo para aplicaciones Java EE como metodologia empresarial* [En línea]. [Consultado el 20 Febrero 2014]. 2008. S.l.: s.n. Disponible en: http://upcommons.upc.edu/handle/2099.1/5608.

34. SALAZAR TERRERO, Lirisandra y RODRÍGUEZ, Raúl González. *Solución informática para el portal web "PREPARACIÓN PARA LA DEFENSA FACULTAD # 4."*La Habana: Universidad de las Ciencias Informáticas, 2012.

35. LETELIER, Patricio y PENADÉS, Mª Carmen. *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP)* [En línea]. [Consultado el 20 Febrero 2014]. N 2006. S.l.: s.n. Disponible en: http://dialnet.unirioja.es/servlet/articulo?codigo=1983605.

36. KNIBERG, Henrik. Scrum y XP desde las trincheras Como hacemos Scrum. En: [En línea]. [Consultado el 20 Febrero 2014]. Enterprice Software Development Series. 2007. Disponible en: http://ort-proyecto.googlecode.com/svn/trunk/00%20-%20Material%20de%20Apoyo/03%20-%20Manuales/scrum-y-xp-desde-las-trincheras.pdf.

Bibliografía

37. FERNÁNDEZ GONZÁLEZ, Jorge. *Introducción a las metodologías ágiles. Otras formas de analizar y desarrollar* [En línea]. [Consultado el 04 Marzo 2014]. S.l.: Universidad Abierta de Cataluña. Disponible en: http://www.scribd.com/doc/179356055/Introduccion-a-las-metodologias-agiles-pdf.

38. BELLO FABELO, Leanet de la C. *Portal web para la biblioteca de la Universidad de las Ciencias Informáticas* [En línea]. [Consultado el 04 Marzo 2014]. La Habana: Universidad de las Ciencias Informáticas, 2012. Disponible en: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_05190_12/1/TD_05190_12.pdf.

39. Joomla 3. En: *Joomla* [En línea]. [Consultado el 04 Marzo 2014]. Disponible en: http://www.joomla.org/3/es.

40. WordPress. En: *WordPress Español* [En línea]. [Consultado el 04 Marzo 2014]. Disponible en: http://es.wordpress.org/.

41. CENTRO DE APOYO TECNOLÓGICO A EMPRENDEDORES, FUNDACIÓN PARQUE CIENTÍFICO y TECNOLÓGICO DE ALBACETE. *Estudio de los sitemas de gestión de contenidos web. Análisis de las mejores soluciones del mercado*. S.l.: Junta de Comunidades de Castilla-La Mancha, 2012.

42. MIKOLUK, Kasia. Drupal vs Joomla vs WordPress: Confrontación de CMS. En: *Udemy* [En línea]. [Consultado el 04 Marzo 2014]. Disponible en: https://www.udemy.com/blog/drupal-vs-joomla-vs-wordpress-confrontacion-de-cms/.

43. GIL RODRÍGUEZ, Fran. *Experto en Drupal. Creación y gestión de portales web con Drupal 7*. S.l.: Forcontu S.L, 2012. ISBN 978-84-939410-5-5.

44. About Drupal. En: *Drupal* [En línea]. [Consultado el 04 Marzo 2014]. Disponible en: https://drupal.org/about.

45. THOMSON, Laura y WELLING, Luke. *Desarrollo Web con PHP y MySQL*. España: s.n., 2005. ISBN 978-84-415-1818-6 84-415-1818-1.

46. PostgreSQL. En: *PostgreSQL-es* [En línea]. [Consultado el 14 Marzo 2014]. Disponible en: http://www.postgresql.org.es/sobre_postgresql.

47. ESPINOZA, Humberto. PostgreSQL. Una Alternativa de DBMS Open Source. En: [En línea]. [Consultado el 14 Marzo 2014]. S.l. 2005. Disponible en: http://www.lgs.com.ve/pres/PresentacionES_PSQL.pdf.

48. PECOS MARTÍNEZ, Daniel. POSTGRESQL VS. MYSQL. En: *geekWare* [En línea]. [Consultado el 14 Marzo 2014]. Disponible en: http://danielpecos.com/documents/postgresql-vs-mysql/.

49. Cuadro Comparativo (Apache- IIS). En: [En línea]. 10 Febrero 2014. [Consultado el 10 Febrero 2014]. Disponible en: http://es.scribd.com/doc/83122571/Cuadro-Comparativo-Apache-IIS.

Bibliografía

50. JÁCOME RAMOS, Patricia Elizabeth y PAZMIÑO MOYA, Silvia Paulina. *Desarrollo e implementación de una Intranet/Extranet para una empresa comercializadora de software* [En línea]. [Consultado el 14 Marzo 2014]. 2007. S.l.: Escuela Politécnica Nacional de Quito. Disponible en: http://bibdigital.epn.edu.ec/bitstream/15000/1053/1/CD-1870.pdf.

51. Apache. En: *Instituto Tecnológico de Celaya* [En línea]. [Consultado el 14 Marzo 2014]. Disponible en: http://sisinfo.itc.mx/ITC-APIRGG/Fundamentos_PHP/Introduccion.htm.

52. Lenguaje de programación. En: *Definición.org* [En línea]. [Consultado el 14 Marzo 2014]. Disponible en: http://www.definicion.org/lenguaje-de-programacion.

53. FREEMAN, Adam. *The Definitive Guide to HTML5*. S.l.: Apress, [no date]. ISBN 978-1-4302-3960-4.

54. CSS. En: *Mozilla Developer Network (MDN)* [En línea]. 11 Febrero 2014. [Consultado el 11 Febrero 2014]. Disponible en: https://developer.mozilla.org/es/docs/CSS.

55. ANDREW, Rachel. *The CSS3 Anthology: Take your sites to new heights*. 3ra. S.l.: SitePoint, 2012. ISBN 978-0-9871530-2-9.

56. RODRÍGUEZ TRUJILLO, Isyed de la Caridad, TERRY GONZÁLEZ, Yasirys y CEBALLOS IZQUIERDO, Yasmani. Javascript y la programación orientada a objetos. En: *Serie Científica de la Universidad de las Ciencias Informáticas*. 2011, Vol. 4, pp. 14.

57. FLANAGAN, David. *JavaScript: The Definitive Guide*. 6ta. S.l.: O'Reilly Media, Inc, 2011.

58. BROOKS, David R. *Introduction to HTML and JavaScript for Scientists and Engineers*. S.l.: Springer, 2007. ISBN 978-1-84628-656-8.

59. AJAX Ya. Qué es AJAX? En: [En línea]. [Consultado el 01 Abril 2014]. Disponible en: http://www.ajaxya.com.ar/temarios/descripcion.php?cod=8&punto=1.

60. ORTIZ FIDALGO, Jackeline y RIOS MORALES, Fredy Hector. *Desarrollo de un componente de software para importar reportes desde documentos Excel a Sistemas Gestores de Bases de Datos en el Centro de Investigaciones del Petróleo (CEINPET)* [En línea]. La Habana: Universidad de las Ciencias Informáticas, 2010. Disponible en: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_03630_10/1/TD_03630_10.pdf.

61. Qué es un entorno de desarrollo integrado, IDE. En: *Programación Desarrollo* [En línea]. [Consultado el 01 Abril 2014]. Disponible en: http://programaciondesarrollo.es/que-es-un-entorno-de-desarrollo-integrado-ide/.

62. PhpStorm 7.1.3. En: *Aple.so* [En línea]. [Consultado el 01 Abril 2014]. Disponible en: http://aple.so/phpstorm-7-1-3.

Bibliografía

63. NetBeans IDE. En: *NetBeans* [En línea]. [Consultado el 01 Abril 2014]. Disponible en: https://netbeans.org/features/index.html.

64. NetBeans PHP 7.3. En: *NetBeans* [En línea]. [Consultado el 11 Febrero 2014]. Disponible en: https://netbeans.org/features/php/.

65. RIEHLE, Dirk. *Framework Design A Role Modeling Approach* [En línea]. [Consultado el 01 Abril 2014]. 2000. S.l.: s.n. Disponible en: http://www.riehle.org/computer-science/research/dissertation/diss-a4.pdf.

66. JavaScript Framework for Building Rich Desktop Web Applications. En: *Sencha* [En línea]. [Consultado el 01 Abril 2014]. Disponible en: http://www.sencha.com/products/extjs/.

67. ÁLVAREZ, Miguel Angel. *Manual de jQuery* [En línea]. [Consultado el 01 Abril 2014]. S.l.: s.n. Disponible en: http://dmaspv.com/files/page/07042011180222_manual%20de%20jquery%20en%20pdf%20desarrollowebcom.pdf.

68. jQuery. En: *jQuery* [En línea]. [Consultado el 01 Abril 2014]. Disponible en: http://jquery.com/.

69. PAVÓN MESTRAS, Juan. *Booststrap 3.0* [En línea]. [Consultado el 01 Abril 2014]. S.l.: s.n. Disponible en: http://www.di.sld.cu/documentos/Tutoriales/Bootstrap.pdf.

70. EGUILUZ, Javier. *Desarrollo web ágil con Symfony2*. 2012. S.l.: s.n.

71. POTENCIER, Fabien y ZANINOTTO, Francois. *Symfony 1.2, La Guía Definitiva* [En línea]. [Consultado el 01 Abril 2014]. S.l.: s.n., [no date]. Disponible en: http://librosweb.es/symfony_1_2/.

72. PRESSMAN, Roger S. *Software Engineering: A Practitioner's Approach*. 7ma. S.l.: s.n., [no date]. ISBN 978-0-07-337597-7.

73. PEREZ ANSALDI, Emiliano. Herramientas de prototipado. Qué son y para qué sirven. En: *Asociación de desarrolladores web de España* [En línea]. [Consultado el 01 Abril 2014]. Disponible en: http://www.adwe.es/diseno-web-2/ux/herramientas-de-prototipado-que-son-y-para-que-sirven.

74. ORTEGA MORELL, Yisel Emerita. *Propuesta del sistema de prensa digital de la Universidad de las Ciencias Informáticas* [En línea]. La Habana: Universidad de las Ciencias Informáticas, 2012. Disponible en: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_05147_12/1/TD_05147_12.pdf.

75. Balsamiq Mockups. En: *Balsamiq* [En línea]. 11 Febrero 2014. [Consultado el 11 Febrero 2014]. Disponible en: http://balsamiq.com/products/mockups/.

76. ROQUE PIEDRA, Yanet. *Portal web de información deportiva para la Universidad de las Ciencias Informáticas* [En línea]. La Habana: Universidad de las Ciencias Informáticas, 2013. Disponible en: http://bibliodoc.uci.cu/RDigitales/2013/septiembre/11/TD_06345_13.pdf.

Bibliografía

77. BECK, Ken. *Una explicación de la programación extrema: aceptar el cambio*. S.l.: Addison-Wesley Iberoamericana Espanya, S. A, 2002. ISBN 8478290559.

78. SOMMERVILLE, Ian. *Ingeniería de Software (Parte IV Desarrollo)*. 7ma. S.l.: PEARSON Addison-Wesley, [no date]. ISBN 84-7829-074-5.

79. MEZQUÍA MARIMÓN, Diosbel. *PERFIL DE INSTALACIÓN DE PORTALES INTRANET PARA EL SISTEMA GESTOR DE CONTENIDOS DRUPAL 7* [En línea]. La Habana: Universidad de las Ciencias Informáticas, 2013. Disponible en: http://bibliodoc.uci.cu/RDigitales/2013/septiembre/25/TD_06582_13.pdf.

80. Patrones de Diseño GOF. En: *Mundo Informático* [En línea]. [Consultado el 01 Abril 2014]. Disponible en: http://infow.wordpress.com/category/patrones-de-disenogof/.

81. Capítulo 7.Diseño - 7.2.1 Patrones de diseño en Drupal. En: [En línea]. Disponible en: http://svn.assembla.com/svn/Diploma75/Documentacion/Proyecto/Referencia/Tesis/Cap%C3%ADtulo%207%20-%20Dise%C3%B1o.odt.

82. BRAVO SAAVEDRA, Teddy Ayax y OCHOA LÓPEZ, Renier. *Repositorio de componentes para el desarrollo de software educativo y multimedia* [En línea]. La Habana: Universidad de las Ciencias Informáticas, 2009. Disponible en: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_2330_09/1/TD_2330_09.pdf.

83. GROSS LAUGART, Kirenia y MARTÍNEZ SANABRIA, Liliana. *Desarrollo del módulo para la gestión de notificaciones del Portal Educativo*. La Habana: Universidad de las Ciencias Informáticas, 2013.

84. JOSKOWICZ, José. *Reglas y Prácticas en eXtreme Programming*. Doctorado de Ingeniería Telemática. España: Universidad de Vigo, 2008.

85. Tendencia de ActionScript3.0. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=ActionScript%203.0&cmpt=q.

86. Tendencia de ActionScript3.0 en la India. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=actionscript&geo=IN&cmpt=geo.

87. Tendencia de ActionScript3.0 en USA. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=actionscript&geo=US&cmpt=geo.

88. Comparación de ActionScript3.0 en los años 2008 y 2014. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=ActionScript%203.0&date=1%2F2008%2012m%2C%201%2F2014%2012m&cmpt=date.

89. Tendencia de CSS3. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=CSS3&cmpt=q.

Bibliografía

90. Tendencia de CSS en Cuba. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=CSS&geo=CU&cmpt=geo.

91. Tendencia de CSS3 en la India. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=CSS3&geo=IN&cmpt=geo.

92. Tendencia de CSS3 en USA. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=CSS3&geo=US&cmpt=geo.

93. Comparación de CSS3 en los años 2008 y 2014. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=CSS3&date=1%2F2008%2012m%2C%201%2F2014%2012m&cmpt=date.

94. Tendencia de HTML5. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=%2Fm%2F02pd26x&cmpt=q.

95. Tendencia de HTML5 en Cuba. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=HTML5&geo=CU&cmpt=geo.

96. Tendencia de HTML5 en la India. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=%2Fm%2F02pd26x&geo=IN&cmpt=geo.

97. Tendencia de HTML5 en USA. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=%2Fm%2F02pd26x&geo=US&cmpt=geo.

98. Comparación de HTML5 en los años 2008 y 2014. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=%2Fm%2F02pd26x&date=1%2F2008%2012m%2C%201%2F2014%2012m&cmpt=date.

99. Tendencia de Adobe Flash. En: *Google Trends* [En línea]. [Consultado el 01 Mayo 2014]. Disponible en: http://trends.google.com/trends/explore#q=%2Fm%2F058b6&cmpt=q.

BIBLIOGRAFÍA

1. BUYTAERT, Dries. Drupal. En: *Drupal.org* [En línea]. 31 Enero 2014. [Consultado 31 Enero 2014]. Disponible: https://drupal.org/documentation.

2. ROLYO ALFREDO HERNÁNDEZ LEÓN, Sayda Coello González. *El Proceso de Investigación Científica*. Ciudad de la Habana: Universitaria del Ministro de Educación Superior, 2011. ISBN 978-959-16-1307-3.

3. GIL, David Frank. *Experto en Drupal. Curso de creación y gestión de portales web con drupal 7*. S.l.: Forcontu S, L, 2012. ISBN 978-84-939410-1-7.

4. SOMMERVILLE, Ian. *Ingeniería del software* [En línea]. Sexta. Madrid: Pearson Educación, 2005. [Consultado 29 Enero 2014]. Informática., 6813. ISBN 84-7829-074-5. Disponible: http://eva.uci.cu/file.php/158/Documentos/Bibliografia_general/Textos_Complementarios/Ediciones_del_Sommerville/Sommerville_7ma_edicion/Sommerville_Parte_III_Diseno.pdf. Depósito Legal: M-31.467-2005Impreso en España

5. PRESSMAN, Roger S. *Ingeniería del software.* Sexta edición. La Habana: Félix Varela, 2005. 2

6. SAMPIERI, Roberto Hernández, COLLADO, Carlos Fernández y LUCIO, Pilar Batista. *Metodología de la investigación*. 5ta. S.l.: McGraw-Hill, 2010. ISBN 978-607-15-0291-9.

7. CRAIG LARMAN. *UML y Patrones. Introducción al análisis y diseño orientado a objetos y al proceso unificado.* 2da. S.l.: s.n., 2004.

8. BECK, Kent. *Una explicación de la programación extrema: aceptar el cambio.* España: Addison Wesley, 2002. ISBN 8478290559.

GLOSARIO DE TÉRMINOS

API: Es la abreviatura de *Aplication Programming Interface*. Conjunto de funciones que permite al programador acceder a servicios de una aplicación a través de uso de un lenguaje de programación.

Blogs: Sitio web que incluye, a modo de diario personal de su autor o autores, contenidos de su interés, actualizados con frecuencia y a menudo comentados por los lectores.

Captcha: Prueba de Turing completamente automática y pública para diferenciar computadoras de humanos. Es controlada por una máquina y consiste en una prueba de Turing inversa.

Flash: Tecnología para crear animaciones gráficas vectoriales independientes del navegador. Se requiere de un *plugin* para mostrar dichas animaciones.

Hooks: Son funciones ganchos que permiten a cada módulo tener un medio para comunicarle qué hacer y cuándo debe hacerlo al sistema.

Hosting: Servicio que provee a los usuarios de internet un sistema para poder almacenar información, imágenes, vídeo, o cualquier contenido accesible vía web.

Ítem: Elementos que conforman el hook_menu de *Drupal*.

Licencia GNU/GPL: En español Licencia Pública General, es una licencia creada por la *Free Software Foundation* y orientada principalmente a los términos de distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre.

Marco de trabajo: Fase intermedia que se coloca por encima de la API para aportar mayor sencillez a las aplicaciones. Contiene funcionalidades implementadas que buscan la reutilización de códigos en aras de facilitar el desarrollo de nuevos productos.

Plugin: Son programas que se agregan a otros ya existentes para ofrecer una nueva funcionalidad. Estos programas no funcionan de forma independiente.

Proxy: Programa o dispositivo que realiza una acción en representación de otro.

Glosario de términos

Responsive: Es una filosofía de diseño y desarrollo web, que mediante el uso de estructuras e imágenes fluidas consigue adaptar el sitio web al entorno del usuario.

Scroll: Se denomina *scroll*, desplazamiento o voluta al movimiento en 2D de los contenidos que conforman el escenario de un videojuego o la ventana que se muestra en una aplicación informática.

Sprints: Es el período en el cual se lleva a cabo el trabajo dentro de la metodología de desarrollo de software Scrum.

Spam: Mensajes no solicitados, no deseados o de remitente no conocido.

Triggers: Disparador en una base de datos, es un procedimiento que se ejecuta cuando se cumple una condición establecida al realizar una operación.

Views: Permite generar vistas de datos a través de una interfaz visual.

Widgets: Es una aplicación o programa pequeño, que proveen acceso a funciones frecuentemente usadas y de información visual.

WYSIWYG: Es el acrónimo de *What You See Is What You Get* o lo que es lo mismo “lo que ves es lo que obtienes”.

XHTML: Extensible *HyperText Markup Language* (lenguaje extensible de marcado de hipertexto), es el lenguaje pensado para sustituir a HTML como estándar para las páginas web.

XML: Lenguaje de marcas extensibles es un lenguaje de marcas desarrollados por la W3C. Permite definir la gramática de lenguajes específicos y la comunicación entre aplicación sin importar el lenguaje con el que se comunican.

ANEXO 1: Tendencias de los lenguajes HTML5, CSS3 y ActionScript3

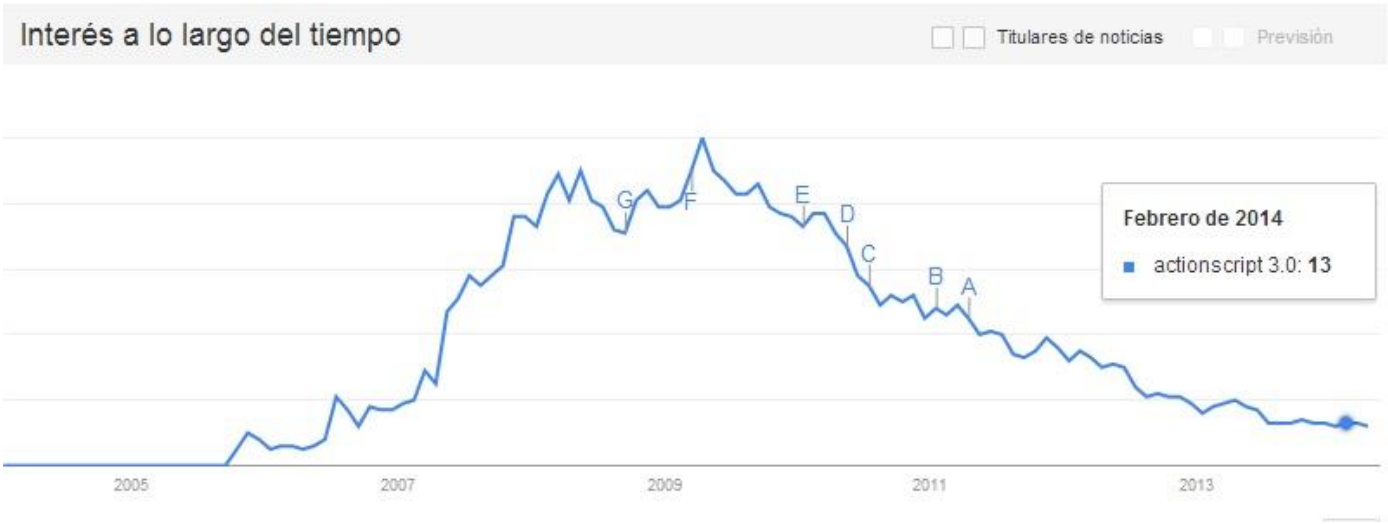


Fig.18 Tendencia del lenguaje ActionScript3.0 a nivel mundial (85)

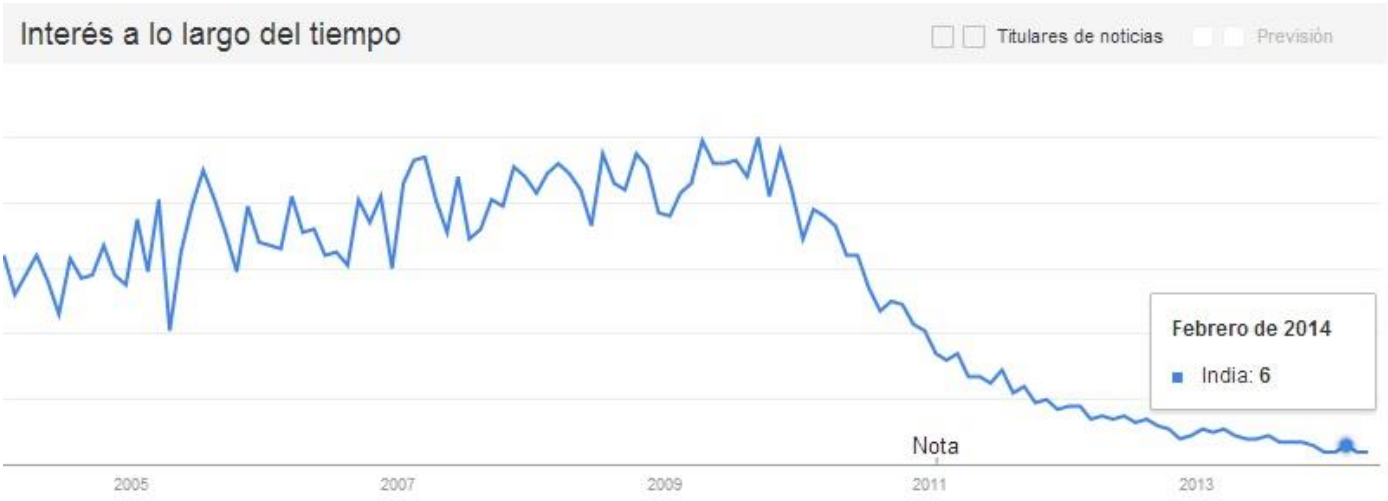


Fig.19 Tendencia del lenguaje ActionScript3.0 en la India (86)

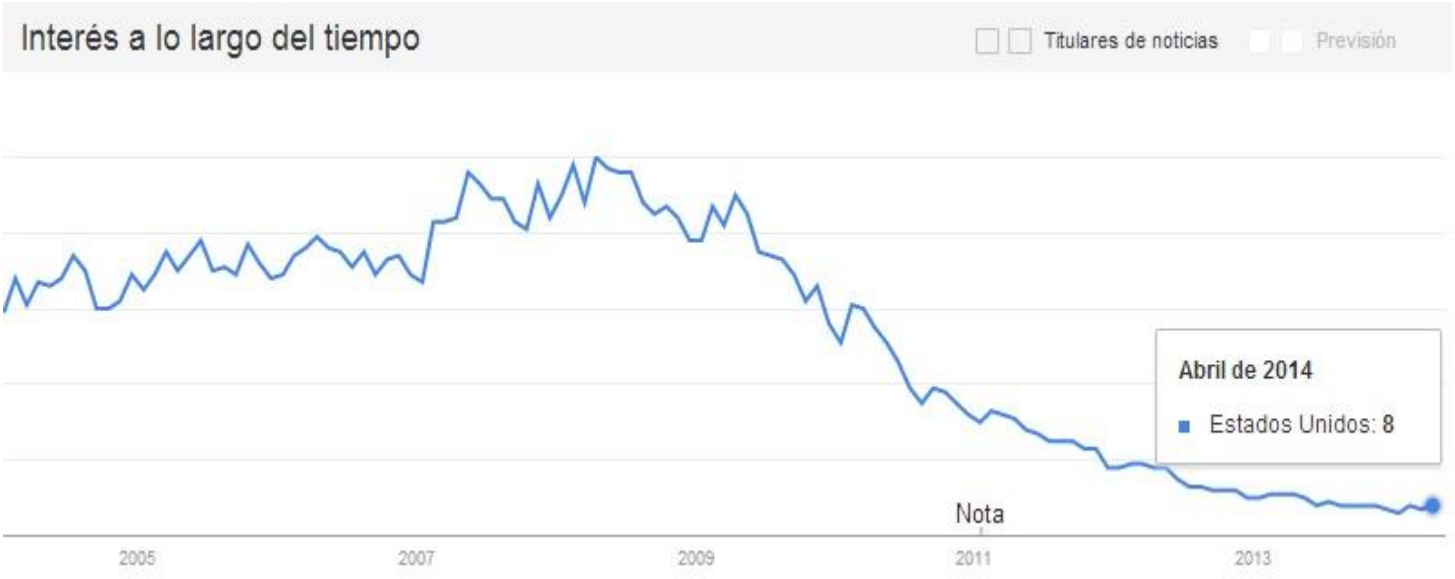


Fig.20 Tendencia del lenguaje ActionScript3.0 en USA (87)

Fig.21 Comparación del lenguaje ActionScript en los años 2008 (azul) y 2014 (rojo) (88)

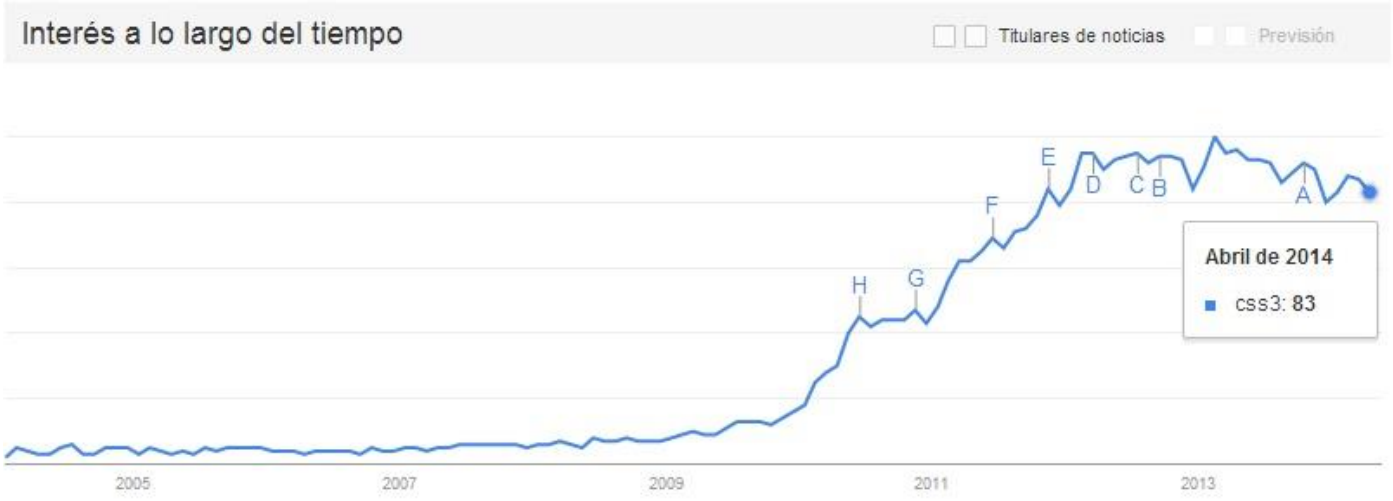


Fig.22 Tendencia del lenguaje CSS3 a nivel mundial (89)

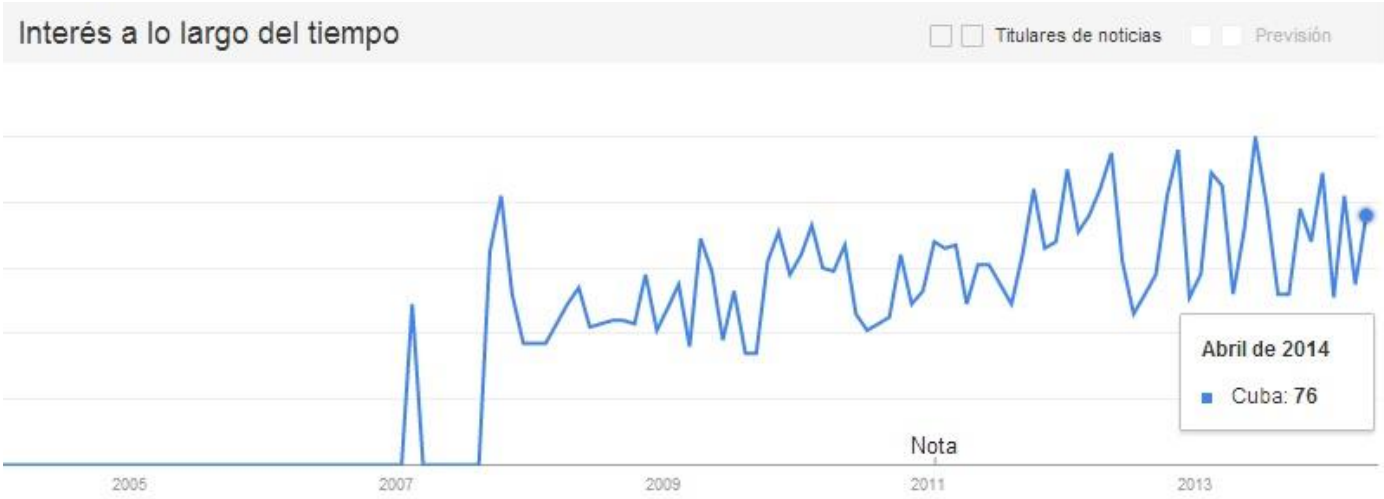


Fig.23 Tendencia del lenguaje CSS en Cuba (90)

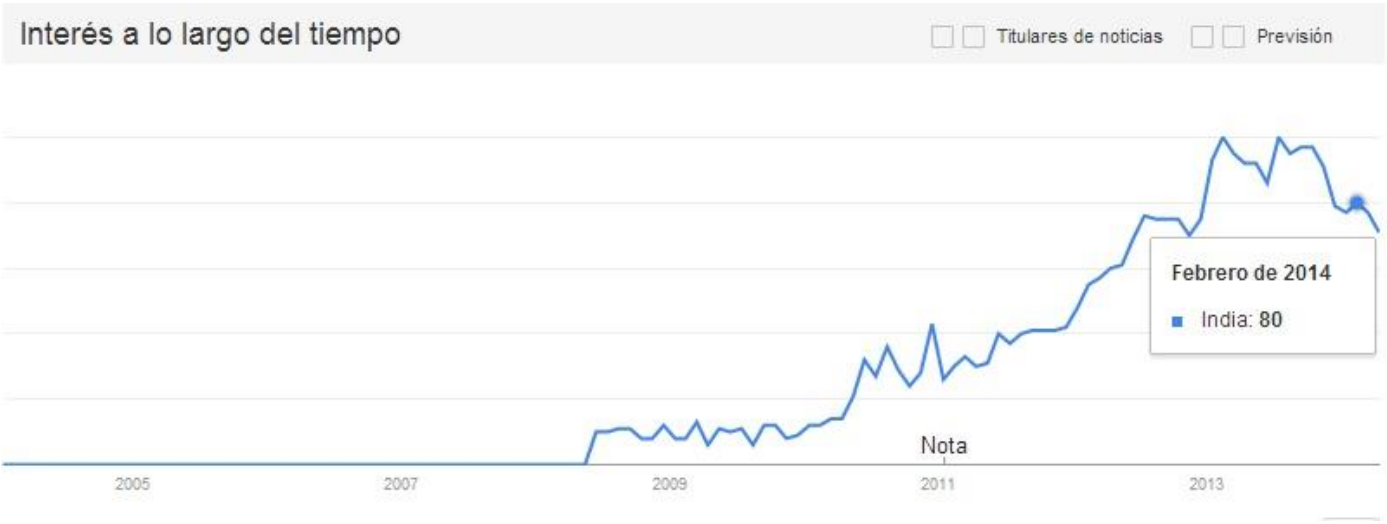


Fig.24 Tendencia del lenguaje CSS3 en la India (91)

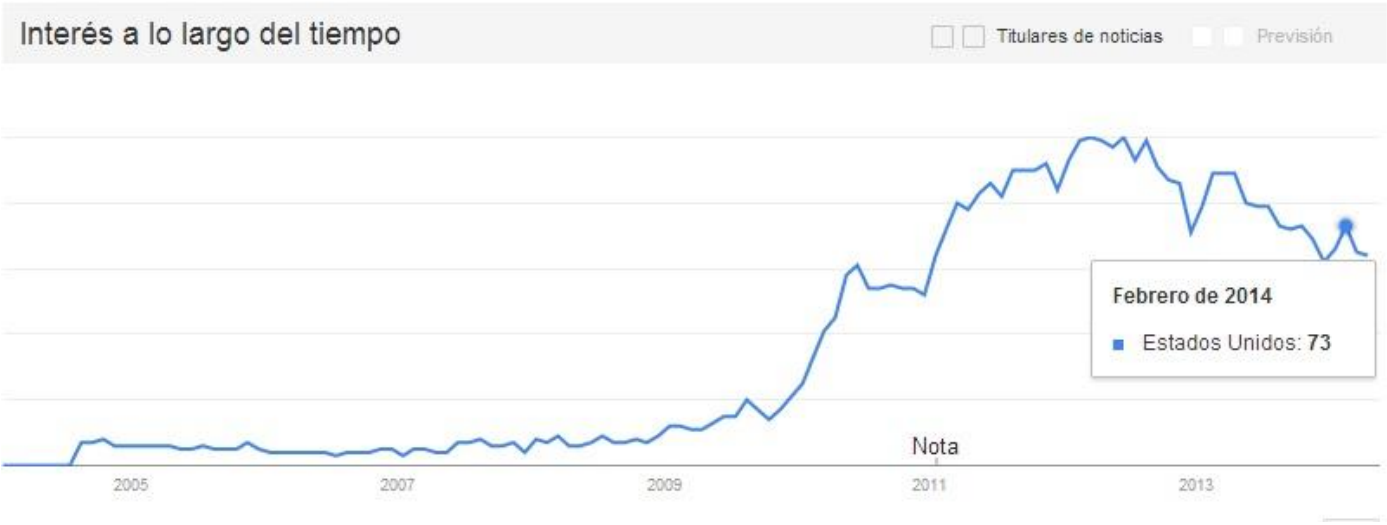


Fig.25 Tendencia del lenguaje CSS3 en USA (92)

Fig.26 Comparación del lenguaje CSS3 en los años 2008 (azul) y 2014 (rojo) (93)

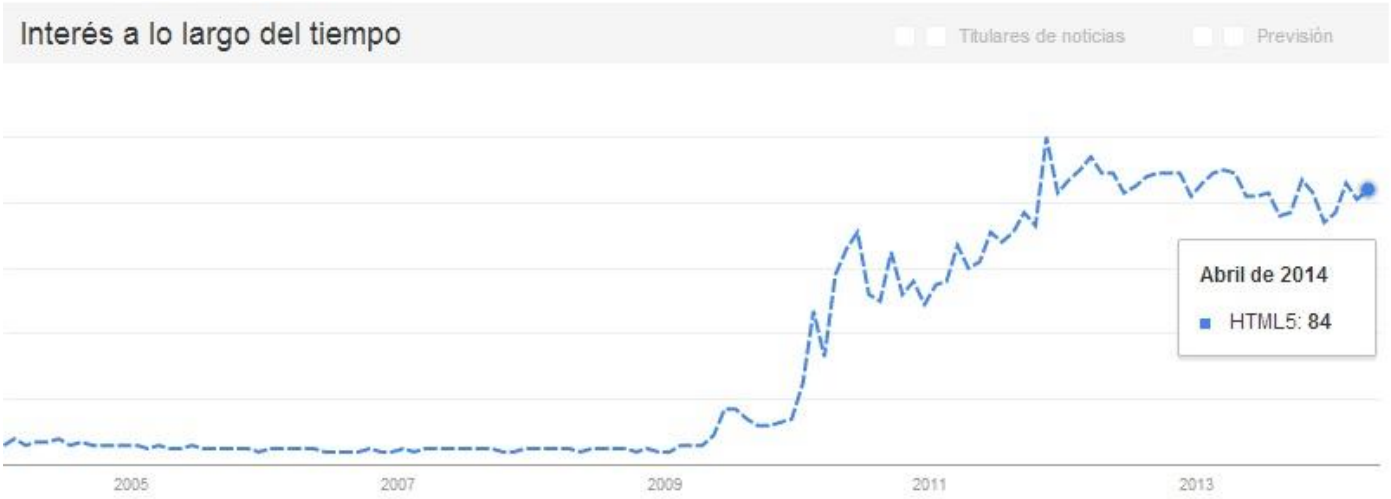


Fig.27 Tendencia del lenguaje HTML5 a nivel mundial (94)

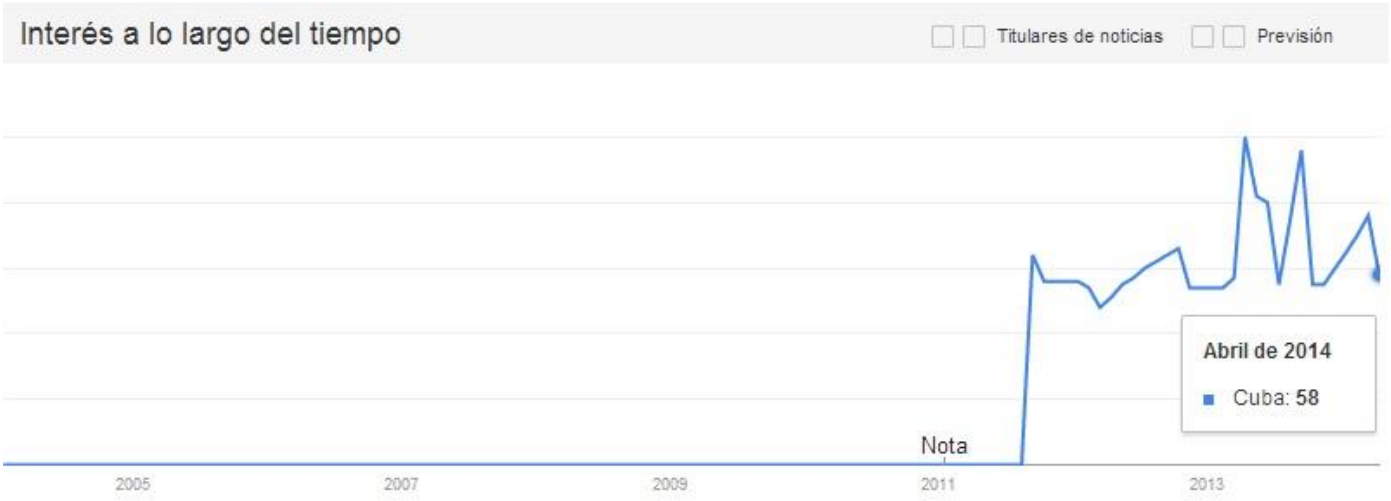


Fig.28 Tendencia del lenguaje HTML5 en Cuba (95)

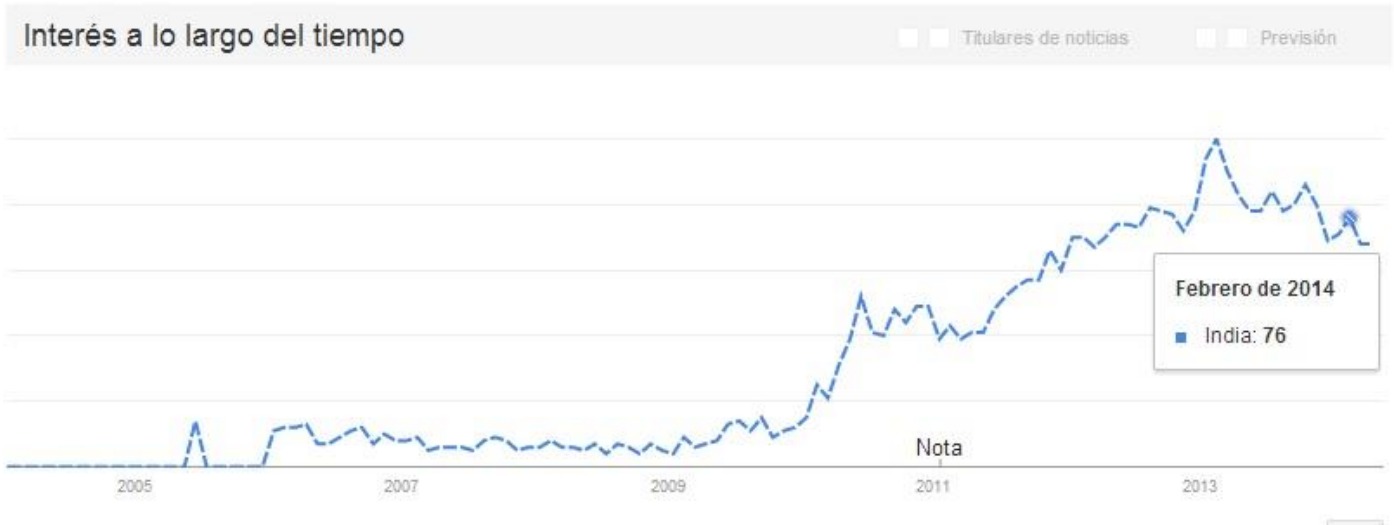


Fig.29 Tendencia del lenguaje HTML5 en la India (96)

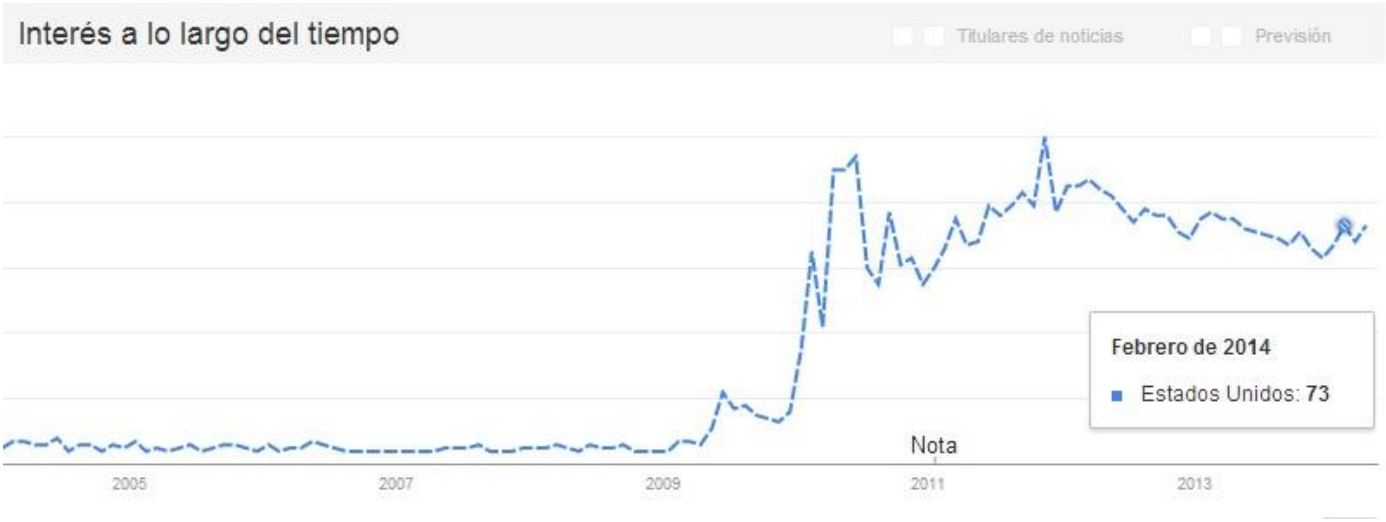


Fig.30 Tendencia del lenguaje HTML5 en USA (97)

Fig.31 Comparación del lenguaje HTML5 en los años 2008(azul) y 2014 (rojo) (98)

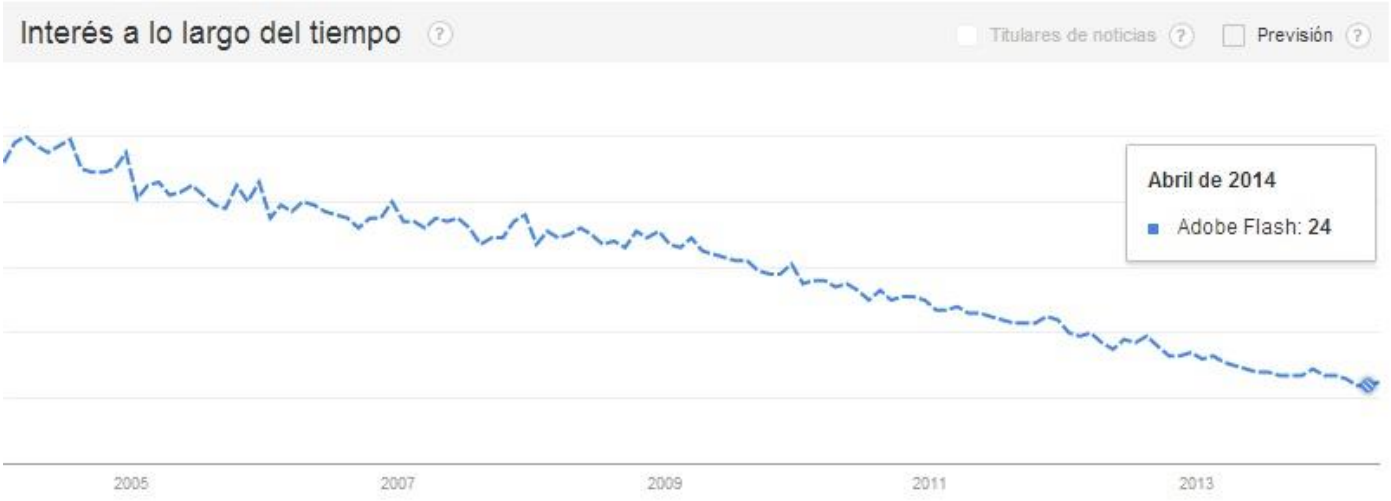


Fig.32 Tendencia de Adobe Flash (99)

ANEXO 2: Historias de usuario

Tabla 15 HU- Modificar cuenta de usuario

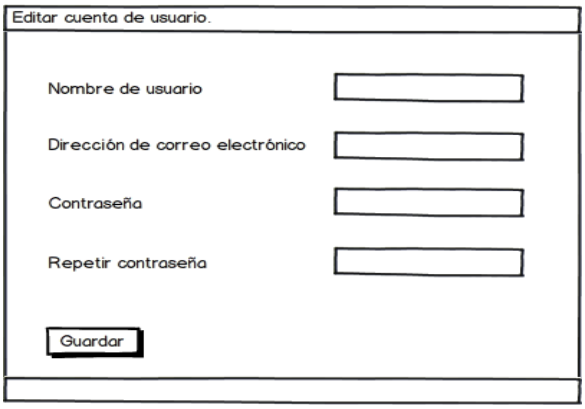
Historia de usuario	
No.: 2	**Nombre:** Modificar cuenta de usuario
Usuario: Registrado	
Prioridad en el negocio: Baja	**Puntos estimados:** 0.3
Riesgo en desarrollo: Baja	**Iteración:** 1
Descripción: El usuario registrado puede modificar su perfil y el administrador además puede editar las cuentas de otros usuarios.	
Prototipo de interfaz: 	

Tabla 16 HU- Eliminar cuenta de usuario

Historia de usuario	
No.: 3	**Nombre:** Eliminar cuenta de usuario
Usuario: Administrador	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3
Riesgo en desarrollo: Baja	**Iteración:** 1
Descripción: Permite al administrador eliminar una cuenta seleccionando el usuario que va a eliminar.	
Prototipo de interfaz:	

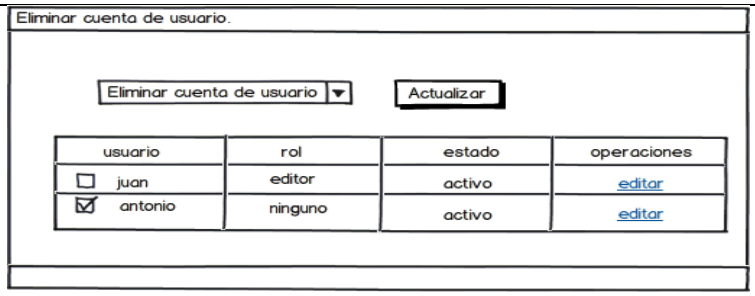


Tabla 17 HU- Autenticar usuario

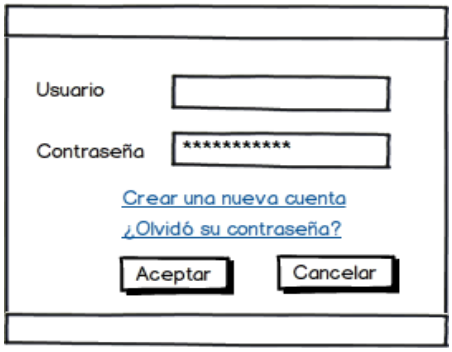
Historia de usuario	
No.: 4	**Nombre:** Autenticar usuario
Usuario: Anónimo	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3
Riesgo en desarrollo: Alta	**Iteración:** 1
Descripción: Permite al usuario autenticarse en el sistema introduciendo los datos usuario y contraseña.	
Observación: El usuario debe haberse creado una cuenta anteriormente.	
Prototipo de interfaz: 	

Tabla 18 HU- Agregar componente

Historia de usuario	
No.: 5	**Nombre:** Agregar componente
Usuario: Registrado	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3

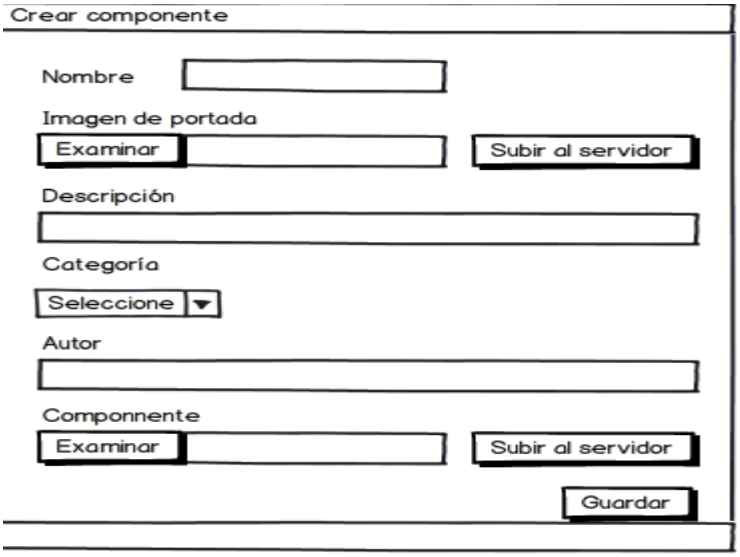
Riesgo en desarrollo: Alta	**Iteración:** 1
Descripción: El usuario registrado puede agregar un componente al introducir los datos: nombre, imagen de portada, categoría (Buscador, Reproductor, *Scroll*, Menú, Galería, otros), descripción, autor y componente y un botón que le permita guardar los datos.	
Prototipo de interfaz: 	

Tabla 19 HU- Modificar componente

Historia de usuario	
No.: 6	**Nombre:** Modificar componente
Usuario: Administrador, editor	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3
Riesgo en desarrollo: Alta	**Iteración:** 1
Descripción: El administrador o el editor seleccionan el componente a editar y modifican los datos que desean.	
Prototipo de interfaz: La misma interfaz de usuario del crear pero con los campos de datos llenos.	

Tabla 20 HU- Eliminar componente

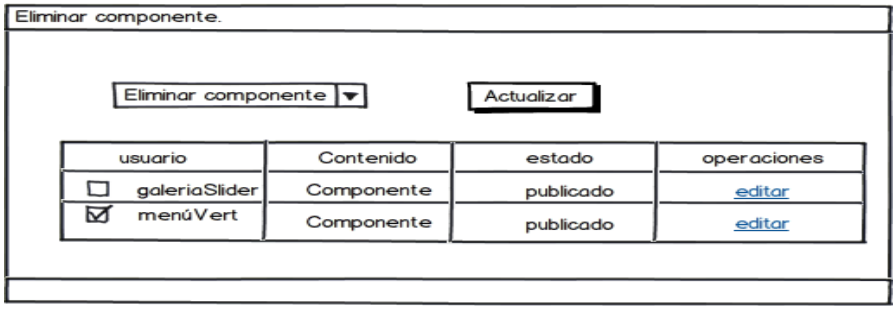
Historia de usuario	
No.: 7	**Nombre:** Eliminar componente
Usuario: Administrador	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3
Riesgo en desarrollo: Baja	**Iteración:** 1
Descripción: El administrador puede eliminar un componente previamente seleccionado.	
Prototipo de interfaz: 	

Tabla 21 HU- Mostrar componente

Historia de usuario	
No.: 8	**Nombre:** Mostrar componente
Usuario: Anónimo, registrado	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3
Riesgo en desarrollo: Media	**Iteración:** 1
Descripción: El usuario accede al menú Componente, el sistema le muestra todos los componentes que existen organizados alfabéticamente por el título.	
Prototipo de interfaz:	

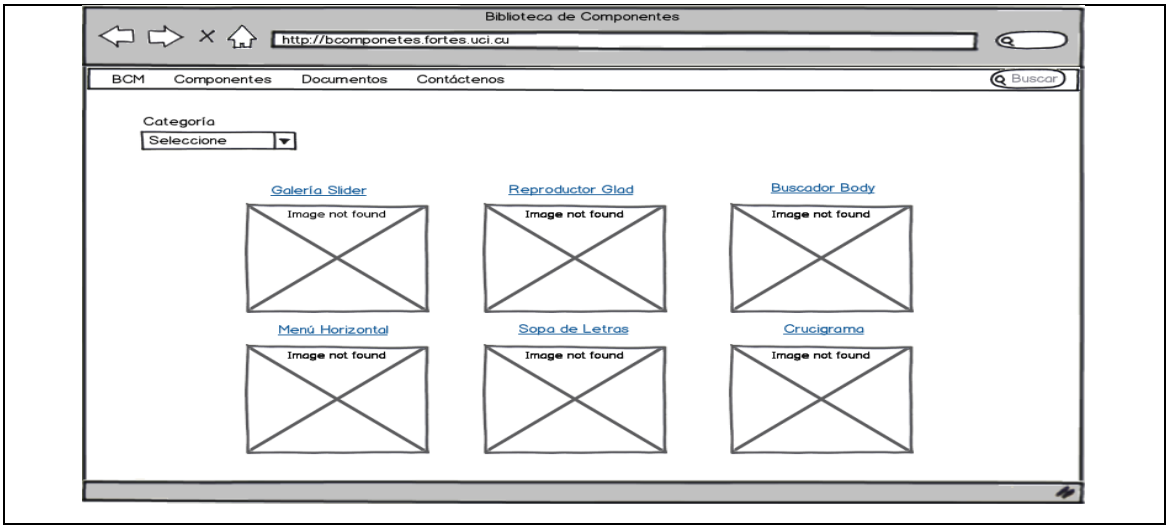


Tabla 22 HU- Filtrar componente por categoría

Historia de usuario	
No.: 9	**Nombre:** Filtrar componente por categoría
Usuario: Anónimo, registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.3
Riesgo en desarrollo: Media	**Iteración:** 1
Descripción: El sistema muestra en la interfaz de todos los componentes una lista de selección donde el usuario selecciona el criterio de filtro (Buscador, Reproductor, *Scroll*, Menú, Galería, otros) y el sistema solo muestra los componentes pertenecientes a esa categoría.	
Prototipo de interfaz: Filtrar por categoría Seleccione	

Tabla 23 HU- Publicar componente

Historia de usuario	
No.: 10	**Nombre:** Publicar componente
Usuario: Editor	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3
Riesgo en desarrollo: Media	**Iteración:** 2

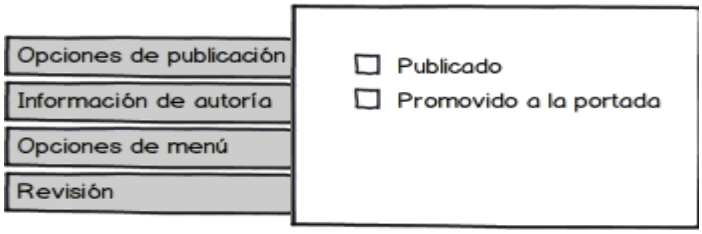
Descripción: Permite al editor publicar el componente si el mismo cumple con los requisitos.
Prototipo de interfaz: 

Tabla 24 HU- Buscar componentes y documentos

Historia de usuario	
No.: 11	**Nombre:** Buscar componentes y documentación
Usuario: Anónimo, registrado	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3
Riesgo en desarrollo: Baja	**Iteración:** 1
Descripción: El sistema muestra en la página de inicio un buscador. El usuario introduce la palabra a buscar y pulsa el botón buscar.	
Observación: Las palabras a buscar deben ser mayor de tres caracteres.	
Prototipo de interfaz: 	

Tabla 25 HU- Descargar componentes

Historia de usuario	
No.: 12	**Nombre:** Descargar componente
Usuario: Anónimo, registrado	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.3
Riesgo en desarrollo: Alta	**Iteración:** 2
Descripción: Permite a los usuarios descargar el componente o las versiones existentes del mismo.	
Observación:	

Tabla 26 HU- Mostrar ficha técnica del componente

Historia de usuario	
No.: 14	**Nombre:** Mostrar ficha técnica del componente
Usuario: Anónimo, registrado	
Prioridad en el negocio: Alta	**Puntos estimados:** 0.4
Riesgo en desarrollo: Alta	**Iteración:** 3
Descripción: Permite a los usuarios ver ficha técnica de los componentes que estén visualizando.	
Observación:	

Tabla 27 HU- Mostrar últimos componentes

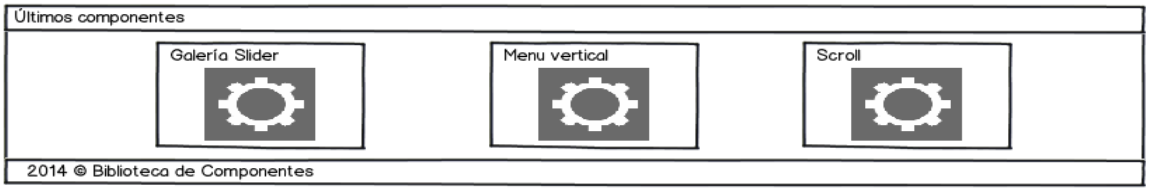
Historia de usuario	
No.: 15	**Nombre:** Mostrar últimos componentes
Usuario: Anónimo, registrado	
Prioridad en el negocio: Baja	**Puntos estimados:** 0.3
Riesgo en desarrollo: Baja	**Iteración:** 2
Descripción: Permite ver cuáles son los últimos componentes creados o agregados en el sistema.	
Prototipo de interfaz: 	

Tabla 28 HU- Agregar documento

Historia de usuario	
No.: 16	**Nombre:** Agregar documento
Usuario: Registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.3
Riesgo en desarrollo: Alta	**Iteración:** 2

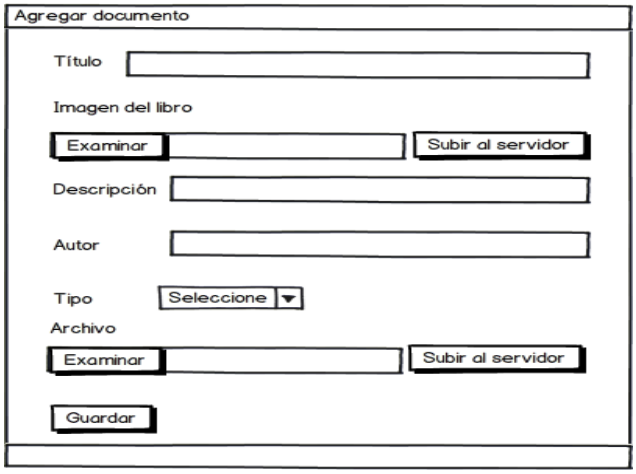
Descripción: El usuario registrado puede agregar documentos a la biblioteca llenando los campos título, imagen del libro, descripción, autor, tipo, archivo.
Prototipo de interfaz: 

Tabla 29 HU- Modificar documento

Historia de usuario	
No.: 17	**Nombre:** Modificar documento
Usuario: Registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.3
Riesgo en desarrollo: Baja	**Iteración:** 2
Descripción: Permite al editor modificar los datos del documento creado. El sistema muestra la misma interfaz que para agregar un documento pero con los campos llenos para que sea editado.	
Observación: El usuario que haya publicado el documento puede modificarlo.	
Prototipo de interfaz: La misma interfaz de usuario de la HU 16 pero con los campos de datos llenos.	

Tabla 30 HU- Eliminar documento

Historia de usuario	
No.: 18	**Nombre:** Eliminar documento

Usuario: Administrador	
Prioridad en el negocio: Media	**Puntos estimados:** 0.3
Riesgo en desarrollo: Baja	**Iteración:** 2
Descripción: El administrador puede eliminar el documento seleccionándolo previamente.	
Observación:	

Tabla 31 HU- Mostrar documentos

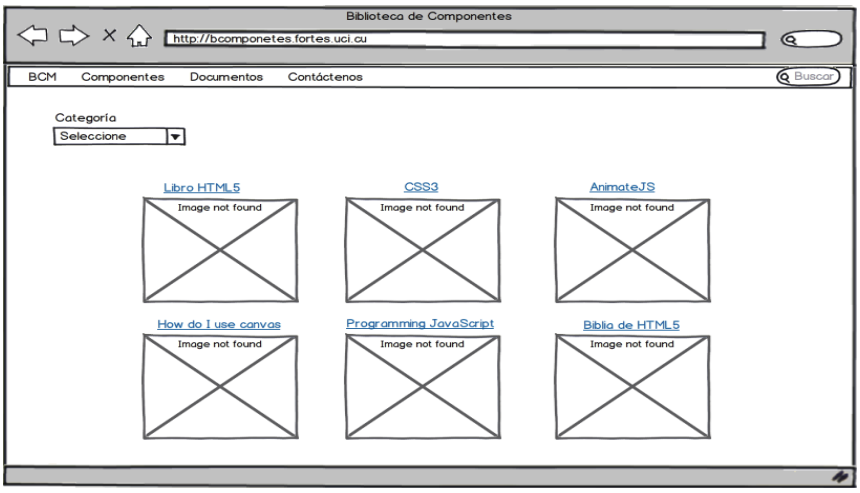
Historia de usuario	
No.: 19	**Nombre:** Mostrar documentos
Usuario: Anónimo, registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.3
Riesgo en desarrollo: Baja	**Iteración:** 2
Descripción: El usuario al acceder al menú Documentos, el sistema muestra toda la documentación que existe organizados por la fecha de publicación. Selecciona un documento y el sistema muestra la ficha técnica de ese documento.	
Observación: Ofrece la posibilidad de descargar el documento.	
Prototipo de interfaz: 	

Tabla 32 HU- Filtrar documentos por tipo

Historia de usuario	
No.: 20	**Nombre:** Filtrar documentos por tipo
Usuario: Anónimo, registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.3
Riesgo en desarrollo: Media	**Iteración:** 2
Descripción: El sistema muestra en la interfaz de todos los documentos una lista de selección donde selecciona el criterio de filtro (CSS, HTML, *JavaScript*, *ActionScript3* y otros) y el sistema solo muestra los documentos pertenecientes a esa clasificación.	
Prototipo de interfaz: 	

Tabla 33 HU- Descargar documento

Historia de usuario	
No.: 21	**Nombre:** Descargar documento
Usuario: Anónimo, registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.3
Riesgo en desarrollo: Media	**Iteración:** 2
Descripción: Permite a los usuarios descargar los documentos que estén visualizando.	
Observación:	

Tabla 34 HU- Enviar notificaciones por correo

Historia de usuario	
No.: 22	**Nombre:** Enviar notificaciones por correo
Usuario: Anónimo, editor	
Prioridad en el negocio: Media	**Puntos estimados:** 0.3
Riesgo en desarrollo: Media	**Iteración:** 2

Descripción: El sistema envía un correo de notificación al usuario editor cada vez que se crea un componente para que sea revisado o cuando se termina de crear una versión. El sistema envía una notificación por correo al usuario que se cree una cuenta, para confirmar la veracidad del mismo.
Observación:

Tabla 35 HU- Crear versiones

Historia de usuario	
No.: 23	**Nombre:** Crear versión de componente
Usuario: Registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.4
Riesgo en desarrollo: Media	**Iteración:** 4
Descripción: Para editar un componente se debe crear una versión. Se debe guardar los datos referentes a la nueva versión.	
Observación:	

Tabla 36 HU- Eliminar versión

Historia de usuario	
No.: 24	**Nombre:** Eliminar versión de componente
Usuario: Administrador, editor	
Prioridad en el negocio: Media	**Puntos estimados:** 0.4
Riesgo en desarrollo: Media	**Iteración:** 4
Descripción: El administrador elimina la versión del sistema.	
Observación: El sistema debe eliminar todas las dependencias que tenga la versión a borrar.	

Tabla 37 HU- Subir versión de componente

Historia de usuario	
No.: 25	**Nombre:** Subir versión de componente

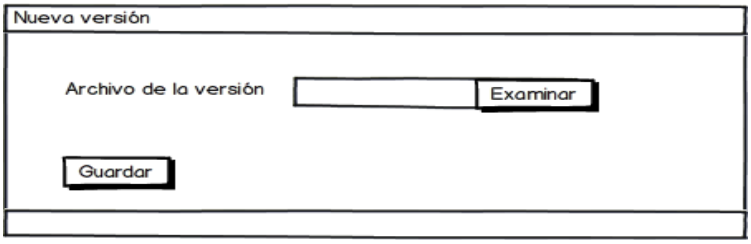
Usuario: Registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.4
Riesgo en desarrollo: Media	**Iteración:** 4
Descripción: Permite al usuario subir al sistema versiones comprimidas (.*zip*) que haya editado fuera de la aplicación.	
Prototipo de interfaz: 	

Tabla 38 HU- Compartir componente

Historia de usuario	
No.: 26	**Nombre:** Compartir componente
Usuario: Registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.4
Riesgo en desarrollo: Media	**Iteración:** 4
Descripción: El usuario puede compartir o dar referencia de un componente en la biblioteca a través de correo. El sistema le envía un correo al destinatario con un enlace para mostrar el componente y otro para descargarlo sin necesidad de entrar a la biblioteca.	
Prototipo de interfaz:	

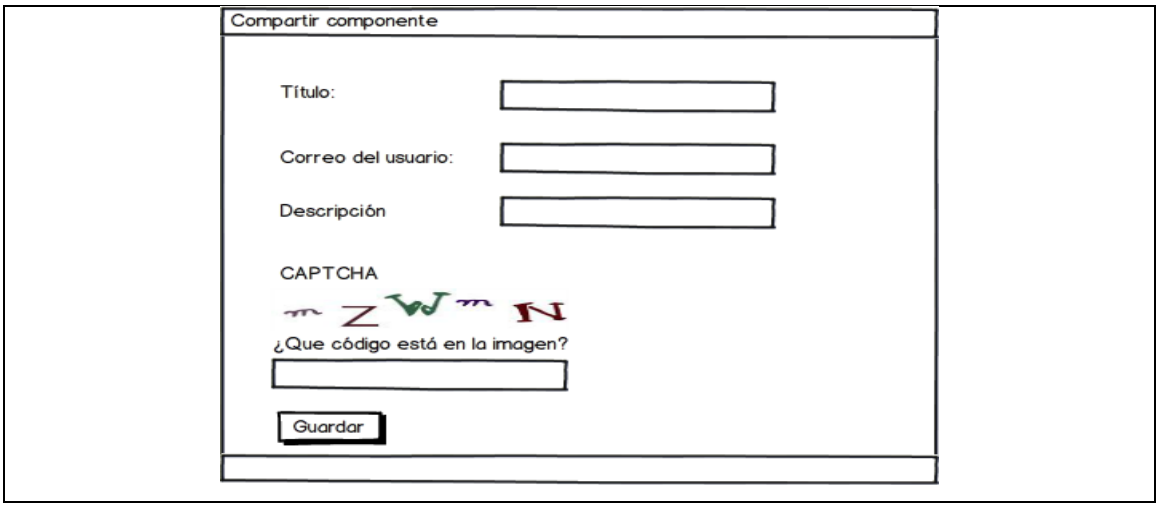


Tabla 39 HU- Añadir componente a la biblioteca personal

Historia de usuario	
No.: 27	**Nombre:** Añadir componente a la biblioteca personal
Usuario: Registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.4
Riesgo en desarrollo: Media	**Iteración:** 4
Descripción: El sistema posibilita al usuario guardar en una biblioteca personal sus componentes favoritos, con el objetivo de tenerlos mejor organizados y centralizados.	
Observación:	

Tabla 40 HU- Eliminar componente de la biblioteca personal

Historia de usuario	
No.: 28	**Nombre:** Eliminar componente de la biblioteca personal
Usuario: Registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.4
Riesgo en desarrollo: Media	**Iteración:** 4

Descripción: El usuario desde su biblioteca personal elimina el componente o la versión que desee.
Observación:

Tabla 41 HU- Mostrar biblioteca personal

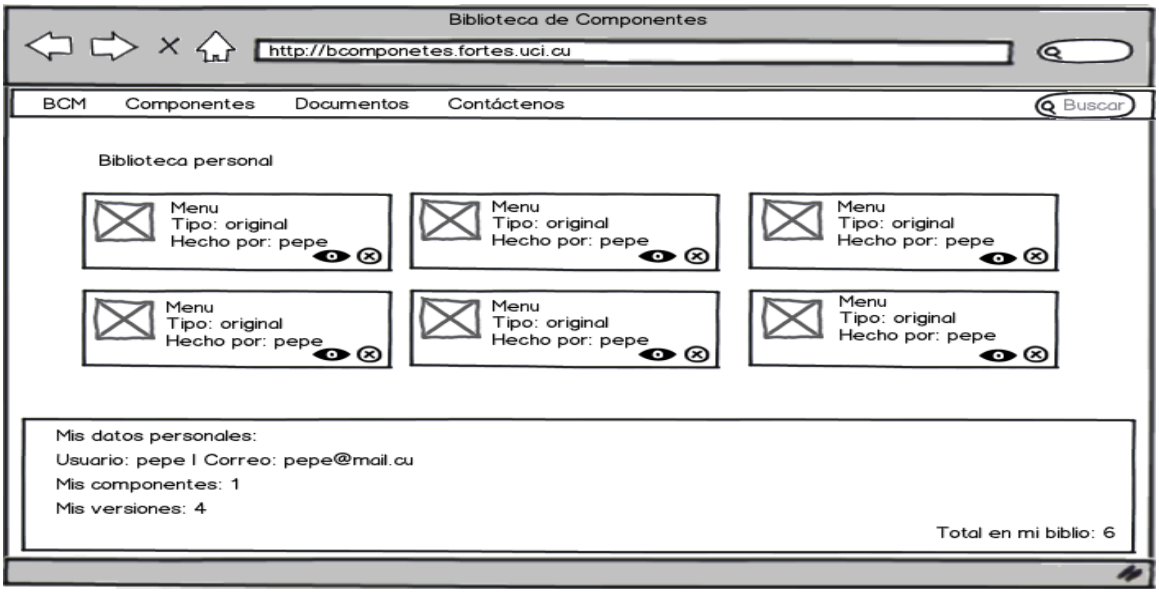
Historia de usuario	
No.: 29	**Nombre:** Mostrar biblioteca personal
Usuario: Registrado	
Prioridad en el negocio: Media	**Puntos estimados:** 0.4
Riesgo en desarrollo: Media	**Iteración:** 4
Descripción: Visualiza al usuario todos los componentes y versiones que él ha agregado a su biblioteca de forma ordenada según se fueron agregando a la biblioteca personal.	
Prototipo de interfaz: 	

Tabla 42 HU- Interfaz

Historia de usuario	
No.: 30	**Nombre:** Interfaz

Descripción: El diseño de la interfaz visual debe ser minimalista, sin uso excesivo de animaciones e imágenes de alta calidad o tamaño que perjudiquen el tiempo de respuesta de las peticiones que se realicen a la aplicación. La interfaz debe ser diseñada de forma tal que permita una sencilla navegación y el fácil entendimiento de las funcionalidades que brinda la biblioteca, además de poseer colores refrescantes para una mejor interacción entre el usuario y la aplicación.
Observación:

Tabla 43 HU- Usabilidad

Historia de usuario	
No.: 31	**Nombre:** Usabilidad
Descripción: La aplicación debe ser diseñada de forma que los usuarios que hagan uso de la misma obtengan los conocimientos necesarios en el menor tiempo posible, para evitar el entrenamiento de los usuarios para trabajar con el sistema y facilitar una mejor explotación de sus funcionalidades.	
Observación:	

Tabla 44 HU- Software

Historia de usuario	
No.: 32	**Nombre:** Software
Descripción: Para instalar el sistema se debe contar: **Máquina servidor:** Sistema Operativo: *Windows* o *Linux*. Servidor web: Apache 2. Lenguaje PHP versión 5 o superior. Gestor de base de datos: *PostgreSQL* 9.1. **Máquina cliente:** Sistema operativo: *Windows* XP o superior, cualquier distribución de Linux.	

Navegador web: *Firefox* versión 4.0 o superior, *Google Chrome* 27.0 o superior, *Internet Explorer* 9.0 o superior.
Observación:

Tabla 45 HU- Hardware

Historia de usuario	
No.: 33	**Nombre:** Hardware
Descripción: La máquina servidor deberá contar con un microprocesador *Dual Core*, 2GB de RAM, un mínimo de 3 GB de disco duro, se debe considerar la posible expansión del volumen de datos.	
Observación:	

Tabla 46 HU- Rendimiento

Historia de usuario	
No.: 34	**Nombre:** Rendimiento
Descripción: Rapidez y eficiencia tanto en los tiempos de respuesta como en la velocidad de procesamiento. Garantizar velocidad estable de navegación a nivel de aplicación. Proporcionar tiempos de respuesta mínimos que no excedan de los 3 segundos, para los procesos del sistema.	
Observación:	

Tabla 47 HU- Seguridad

Historia de usuario	
No.: 35	**Nombre:** Seguridad
Descripción: Acceder a la información autorizada de acuerdo al rol de cada usuario en el sistema. Evitar que la información en el sistema sea modificada por usuarios anónimos.	
Observación:	

Tabla 48 HU- Disponibilidad

Historia de usuario	
No.: 36	**Nombre:** Disponibilidad
Descripción: Cada usuario con facultades debe tener acceso pleno al uso de la aplicación 7 días a la semana, 365 días al año.	
Observación:	

Tabla 49 HU- Portabilidad

Historia de usuario	
No.: 37	**Nombre:** Portabilidad
Descripción: El sistema debe ser capaz de ejecutarse en diferentes entornos sin sufrir cambios en su estructura.	
Observación:	

Tabla 50 HU- Adaptabilidad

Historia de usuario	
No.: 38	**Nombre:** Adaptabilidad
Descripción: El sistema debe ser adaptable a la resolución de la pantalla de los diferentes dispositivos que utilice el cliente.	
Observación:	

Tabla 51 HU- Capacidad

Historia de usuario	
No.: 39	**Nombre:** Capacidad
Descripción: Considerar características técnicas mínimas para la ejecución en clientes. El sistema debe permitir aproximadamente 120 conexiones de manera simultánea.	
Observación:	

ANEXO 3: Tarjetas CRC

Tabla 52 Tarjeta CRC Gestionar componentes y documentos

Módulo node	
Responsabilidades	**Colaboraciones**
Gestionar componente Publicar componente Mostrar componente Filtrar componentes por categoría Últimos componentes Gestionar documento Mostrar documentación Filtrar documento por tipo Descargar documento	*user* *view* *block* *ctools* *views fluid grid*

Tabla 53 Tarjeta CRC Correos HTML

Módulo m_notificacion	
Responsabilidades	**Colaboraciones**
Enviar notificaciones por correo con formato HTML	*class.smtp* *class.phpmailer*

Tabla 54 Tarjeta CRC Gestionar usuarios

Módulo user	
Responsabilidades	**Colaboraciones**
Autenticar usuario Gestionar usuario Asignar roles	*user*

Tabla 55 Tarjeta CRC Buscar componentes y documentos

Módulo search	
Responsabilidades	**Colaboraciones**

Buscar componente Buscar documentos	*node* *search*

Tabla 56 Tarjeta CRC Enviar notificaciones por correo

Módulo SMTP	
Responsabilidades	**Colaboraciones**
Notificación de componente en publicación Notificación de creación de cuentas Notificación de olvidar contraseña	*SMTP* *user* *node*

ANEXO 4: Tareas de ingeniería de las HU

Tabla 57 Tareas de ingeniería HU3

Tarea	
Número de tarea: 1	**Número de HU:** 3
Nombre de la tarea: Eliminación de cuentas de usuarios	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 19/02/2014	**Fecha de terminación:** 19/02/2014
Encargado: Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite la eliminación de una cuenta de usuario y aplicar el estudio realizado en la implementación de la HU.	

Tabla 58 Tarea de ingeniería HU4

Tarea	
Número de tarea: 1	**Número de HU:** 4
Nombre de la tarea: Autenticación de usuarios	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 20/02/2014	**Fecha de terminación:** 20/02/2014
Encargado: Miguel Pérez Fernández	
Descripción: Se deben implementar las funcionalidades que permitan a un usuario autenticarse en el sistema.	

Tabla 59 Tareas de ingeniería HU5

Tarea	
Número de tarea: 1	**Número de HU:** 5
Nombre de la tarea: Crear un tipo de contenido	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 21/02/2014	**Fecha de terminación:** 21/02/2014
Encargado: Miguel Pérez Fernández	

Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite crear un tipo de contenido y aplicar el estudio realizado en la implementación de la HU.

Tabla 60 Tareas de ingeniería HU5

Tarea	
Número de tarea: 2	**Número de HU:** 5
Nombre de la tarea: Agregar campos a un contenido	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 21/02/2014	**Fecha de terminación:** 21/02/2014
Encargado: Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite agregar campos a un tipo de contenido y aplicar el estudio realizado en la implementación de la HU.	

Tabla 61 Tareas de ingeniería HU5

Tarea	
Número de tarea: 3	**Número de HU:** 5
Nombre de la tarea: Enviar correo de notificación	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 21/02/2014	**Fecha de terminación:** 21/02/2014
Encargado: Alejandro Rodríguez de la Cruz\Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo *Drupal* envía un correo después de guardar un tipo de contenido y aplicar el estudio realizado en la implementación de la HU.	

Tabla 62 Tarea de ingeniería HU6

Tarea	
Número de tarea: 1	**Número de HU:** 6
Nombre de la tarea: Modificar componente	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 24/02/2014	**Fecha de terminación:** 24/02/2014
Encargado: Miguel Pérez Fernández	

Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite editar los contenidos que se encuentran en el sistema y aplicar el estudio realizado en la implementación de la HU.

Tabla 63 Tarea de ingeniería HU7

Tarea	
Número de tarea: 1	**Número de HU:** 7
Nombre de la tarea: Eliminar componente	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 25/02/2014	**Fecha de terminación:** 25/02/2014
Encargado: Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* elimina los contenido que se encuentran en el sistema y aplicar el estudio realizado en la implementación de la HU.	

Tabla 64 Tareas de ingeniería HU8

Tarea	
Número de tarea: 1	**Número de HU:** 8
Nombre de la tarea: Mostrar vistas con componente	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 26/02/2014	**Fecha de terminación:** 26/02/2014
Encargado: Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* muestra los contenidos a través de vistas (*views*) y aplicar el estudio realizado en la implementación de la HU.	

Tabla 65 Tareas de ingeniería HU8

Tarea	
Número de tarea: 2	**Número de HU:** 8
Nombre de la tarea: Darle estilo a la vista	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 26/02/2014	**Fecha de terminación:** 26/02/2014
Encargado: Miguel Pérez Fernández	

Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* visualiza el contenido en forma de bloque que se asemejen a una galería y aplicar el estudio realizado en la implementación de la HU.

Tabla 66 Tareas de ingeniería HU8

Tarea	
Número de tarea: 3	**Número de HU:** 8
Nombre de la tarea: Paginación de la vista	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 26/02/2014	**Fecha de terminación:** 26/02/2014
Encargado: Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* hace la paginación de la vista sin necesidad de recargar la página y aplicar el estudio realizado en la implementación de la HU.	

Tabla 67 Tareas de ingeniería HU9

Tarea	
Número de tarea: 1	**Número de HU:** 9
Nombre de la tarea: Filtrar contenido	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 27/02/2014	**Fecha de terminación:** 27/02/2014
Encargado: Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* a través de las vistas, filtra los contenidos por un campo previamente definido y aplicar el estudio realizado en la implementación de la HU.	

Tabla 68 Tareas de ingeniería HU11

Tarea	
Número de tarea: 1	**Número de HU:** 11
Nombre de la tarea: Buscar contenido	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 03/03/2014	**Fecha de terminación:** 03/03/2014

Encargado: Miguel Pérez Fernández
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite buscar un contenido por los nodos existentes en el sistema y cada qué tiempo son indexados estos contenidos. Aplicar el estudio realizado en la implementación de la HU.

Tabla 69 HU de la iteración 2

Módulo	Historias de usuario (HU)	Tiempo de implementación	
		Estimado	Real
node	10	0.3	0.3
bibliocom	12	0.3	0.2
node	15	0.3	0.2
node	16	0.3	0.3
node	17	0.3	0.3
node	18	0.3	0.2
node	19	0.3	0.3
node	20	0.3	0.2
node	21	0.3	0.3
m_notificacion y SMTP	22	0.3	0.2

Tabla 70 Tareas de ingeniería HU10

Tarea	
Número de tarea: 1	**Número de HU:** 10
Nombre de la tarea: Publicar contenido	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 05/03/2014	**Fecha de terminación:** 06/03/2014
Encargado: Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* publica los contenidos en el sistema. Aplicar el estudio realizado en la implementación de la HU.	

Tabla 71 Tareas de ingeniería HU10

Tarea	
Número de tarea: 2	**Número de HU:** 10
Nombre de la tarea: Publicar versiones	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 05/03/2014	**Fecha de terminación:** 06/03/2014
Encargado: Miguel Pérez Fernández	
Descripción: Se implementa la funcionalidad para publicar las versiones, la misma realiza una actualización en el campo estado de la tabla c_versiones. El estado es cambiado a *pub*[2]. Si el estado de esa versión es *pub* entonces se muestra en el cintillo de versiones del sistema.	

Tabla 72 Tareas de ingeniería HU12

Tarea	
Número de tarea: 1	**Número de HU:** 12
Nombre de la tarea: Almacenamiento de los componentes	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 06/03/2014	**Fecha de terminación:** 06/03/2014
Encargado: Alejandro Rodríguez de la Cruz\Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a dónde el CMS *Drupal* almacena en la base de datos la dirección de los componentes subidos al sistema. Aplicar el estudio realizado en la implementación de la HU.	

Tabla 73 Tareas de ingeniería HU12

Tarea	
Número de tarea: 2	**Número de HU:** 12
Nombre de la tarea: Descargar componente original	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 06/03/2014	**Fecha de terminación:** 06/03/2014

[2] Indica que el componente o versión ha sido publicado.

Encargado: Miguel Pérez Fernández
Descripción: Se implementa partiendo de la búsqueda de la dirección de donde fue guardado ese componente cuando se subió al sistema. Esta dirección es pasada al enlace que descarga el componente.

Tabla 74 Tareas de ingeniería HU12

Tarea	
Número de tarea: 3	**Número de HU:** 12
Nombre de la tarea: Descargar versión de componente	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 06/03/2014	**Fecha de terminación:** 06/03/2014
Encargado: Miguel Pérez Fernández	
Descripción: Se implementa partiendo de la obtención del campo url de la tabla c_versiones que es donde se guardan las versiones terminadas. Esta dirección es pasada al enlace que descarga el componente.	

Tabla 75 Tarea de ingeniería HU15

Tarea	
Número de tarea: 1	**Número de HU:** 15
Nombre de la tarea: Seleccionar últimos componentes	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 07/03/2014	**Fecha de terminación:** 07/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* a través de las vistas, muestra los últimos contenidos subidos al sistema. Aplicar el estudio realizado en la implementación de la HU.	

Tabla 76 Tarea de ingeniería HU16

Tarea	
Número de tarea: 1	**Número de HU:** 16
Nombre de la tarea: Crear un tipo de contenido	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3

Fecha de inicio: 10/03/2014	**Fecha de terminación:** 11/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Aplicar el estudio realizado sobre la creación de un tipo de contenido en la implementación de la HU.	

Tabla 77 Tarea de ingeniería HU17

Tarea	
Número de tarea: 1	**Número de HU:** 17
Nombre de la tarea: Modificar documento	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 11/03/2014	**Fecha de terminación:** 12/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite modificar los datos de los documentos que se encuentran en el sistema. Aplicar el estudio realizado en la implementación de la HU.	

Tabla 78 Tarea de ingeniería HU18

Tarea	
Número de tarea: 1	**Número de HU:** 18
Nombre de la tarea: Eliminar documento	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 13/03/2014	**Fecha de terminación:** 13/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Aplicar el estudio realizado sobre la eliminación de un tipo de contenido, adaptándolo al tipo de contenido documento.	

Tabla 79 Tareas de ingeniería HU19

Tarea	
Número de tarea: 1	**Número de HU:** 19
Nombre de la tarea: Mostrar vistas con documento	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3

Fecha de inicio: 14/03/2014	**Fecha de terminación:** 17/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Aplicar el estudio realizado sobre la forma en que el CMS permite mostrar los contenidos en forma de bloques que se asemejen a una galería, para el tipo de contenido documento.	

Tabla 80 Tarea de ingeniería HU20

Tarea	
Número de tarea: 1	**Número de HU:** 20
Nombre de la tarea: Filtrar contenido	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 17/03/2014	**Fecha de terminación:** 17/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Aplicar el estudio realizado sobre el filtrado de contenidos del CMS a partir de un campo previamente seleccionado, para el tipo de contenido documento.	

Tabla 81 Tarea de ingeniería HU21

Tarea	
Número de tarea: 1	**Número de HU:** 21
Nombre de la tarea: Descargar documento	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 18/03/2014	**Fecha de terminación:** 19/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Se implementa partiendo de la búsqueda de la dirección de donde fue guardado ese componente cuando se subió al sistema. Esta dirección se le es pasaba al enlace que descarga el documento.	

Tabla 82 Tareas de ingeniería HU22

Tarea	
Número de tarea: 1	**Número de HU:** 22
Nombre de la tarea: Envío de notificaciones por el núcleo de *Drupal*	

Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 20/03/2014	**Fecha de terminación:** 20/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* envía las notificaciones por correo desde el núcleo. Ver el envío de los correos de cuenta de usuario y el módulo SMTP. Aplicar el estudio realizado en la implementación de la HU.	

Tabla 83 Tareas de ingeniería HU22

Tarea	
Número de tarea: 2	**Número de HU:** 22
Nombre de la tarea: Envío de correo con formato HTML	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 20/03/2014	**Fecha de terminación:** 20/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite el envío de correo con formato HTML.	

Tabla 84 Tareas de ingeniería HU22

Tarea	
Número de tarea: 3	**Número de HU:** 22
Nombre de la tarea: Crear plantillas para las notificaciones	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 20/03/2014	**Fecha de terminación:** 20/03/2014
Encargado: Alejandro Rodríguez de la Cruz\Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo un correo puede cargar una plantilla HTML para darle formato al correo.	

Tabla 85 Tareas de ingeniería HU22

Tarea	
Número de tarea: 4	**Número de HU:** 22

Nombre de la tarea: Enviar notificaciones por correo con formato HTML	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 20/03/2014	**Fecha de terminación:** 20/03/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Se debe implementar las funcionalidades que permitan el envío de correo con formato HTML. El mismo se puede hacer utilizando el *hook_mail()* de la API de *Drupal* o implementar una función que cargue los parámetros necesarios para enviar correo.	

Tabla 86 HU de la iteración 3

Módulo	Historias de usuario (HU)	Tiempo de implementación	
		Estimado	Real
bibliocom	13	2.6	2.4
bibliocom	14	0.4	0.4

Tabla 87 Tareas de ingeniería HU13

Tarea	
Número de tarea: 1	**Número de HU:** 13
Nombre de la tarea: Mostrar directorio del componente	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.6
Fecha de inicio: 24/03/2014	**Fecha de terminación:** 08/04/2014
Encargado: Alejandro Rodríguez de la Cruz\Miguel Pérez Fernández	
Descripción: Se debe implementar una función que escanee el directorio del componente, estructure la vista de carpetas y devuelva al nodo ese resultado para ser renderizado. Cuando se escanee el directorio, se deben guardar los datos de los archivos como son: el nombre y la dirección de cada uno de ellos.	

Tabla 88 Tareas de ingeniería HU13

Tarea	
Número de tarea: 2	**Número de HU:** 13
Nombre de la tarea: Guardar variables del cliente en el servidor	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.6

Fecha de inicio: 24/03/2014	**Fecha de terminación:** 08/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Consultar bibliografía referente a cómo se pueden guardar variables *JavaScript* en variables PHP.	

Tabla 89 Tareas de ingeniería HU13

Tarea	
Número de tarea: 3	**Número de HU:** 13
Nombre de la tarea: Librería para reconocer las sintaxis de los lenguajes	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.6
Fecha de inicio: 24/03/2014	**Fecha de terminación:** 08/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Consultar bibliografía referente a las librerías *js*, con el objetivo de reconocer las sintaxis de los lenguajes que estas utilizan para su adaptación al sistema.	

Tabla 90 Tareas de ingeniería HU13

Tarea	
Número de tarea: 4	**Número de HU:** 13
Nombre de la tarea: Mostrar en tiempo real los cambios	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.6
Fecha de inicio: 24/03/2014	**Fecha de terminación:** 08/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Se debe utilizar la librería *codemirror*, con el lenguaje *html mixed*.	

Tabla 91 Tareas de ingeniería HU13

Tarea	
Número de tarea: 5	**Número de HU:** 13
Nombre de la tarea: Modificar archivos	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.6
Fecha de inicio: 24/03/2014	**Fecha de terminación:** 08/04/2014

Encargado: Alejandro Rodríguez de la Cruz
Descripción: Utilizando Ajax se modifican los cambios hechos en el *textarea* que carga el archivo. Estos cambios se obtienen guardando en una variable *JavaScript* el contenido de ese *textarea* luego con Ajax ese contenido sobrescribe el contenido del fichero pasado por parámetro.

Tabla 92 Tareas de ingeniería HU13

Tarea	
Número de tarea: 6	**Número de HU:** 13
Nombre de la tarea: Terminar componente	
Tipo de tarea: Desarrollo	**Puntos estimados:** 2.6
Fecha de inicio: 24/03/2014	**Fecha de terminación:** 08/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Cuando un componente es terminado se comprime la carpeta del componente y es insertado en la tabla c_versiones la url de donde se comprimió ese directorio. Luego es enviado al editor del sistema un correo para que publique el componente.	

Tabla 93 Tarea de ingeniería HU14

Tarea	
Número de tarea: 1	**Número de HU:** 14
Nombre de la tarea: Mostrar datos del componente	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 09/04/2014	**Fecha de terminación:** 11/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Para mostrar la ficha técnica del componente los datos son obtenidos con la variable del nodo cargado para no hacer consultas a la base de datos. Solo hay que llamar la variable con el atributo del nodo que se desea obtener. De ayuda se puede imprimir la variable del nodo para ver la estructura del arreglo anidado con todos los datos del nodo. Se debe mostrar como ficha técnica el nombre del componente, el usuario que creó el componente, la categoría y la descripción.	

Tabla 94 HU de la iteración 4

Módulo	Historias de usuario (HU)	Tiempo de implementación	
		Estimado	Real
bibliocom	23	0.4	0.4
bibliocom	24	0.4	0.4
bibliocom	25	0.4	0.4
m_compartir	26	0.4	0.4
bibliocom	27	0.4	0.4
bibliocom	28	0.4	0.4
bibliocom	29	0.4	0.4

Tabla 95 Tareas de ingeniería HU23

Tarea	
Número de tarea: 1	**Número de HU:** 23
Nombre de la tarea: Copiar carpetas	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 21/04/2014	**Fecha de terminación:** 23/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Consultar bibliografía referente a cómo el lenguaje PHP permite la copia de carpetas, pues la función por defecto para copiar, solo copia ficheros.	

Tabla 96 Tareas de ingeniería HU23

Tarea	
Número de tarea: 2	**Número de HU:** 23
Nombre de la tarea: Crear directorio	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 21/04/2014	**Fecha de terminación:** 23/04/2014
Encargado: Alejandro Rodríguez de la Cruz\Miguel Pérez Fernández	
Descripción: Consultar bibliografía referente a cómo el lenguaje PHP crea carpetas en el sistema. Aplicar el estudio realizado en la implementación de la HU.	

Tabla 97 Tareas de ingeniería HU23

Tarea	
Número de tarea: 3	**Número de HU:** 23
Nombre de la tarea: Copiar carpetas	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 21/04/2014	**Fecha de terminación:** 23/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Se deben implementar las funcionalidades que permitan copiar el directorio original del fichero a otro fichero nombrado con el id del nodo separado por un guion y después el número de la versión. Se inserta (*db_insert*) en la tabla c_versiones los datos correspondientes a esa versión especificando que el campo estado tenga valor "nop[3]".	

Tabla 98 Tareas de ingeniería HU24

Tarea	
Número de tarea: 1	**Número de HU:** 24
Nombre de la tarea: Envío de variables por la url con *Drupal*	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 23/04/2014	**Fecha de terminación:** 25/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Consultar bibliografía referente a cómo el CMS *Drupal* permite el envío de variables por la url. Aplicar el estudio realizado en la implementación de la HU.	

Tabla 99 Tareas de ingeniería HU24

Tarea	
Número de tarea: 2	**Número de HU:** 24
Nombre de la tarea: Eliminar versión de componente	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 23/04/2014	**Fecha de terminación:** 25/04/2014

[3] Significa que el componente no está publicado.

Encargado: Alejandro Rodríguez de la Cruz
Descripción: Se debe implementar la funcionalidad que permita eliminar las versiones del sistema. Las versiones se pueden eliminar usando los parámetros (*nid* y *num_vers*) por el método *GET* que se usa PHP o a través de los *hook_menu()* de *Drupal*. Cada versión es eliminada con la función *db_delete,* se debe tener en cuenta que al eliminar una versión la misma, sean eliminadas de todas las dependencias que esta tiene (eliminar en cascada). Por último es eliminado el directorio de dicha versión, dejando el contenido comprimido como respaldo en el servidor.

Tabla 100 Tareas de ingeniería HU25

Tarea	
Número de tarea: 1	**Número de HU:** 25
Nombre de la tarea: Descomprimir ficheros	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 28/04/2014	**Fecha de terminación:** 30/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Consultar bibliografía referente a cómo se descomprime fichero con PHP.	

Tabla 101 Tareas de ingeniería HU25

Tarea	
Número de tarea: 2	**Número de HU:** 25
Nombre de la tarea: Subir versiones externas	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 28/04/2014	**Fecha de terminación:** 30/04/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Se debe implementar la funcionalidad que permita descomprimir los ficheros que son subidos al sistema, colocándolos en una carpeta con la nomenclatura de la versión que se está creando. Luego, son insertados en la tabla c_versiones los datos correspondientes a esa versión.	

Tabla 102 Tarea de ingeniería HU26

Tarea	
Número de tarea: 1	**Número de HU:** 26
Nombre de la tarea: Compartir componente	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 02/05/2014	**Fecha de terminación:** 04/05/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Se debe implementar la funcionalidad que permita compartir componentes. Se carga un formulario con los campos título, correo, y descripción. Con estos datos, se capturan los parámetros para enviar el correo con la ayuda del módulo que envía notificaciones por correo con HTML. Internamente se obtienen los datos del usuario y del componente a compartir para estructurar el cuerpo del correo.	

Tabla 103 Tarea de ingeniería HU27

Tarea	
Número de tarea: 1	**Número de HU:** 27
Nombre de la tarea: Añadir componentes a la biblioteca personal	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 04/05/2014	**Fecha de terminación:** 06/05/2014
Encargado: Alejandro Rodríguez de la Cruz	
Descripción: Se debe implementar la funcionalidad que permita añadir un componente a la biblioteca personal del usuario. Se obtiene el meta (Metadato del componente. Contiene la información de si el componente es una versión o un componente original, en caso de ser una versión, se obtiene el valor de id del nodo y el número de la versión separado por un guion, de lo contrario se obtiene solamente el valor del id del nodo). Se crea una tabla en la base de datos llamada c_bpersonal. Para añadir un componente se inserta (*db_insert*) en la tabla los campos meta y *uid*.	

Tabla 104 Tarea de ingeniería HU28

Tarea	
Número de tarea: 1	**Número de HU:** 28
Nombre de la tarea: Eliminar componente de la biblioteca personal	

Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 06/05/2014	**Fecha de terminación:** 08/05/2014
Encargado: Alejandro Rodríguez de la Cruz\Miguel Pérez Fernández	
Descripción: Se debe implementar la funcionalidad que permita eliminar un componente de la biblioteca personal del usuario. Se envía por el método *GET* el valor del meta. Se elimina de la tabla c_bpersonal (*db_delete*) aquellos componentes que cumplen con los parámetros enviados para el usuario que ejecutó esa función.	

Tabla 105 Tarea de ingeniería HU29

Tarea	
Número de tarea: 1	**Número de HU:** 29
Nombre de la tarea: Mostrar biblioteca personal	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 09/05/2014	**Fecha de terminación:** 11/05/2014
Encargado: Alejandro Rodríguez de la Cruz\Miguel Pérez Fernández	
Descripción: Se realiza una consulta a la tabla c_bpersonal cogiendo todos los datos. Se renderizan los datos para imprimirlo con una plantilla diferente a la del tema base.	

ANEXO 5: Casos de prueba de aceptación

Tabla 106 Caso de prueba aceptación HU3

Caso de prueba aceptación	
Código: HU3_P1	**Historia de usuario:** 3
Nombre: Eliminar cuenta de usuario	
Descripción: Prueba para la funcionalidad que permite al administrador eliminar la cuenta de usuario.	
Condiciones de ejecución: El usuario debe estar registrado como administrador. La cuenta a eliminar debe existir.	
Entrada/Pasos de ejecución: El usuario accede al menú de administración *Personas*, selecciona la cuenta de usuario a eliminar, selecciona la opción *Cancelar las cuentas de usuarios seleccionadas* del enunciado *Actualizar opciones,* pulsa el botón *Actualizar.* Selecciona la opción *Eliminar la cuenta y atribuir todo su contenido al usuario Anónimo.* Finaliza dando clic en *Cancelar cuenta.*	
Resultado esperado: Se elimina la cuenta y se muestra el mensaje *Se ha eliminado "nombre del usuario"*.	
Evaluación de la prueba: Satisfactoria	

Tabla 107 Caso de prueba aceptación HU4

Caso de prueba aceptación	
Código: HU4_P1	**Historia de usuario:** 4
Nombre: Autenticar usuario.	
Descripción: Prueba para la funcionalidad que permite de autenticarse en el sistema.	
Condiciones de ejecución: El usuario debe estar anónimo y tener una cuenta creada en el sistema.	
Entrada/Pasos de ejecución: El usuario accede a la opción *Acceder,* aparece un formulario con los campos Nombre de usuario y Contraseña. Llena los campos con sus datos personales y finaliza pulsando el botón *Iniciar sesión*.	

Resultado esperado: Se le muestra el mensaje *Se ha registrado correctamente* y entra en el sistema.
Evaluación de la prueba: Satisfactoria

Tabla 108 Caso de prueba aceptación HU5

Caso de prueba aceptación	
Código: HU5_P1	**Historia de usuario:** 5
Nombre: Agregar componente.	
Descripción: Prueba para la funcionalidad que permite agregar un componente en el sistema.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Entrada/Pasos de ejecución: El usuario accede al menú *Componentes/Agregar componentes,* aparece un formulario con los campos Nombre, Imagen del componente, Autor, Categoría y Componente. El campo Componente solo acepta formato de compresión zip. Luego de llenarse los datos se da clic en *Guardar*.	
Resultado esperado: Se le envía una notificación al Editor para que publique el componente. Es mostrado el mensaje *Se ha creado "nombre del componente"*.	
Evaluación de la prueba: Satisfactoria	

Tabla 109 Caso de prueba aceptación HU6

Caso de prueba aceptación	
Código: HU6_P1	**Historia de usuario:** 6
Nombre: Modificar componente.	
Descripción: Prueba para la funcionalidad que permite modificar un componente en el sistema.	
Condiciones de ejecución: El usuario debe estar registrado como administrador o editor en el sistema.	

Entrada/Pasos de ejecución: El usuario accede al menú de administración *Contenido,* da clic en Editar sobre el componente que desea modificar. Aparece un formulario lleno con los campos Nombre, Imagen del componente, Autor, Categoría y Componente. El campo Componente solo acepta formato de compresión zip. Luego se modifican los datos del componente damos clic en *Guardar*.
Resultado esperado: Se redirecciona la página hacia el listado de contenido y aparece el contenido modificado, señalado con la palabra *actualizado* en rojo.
Evaluación de la prueba: Satisfactoria

Tabla 110 Caso de prueba aceptación HU7

Caso de prueba aceptación	
Código: HU7_P1	**Historia de usuario:** 7
Nombre: Eliminar componente.	
Descripción: Prueba para la funcionalidad que permite eliminar un componente en el sistema.	
Condiciones de ejecución: El usuario debe estar registrado como administrador en el sistema.	
Entrada/Pasos de ejecución: El usuario accede al menú de administración *Contenido*, selecciona el componente a eliminar, da clic en eliminar, se pregunta ¿Seguro que quiere eliminar "componente"?, para finalizar pulsa el botón *Eliminar*.	
Resultado esperado: Se redirecciona la página hacia el listado de componentes y se muestra el mensaje *Se ha eliminado "nombre componente".*	
Evaluación de la prueba: Satisfactoria	

Tabla 111 Caso de prueba aceptación HU8

Caso de prueba aceptación	
Código: HU8_P1	**Historia de usuario:** 8
Nombre: Mostrar componente.	

Descripción: Prueba para la funcionalidad que permite visualizar los componentes en el sistema.
Condiciones de ejecución:
Entrada/Pasos de ejecución: El usuario accede al menú *Componentes/Ver componentes*. Se visualizan los componentes en el sistema. Selecciona el componente que desea ver.
Resultado esperado: Se muestra el componente
Evaluación de la prueba: Satisfactoria

Tabla 112 Caso de prueba aceptación HU9

Caso de prueba aceptación	
Código: HU9_P1	**Historia de usuario:** 9
Nombre: Filtrar componente por categoría.	
Descripción: Prueba para la funcionalidad que permite buscar los componentes a través de filtros.	
Condiciones de ejecución:	
Entrada/Pasos de ejecución: El usuario accede al menú *Componentes/Ver componentes*. Se visualizan los componentes en el sistema. Aparece un listado de opciones llamado *Categoría,* se despliegan las opciones *Galería, Reproductor, Scroll, Menú, Buscador y Otros.* Selecciona la opción por la que se va a filtrar.	
Resultado esperado: Se muestran solo los componentes que pertenecen a esa categoría.	
Evaluación de la prueba: Satisfactoria	

Tabla 113 Caso de prueba aceptación HU10

Caso de prueba aceptación	
Código: HU10_P1	**Historia de usuario:** 10
Nombre: Publicar componente.	
Descripción: Prueba para la funcionalidad que permite publicar un componente en el sistema.	

Condiciones de ejecución: El usuario debe estar registrado como administrador o editor en el sistema.
Entrada/Pasos de ejecución: El usuario accede al menú de administración *Contenido,* da clic en editar sobre el componente que desea modificar. Aparece un formulario con los campos del componente, debajo aparece un menú vertical. Selecciona la pestaña *Opciones de publicación,* luego se marca donde dice *Publicado*. Pulsamos el botón *Guardar* para finalizar.
Resultado esperado: El componente se muestra en *Últimos componentes* o en *Ver componentes*.
Evaluación de la prueba: Satisfactoria

Tabla 114 Caso de prueba aceptación HU11

Caso de prueba aceptación	
Código: HU11_P1	**Historia de usuario:** 11
Nombre: Buscar componentes y documentación.	
Descripción: Prueba para las funcionalidades que permiten buscar componentes y documentos en el sistema.	
Condiciones de ejecución: El componente debe existir en el sistema, el sitio se indexa cada 3 horas por lo que los componentes creados en ese período no se visualiza.	
Entrada/Pasos de ejecución: En la pantalla de inicio se muestra un buscador, introducimos la palabra a buscar y pulsamos el botón *Buscar*.	
Resultado esperado: El sistema muestra un listado con los resultados de la búsqueda.	
Evaluación de la prueba: Satisfactoria	

Tabla 115 Caso de prueba aceptación HU12

Caso de prueba aceptación	
Código: HU12_P1	**Historia de usuario:** 12
Nombre: Descargar componentes.	
Descripción: Prueba para la funcionalidad que permite descargar componente o una versión en el sistema.	

Condiciones de ejecución: Estar visualizando el componente.
Entrada/Pasos de ejecución: Pulsamos el botón *Descargar*. El sistema pregunta al usuario si lo desea abrir o guardar. Cuando le da clic a *Guardar* finaliza la HU12.
Resultado esperado: El componente es descargado.
Evaluación de la prueba: Satisfactoria

Tabla 116 Caso de prueba aceptación HU13

Caso de prueba aceptación	
Código: HU13_P1	**Historia de usuario:** 13
Nombre: Personalizar componente.	
Descripción: Prueba para la funcionalidad que permite descargar componente y versiones en el sistema.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Entrada/Pasos de ejecución: El usuario accede al menú *Componentes/Ver componentes*. Selecciona el componente que va a personalizar. Crea una nueva versión en el sistema, pulsa la pestaña Editor, donde puede modificar todos los archivos del componente y ver en tiempo real los cambios que realiza. Puede guardar los cambios pulsando *Guardar,* una vez que haya terminado de editar el componente pulsa el botón *Terminado*.	
Resultado esperado: El componente es guardado como una nueva versión.	
Evaluación de la prueba: Satisfactoria	

Tabla 117 Caso de prueba aceptación HU14

Caso de prueba aceptación	
Código: HU14_P1	**Historia de usuario:** 14
Nombre: Mostrar ficha técnica del componente.	
Descripción: Prueba para la funcionalidad que permite ver la ficha técnica de un componente del sistema.	

Condiciones de ejecución:
Entrada/Pasos de ejecución: El usuario accede al menú *Componentes/Ver componentes*. Selecciona un componente, inmediatamente puede ver su ficha técnica al pulsar el botón "*Ficha Técnica*". Se muestra la ficha del componente.
Resultado esperado: Se muestra la ficha técnica del componente seleccionado.
Evaluación de la prueba: Satisfactoria

Tabla 118 Caso de prueba aceptación HU15

Caso de prueba aceptación	
Código: HU15_P1	**Historia de usuario:** 15
Nombre: Mostrar últimos componentes.	
Descripción: Prueba para la funcionalidad que permite mostrar los últimos componentes en el sistema.	
Condiciones de ejecución: Estar en la pantalla de inicio.	
Entrada/Pasos de ejecución: Visualiza en la parte inferior del sistema un listado con los 3 últimos componentes subidos al sistema. Al pulsar sobre uno de ellos se visualiza el componente seleccionado.	
Resultado esperado: Se visualizan los componentes.	
Evaluación de la prueba: Satisfactoria	

Tabla 119 Caso de prueba aceptación HU16

Caso de prueba aceptación	
Código: HU16_P1	**Historia de usuario:** 16
Nombre: Agregar documento.	
Descripción: Prueba para la funcionalidad que permite agregar un documento en el sistema.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	

Entrada/Pasos de ejecución: El usuario accede al menú *Documentos/Agregar documentos,* aparece un formulario con los campos Título, Portada del documento, Autor del libro, Clasificación y Documento. El campo Documento solo acepta formato de compresión zip. Luego de llenarse los datos damos clic en *Guardar*. Todos los campos son obligatorios si no se llenan muestra el mensaje *El campo "nombre campo" es obligatorio.*
Resultado esperado: Es mostrado el mensaje *Se ha creado "nombre del documento"*.
Evaluación de la prueba: Satisfactoria

Tabla 120 Caso de prueba aceptación HU17

Caso de prueba aceptación	
Código: HU17_P1	**Historia de usuario:** 17
Nombre: Modificar documento.	
Descripción: Prueba para la funcionalidad que permite modificar un documento en el sistema.	
Condiciones de ejecución: El usuario debe estar registrado como administrador.	
Entrada/Pasos de ejecución: El usuario accede al menú de administración *Contenido,* da clic en editar sobre el documento que desea modificar. Aparece un formulario con los campos Título, Portada del documento, Autor del libro, Clasificación y Documento. El campo Documento solo acepta formato de compresión zip. Luego se modifican los datos del documento damos clic en *Guardar*.	
Resultado esperado: Se redirecciona la página hacia el listado de contenidos y aparece el contenido modificado, señalado con la palabra *actualizado* en rojo.	
Evaluación de la prueba: Satisfactoria	

Tabla 121 Caso de prueba aceptación HU18

Caso de prueba aceptación	
Código: HU18_P1	**Historia de usuario:** 18
Nombre: Eliminar documento.	

Descripción: Prueba para la funcionalidad que permite eliminar un documento en el sistema.
Condiciones de ejecución: El usuario debe estar registrado como administrador en el sistema.
Entrada/Pasos de ejecución: El usuario accede al menú de administración *Contenido*, selecciona el documento a eliminar, da clic en *Eliminar*, el sistema pregunta ¿Seguro que quiere eliminar "documento"?, para finalizar pulsa el botón *Eliminar*.
Resultado esperado: Se redirecciona la página hacia el listado de documentos y se muestra el mensaje *Se ha eliminado "nombre del documento".*
Evaluación de la prueba: Satisfactoria

Tabla 122 Caso de prueba aceptación HU19

Caso de prueba aceptación	
Código: HU19_P1	**Historia de usuario:** 19
Nombre: Mostrar documento.	
Descripción: Prueba para la funcionalidad que permite visualizar los documentos en el sistema.	
Condiciones de ejecución:	
Entrada/Pasos de ejecución: El usuario accede al menú *Documentos/Ver documentos*. Se visualizan los documentos en el sistema. Selecciona el documento que desea ver.	
Resultado esperado: Se muestra el documento.	
Evaluación de la prueba: Satisfactoria	

Tabla 123 Caso de prueba aceptación HU20

Caso de prueba aceptación	
Código: HU20_P1	**Historia de usuario:** 20
Nombre: Filtrar documento por tipo.	

Descripción: Prueba para la funcionalidad que permite buscar un documento a través de filtros.
Condiciones de ejecución:
Entrada/Pasos de ejecución: El usuario accede al menú *Documentos/Ver documentos*. Se visualizan los documentos en el sistema. Aparece un listado de opciones llamado *Tipo,* se despliegan las opciones *HTML5, CSS3, Flash y Otros.* Selecciona la opción por la que se va a filtrar.
Resultado esperado: Se muestran solo los documentos que pertenecen a ese tipo.
Evaluación de la prueba: Satisfactoria

Tabla 124 Caso de prueba aceptación HU21

Caso de prueba aceptación	
Código: HU21_P1	**Historia de usuario:** 21
Nombre: Descargar documento.	
Descripción: Prueba para la funcionalidad que permite descargar documento.	
Condiciones de ejecución: Estar visualizando el documento.	
Entrada/Pasos de ejecución: Pulsamos sobre el enlace que dice *Descargar*.	
Resultado esperado: El documento es descargado.	
Evaluación de la prueba: Satisfactoria	

Tabla 125 Caso de prueba aceptación HU22

Caso de prueba aceptación	
Código: HU22_P1	**Historia de usuario:** 22
Nombre: Enviar notificaciones por correo.	
Descripción: Prueba para las funcionalidades que envían notificaciones por correo.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Entrada/Pasos de ejecución: Probar que las HU 1, HU 5, envían notificaciones por correo.	

Resultado esperado: Correo enviado al usuario.
Evaluación de la prueba: Satisfactoria

Tabla 126 Caso de prueba aceptación HU23

Caso de prueba aceptación	
Código: HU23_P1	**Historia de usuario:** 23
Nombre: Crear versiones.	
Descripción: Prueba para la funcionalidad que permite crear versiones de componentes en el sistema.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Entrada/Pasos de ejecución: El usuario debe acceder al componente que vaya a editar, pulsa el botón *Crear versión*. Hasta que no pulse el botón *Terminado* el componente no es registrado en el sistema.	
Resultado esperado: La dirección del navegador cambia al de la versión creada. Se muestra el mensaje *Versión "número de la versión" creada correctamente.*	
Evaluación de la prueba: Satisfactoria	

Tabla 127 Caso de prueba aceptación HU24

Caso de prueba aceptación	
Código: HU24_P1	**Historia de usuario:** 24
Nombre: Eliminar versiones.	
Descripción: Prueba para la funcionalidad que permite eliminar versiones de componentes en el sistema.	
Condiciones de ejecución: El usuario debe estar registrado como administrador en el sistema.	
Entrada/Pasos de ejecución: El usuario debe acceder al componente que vaya a eliminar, pulsa el botón *Eliminar versión*.	
Resultado esperado: Se muestra el mensaje *Versión "número de la versión" eliminada correctamente.*	
Evaluación de la prueba: Satisfactoria	

Tabla 128 Caso de prueba aceptación HU25

Caso de prueba aceptación	
Código: HU25_P1	**Historia de usuario:** 25
Nombre: Subir versiones.	
Descripción: Prueba para la funcionalidad que permite subir versiones de componentes que no son editadas en el sistema.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Entrada/Pasos de ejecución: El usuario debe acceder al componente que vaya a crearle la nueva versión, pulsa el botón *Subir versión*. Aparece un formulario que ofrece al usuario subir un archivo en formato zip.	
Resultado esperado: Al recargar el sistema se muestra la versión creada.	
Evaluación de la prueba: Satisfactoria	

Tabla 129 Caso de prueba aceptación HU26

Caso de prueba aceptación	
Código: HU27_P1	**Historia de usuario:** 26
Nombre: Compartir componentes.	
Descripción: Prueba para la funcionalidad que permite compartir componentes con otros usuarios.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Entrada/Pasos de ejecución: Pulsa el botón *Compartir*. Aparece un formulario con los campos Título, Correo del usuario y descripción. Cuando se llenan los campos, da clic en el botón *Guardar* y es compartido el componente con el destinatario.	
Resultado esperado: Se muestra una notificación que dice *Componente compartido con "correo del usuario".* Correo enviado al destinatario.	
Evaluación de la prueba: Satisfactoria	

Tabla 130 Caso de prueba aceptación HU27

Caso de prueba aceptación	
Código: HU28_P1	**Historia de usuario:** 27
Nombre: Añadir componente a la biblioteca personal.	

Descripción: Prueba para la funcionalidad que permite añadir componente a la biblioteca personal del usuario.
Condiciones de ejecución: El usuario debe estar registrado en el sistema.
Entrada/Pasos de ejecución: Abrir el componente que desea añadir, pulsa el botón *Añadir favorito.*
Resultado esperado: Se muestra un mensaje que dice *Se agregó el componente a su biblioteca personal.*
Evaluación de la prueba: Satisfactoria

Tabla 131 Caso de prueba aceptación HU28

Caso de prueba aceptación	
Código: HU29_P1	**Historia de usuario:** 28
Nombre: Eliminar componente de la biblioteca personal.	
Descripción: Prueba para la funcionalidad que permite eliminar un componente a la biblioteca personal del usuario.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	
Entrada/Pasos de ejecución: El usuario accede al menú *Mi biblioteca.* Selecciona el componente que desea eliminar de la biblioteca pulsando el botón *x*. Se le muestra un mensaje de confirmación *Desea eliminarlo realmente*. Pulsa el botón *Aceptar* y finaliza la HU.	
Resultado esperado: Se muestra un mensaje que dice *Componente eliminado de su biblioteca personal.*	
Evaluación de la prueba: Satisfactoria	

Tabla 132 Caso de prueba aceptación HU29

Caso de prueba aceptación	
Código: HU30_P1	**Historia de usuario:** 29
Nombre: Mostrar biblioteca personal.	
Descripción: Prueba para la funcionalidad que permite añadir componente a la biblioteca personal del usuario.	
Condiciones de ejecución: El usuario debe estar registrado en el sistema.	

Entrada/Pasos de ejecución: El usuario accede al menú *Mi biblioteca.* Se visualizan todos los componentes y versiones que ese usuario ha agregado a su biblioteca personal. Se muestran los datos de los componentes así como las opciones de ver y eliminar. Al final de la biblioteca aparecen los datos estadísticos del usuario.
Resultado esperado: Se cargan los componentes que el usuario ha agregado a la biblioteca personal así como los datos estadísticos.
Evaluación de la prueba: Satisfactoria