

Universidad de las Ciencias Informáticas.



Directorio de Software en línea

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

Autores: Yanirys Ortíz Lima
Odaimy Alvarez Acosta

Tutor: Ing. Yoandy Pérez Villazón

Co-Tutor: Ing. Alexander Martínez Fajardo

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de ____ del año 2011.

Odaimy Alvarez Acosta

Firma del Autor

Yanirys Ortíz Lima

Firma del Autor

Ing. Yoandy Pérez Villazón

Firma del Tutor

Ing. Alexander Martínez Fajardo

Firma del Co-Tutor

Agradecimientos

Hacer realidad este sueño no hubiese sido posible sin esta oportunidad que nos brinda la Revolución y el apoyo de muchas personas durante estos 5 años. Queremos agradecer a todos los que de una forma u otra a todos los que han colaborado con nuestra formación, profesores, amigos y familiares. Al tribunal por guiarnos en cada corte, a los tutores Yoandy y Alexander, especialmente a Ale y Javier por estar siempre dispuestos a ayudarnos. A todos ellos muchas gracias.

En especial Agradecemos:

A mi mamá por ser maravillosa en mi vida, la persona que siempre me brinda su amor y cariño en todo momento y siempre está pensando en mi.

A mi papá por ser ejemplo en sacrificio, además de ser siempre tan comprensivo y cariñoso conmigo.

A mi hermana por ser mi guía y siempre ser mi Tata linda, sabes que yo te adoro.

A mi novio Anier por estar a mi lado estos últimos nueve años por tu apoyo cuando pensé que no podía, por darme siempre tu cariño, tu comprensión y amor. Te amo mi Tati.

A esa personita maravillosa que es mi sobrina, que es la luz de cada día de mi familia.

Yanirys Ortíz Lima

Dedicatoria

A mi madre y mi padre por brindarme su ejemplo, amor y comprensión que son personas extraordinarias, que tanto se han esforzado porque yo termine mis estudios.

A mi hermana que aunque en estos momentos no esté a mi lado se que esté pensado en mi, por ser tan especial.

A mi novio que siempre me ha apoyado y a pasado conmigo buenos y malos momentos.

Yanirys Ortíz Lima

Resumen

El presente trabajo de diploma se centra en la necesidad de desarrollar un sistema que permita la obtención eficiente de las alternativas compatibles con software libre, posterior al levantamiento de información que se realiza en el proceso de migración a Software Libre Código Abierto en nuestro país.

Para darle cumplimiento a la presente investigación se realizó un análisis de las herramientas de búsqueda y recuperación de información para la web, así como de los directorios de software existentes. Se realizó un estudio de las tecnologías, herramientas y lenguajes utilizados en la construcción del sistema, definiéndose el uso del lenguaje PHP, el framework Symfony y JQuery. Además del Sistema Gestor de Bases de Datos PostgreSQL como soporte al desarrollo de la solución propuesta. Se identificaron y documentaron todas las funcionalidades que el sistema debía cumplir y el proceso de desarrollo de software estuvo guiado por la metodología ágil SXP.

Se logró un sistema que cumple con los requisitos establecidos por el cliente, brindando un conjunto de servicios web para la integración a la plataforma de migración a código abierto. Además de automatizar la tarea de sustitución de las aplicaciones privativas por herramientas libres alternativas dentro del proceso de migración a Software Libre y Código Abierto definido y aplicado en nuestro país por el proyecto SIMAYS.

Palabras Claves: Código Abierto, metodología ágil, servicios web, Software Libre

Índice

Declaración de Autoría.....	2
Agradecimientos.....	3
Dedicatoria	4
Resumen.....	5
Introducción.....	1
Capítulo 1. Fundamentación teórica.....	5
1.1 Conceptos asociados al dominio del problema.....	5
1.2 Herramientas web para la búsqueda y recuperación de información	6
1.2.1 Tipos de herramientas de búsqueda y recuperación de información.....	6
1.2.2 Criterios para la selección de la herramienta de búsqueda y recuperación.....	7
1.3 Análisis de los directorios de software existentes.....	7
1.3.1 Directorios de software existentes.....	7
1.3.2 Valoración del estudio realizado	11
1.4 Metodología de desarrollo	11
1.4.1 Metodologías ágiles	12
1.5 Lenguajes, tecnologías y herramientas de desarrollo utilizadas.....	13
1.5.1 Lenguajes.....	13
1.5.2 Tecnologías.....	15
1.5.2.1 Framework para el lenguaje PHP	16
1.5.2.2 Sistema gestor de base de datos	17
1.5.2.3 Servidor de Aplicaciones Web	18
1.5.2.4 Plugins y librerías utilizadas.....	19
1.5.2.5 Servicios Web	20
1.5.3 Herramientas de soporte al proceso de desarrollo de software	23
1.6 Conclusiones parciales	24
Capítulo 2. Definición del directorio de software.....	25
2.1 Lista de Reserva del Producto (LRP).....	25
2.2 Lista de Historias de Usuario	29
2.2.1 Historia de Usuario Gestionar Versiones de Software.....	29
2.2.2 Historia de Usuario Gestionar Software.....	32
2.2.3 Historia de Usuario Gestionar imágenes.....	33
2.2.4 Historia de Usuario Gestionar alternativas a las versiones de software.....	33
2.2.5 Historia de Usuario Gestionar Comentarios.....	37
2.2.6 Historia de Usuario Gestionar listados de software.....	37

2.2.7 Historia de Usuario Gestionar repositorios.....	38
2.2.8 Historia de Usuario Gestionar Categorías de Software.....	40
2.2.9 Historia de Usuario Gestionar Licencias de Software	42
2.2.10 Historia de Usuario Gestionar Plataformas.....	44
2.2.11 Historia de Usuario Enviar e-mail.....	46
2.2.12 Historia de Usuario Listar los tipos de alternativas.....	46
2.2.13 Historia de Usuario Evaluar software y alternativas de software.....	47
2.2.14 Historia de Usuario Listar idiomas.....	48
2.2.15 Historia de Usuario Guardar descargas.....	48
2.2.16 Historia de Usuario Adicionar software a un listado.....	49
2.2.17 Historia de Usuario Importar paquetes de un repositorio	49
2.2.18 Historia de Usuario Mostrar paquetes de un repositorio	50
2.2.19 Historia de Usuario Eliminar paquetes de un repositorio	51
2.3 Plan de Releases	52
2.4 Conclusiones parciales	53
Capítulo 3. Desarrollo del directorio de software e integración con la plataforma de migración ...	54
3.1 Desarrollo del directorio de software.....	54
3.1.1 Arquitectura del directorio de software.....	54
3.1.1.1 Modelo de diseño.....	54
3.1.1.2 Diagrama de componentes.....	57
3.1.1.3 Esquema de la base de datos utilizada en la solución propuesta.....	58
3.1.2 Estándar de codificación utilizado.....	59
3.2 Desarrollo del módulo de interfaz de usuario del directorio de software.....	59
3.2.1 Requisitos para la integración.....	59
3.2.2 Modelo de diseño.....	59
3.3 Conclusiones parciales.....	62
Capítulo 4. Análisis de los resultados.....	63
4.1 Pruebas funcionales.....	63
4.1.1 Pruebas funcionales realizadas al sistema.....	64
4.2 Impacto y aporte de la solución propuesta.....	64
4.3 Conclusiones parciales.....	64
Conclusiones.....	65
Recomendaciones.....	66
Referencias bibliográficas.....	67
Bibliografía.....	69
Glosario de Términos.....	71

Introducción

La necesidad de información nace con los primeros hombres que habitaron la Tierra. En los primeros tiempos era registrada en diferentes medios como la piedra, la arcilla, la madera y otros, pero estos medios tenían límites para reproducirla y difundirla, que sólo fue eliminada con la llegada de la imprenta. Es entonces cuando surge la biblioteca como la primera institución ideada por el hombre con el fin de preservar la información escrita, almacenada en determinados soportes.

Años más tarde, con la aparición de la primera computadora y el surgimiento de la informática, surge una nueva era en el procesamiento automático de la información. Debido a su capacidad para almacenar, analizar y recuperar eficientemente gran cantidad de datos, el computador se convierte en algo esencial en la gestión de la información en todos los entornos del desarrollo de la humanidad.

Con el surgimiento de Internet, la red de redes, que revolucionó el mundo de la informática, millones de personas tienen acceso a una cantidad extensa y diversa de información en línea. Pero la existencia de tal cantidad de información, muchas veces desorganizada, el carácter cambiante de la propia red y que no toda la información tiene la misma calidad o valor, provoca que el proceso de búsqueda de la misma se convierta en algo difícil y tedioso.

Es por ello, que han diseñado herramientas que se desenvuelven en Internet; como directorios de materias y motores de búsquedas, cuyo objetivo es organizar, localizar y facilitar el acceso a la información y recursos de la red. Los directorios de materias son listas organizadas que nos permiten acceder a la información de forma estructurada y jerárquica. Se clasifican en categorías y el usuario enlaza de lo más general a lo más específico. Mientras que los motores de búsquedas son sistemas que permite buscar información en Internet a partir de palabras claves que identifican el contenido de la pregunta realizada.

Actualmente el software libre ha tomado auge en los países pobres y en vías de desarrollo como alternativa al software privativo, ya que este ha evolucionado el modelo de desarrollo económico y se ha consolidado en muchas partes del mundo, teniendo un gran respaldo por parte de académicos, organizaciones educativas, empresas, desarrolladores y usuarios de software. El software libre presenta una serie de ventajas que promueven su uso, como son su seguridad, defiende fielmente los estándares abiertos, libertad para utilizarse en cualquier sitio, favorece la cultura de la colaboración, aumenta la capacidad tecnológica y reduce la dependencia de proveedores. Es por ello que la migración a software libre se ha convertido en un tema que ha

trascendido del aspecto técnico para llegar a ser un tema estratégico y político en muchos países como Cuba. La migración a software libre no es una moda, sino es, un cambio tecnológico y social donde las personas deben estar convencidas de las ventajas del uso de las nuevas tecnologías y apoyen este proceso.

En nuestro país la migración a software libre y código abierto se lleva a cabo paulatinamente para de esta manera lograr la soberanía tecnológica y la independencia de los proveedores. Este proceso es asumido por el proyecto de Servicios Integrales de Migración, Asesoría y Soporte (SIMAYS) de la Universidad de las Ciencias Informáticas (UCI), perteneciente al Grupo Técnico Nacional para la migración a software libre y código abierto del país, está soportado sobre flujos de trabajo, que abarcan un conjunto limitado de actividades que conducen al objetivo final del proceso de migración en las entidades involucradas. Una de las actividades dentro de este proceso es la búsqueda de alternativas compatibles con software libre a las aplicaciones privativas. Esta búsqueda se realiza muchas veces en Internet, cuando por lo general pueden estar en los repositorios locales de la Universidad de las Ciencias Informáticas, implicando un costo en tiempo en la definición de las propuestas libres. También, al realizar manualmente la búsqueda de las alternativas, empleando como base el levantamiento de información realizado en las empresas, debe ser analizado a varios niveles el hecho de que una aplicación privativa tenga 2 o 3 aplicaciones libres que constituyan equivalentes de la misma, por lo que la búsqueda en Internet es constante y puede incidir el error del análisis humano, además de demorar el proceso.

De acuerdo a la situación descrita anteriormente se identifica como **problema científico**: ¿Cómo hacer eficiente la búsqueda de las alternativas compatibles con software libre en el proceso de migración a Software Libre y Código Abierto?

Estableciéndose como **objeto de estudio** los procesos de búsqueda y recuperación de información y como **campo de acción** el directorio de software, presentándose como **objetivo general** desarrollar un directorio de software que permita la obtención de las alternativas compatibles con software libre en el proceso de migración a Software Libre y Código Abierto. Definiendo como **idea a defender** la siguiente: con la creación de un directorio de software, para la integración a la plataforma de migración a código abierto, se hará eficiente la búsqueda de las alternativas libres durante el proceso de migración a Software Libre y Código Abierto, llevado a cabo en nuestro país.

Para lograr el objetivo planteado anteriormente se determinaron varios **objetivos específicos**:

- Analizar las herramientas de búsqueda y recuperación de información.
- Seleccionar las herramientas, lenguajes, tecnologías y la metodología a emplear en el

desarrollo del sistema propuesto.

- Definir las funcionalidades del directorio de software.
- Diseñar e implementar las funcionalidades del sistema mediante servicios web.
- Integrar el sistema desarrollado a la plataforma de migración.
- Probar la solución propuesta.

Para darle cumplimiento a los objetivos trazados se han propuesto las siguientes **tareas de investigación**:

- Revisión de la bibliografía especializada en las herramientas de búsqueda y recuperación de información para la web.
- Análisis de las herramientas, lenguajes, tecnologías y la metodología a utilizar en el desarrollo del sistema propuesto.
- Entrevistas al cliente para definir los requisitos del sistema.
- Definición de una arquitectura para el sistema.
- Implementación del directorio de software.
- Implementación del módulo interfaz de usuario del directorio de software.
- Diseño y ejecución de pruebas funcionales a la aplicación para asegurar la calidad del producto.

Los métodos científicos que serán utilizados en esta investigación serán los teóricos; los cuales permiten revelar las relaciones esenciales del objeto de investigación, no observables directamente. Específicamente haremos uso del analítico-sintético y del inductivo-deductivo. El método analítico-sintético se enfoca en determinar la esencia de los fenómenos, los rasgos que los caracterizan y los distinguen. Este método permitió luego del estudio de la bibliografía existente extraer los elementos fundamentales relacionados con la búsqueda y recuperación de información.

El método inductivo-deductivo permite llegar a proposiciones generales a partir de hechos aislados que confirman la teoría, o a partir de estas teorías arribar a conclusiones sobre casos particulares que se verifican en la práctica. Este método permitió inferir a partir de los directorios de software existentes la forma en que ellos llevan a cabo la búsqueda y recuperación de información.

También se hace uso de la entrevista como técnica de recopilación de información “La entrevista

se realiza mediante una conversación planificada, con la que además de lo que se investiga, tiene importancia educativa, depende en gran medida del nivel de comunicación entre el investigador y los participantes en la misma” [1].

Se realizó una entrevista al líder del proyecto SIMAYS con el objetivo de comprender las características de la aplicación a desarrollar y profundizar en los detalles de sus principales funcionalidades [Anexo 1].

El documento está estructurado de la siguiente manera:

- **Capítulo 1: Fundamentación Teórica:** Estudio de las herramientas de búsqueda y recuperación de información para la web y del estado actual del arte de los directorios de software, así como sus principales características. Análisis del estudio realizado y conclusiones sobre el sistema que se desarrollará. Argumentación de la Metodología y Tecnologías de desarrollo empleada para el desarrollo del sistema propuesto.
- **Capítulo 2: Definición del directorio de software:** Identificación de las funcionalidades del sistema, así como de los componentes a desarrollar y su planificación.
- **Capítulo 3: Desarrollo del directorio de software e integración con la plataforma de migración:** Descripción de la arquitectura del sistema. Realización del modelo de diseño, estructuración del directorio de software a través del diagrama de componentes, así como el diseño del modelo de datos. La estandarización del código. Creación del módulo de interfaz de usuario del directorio de software, mostrando el modelo de diseño del mismo.
- **Capítulo 4: Análisis de los resultados:** Descripción de los casos de pruebas de aceptación. Análisis del impacto de la solución propuesta.

La presente investigación aportará servicios web para la propuesta de integración de los componentes de la plataforma de migración a código abierto. Unido a esto, mejorará los niveles de eficiencia del proceso de búsqueda de alternativas a las aplicaciones privativas en la migración a software libre y código abierto llevado a cabo en nuestro país.

Capítulo 1. Fundamentación teórica

En el presente capítulo serán analizados los fundamentos de los procesos de búsqueda y recuperación de información, las herramientas web existentes para la búsqueda y recuperación de información; así como algunas soluciones de directorios de software existentes en el mundo. Se muestran además las diferentes herramientas, tecnologías y metodología utilizadas en la implementación del sistema.

1.1 Conceptos asociados al dominio del problema

Recuperación de Información

La recuperación de la información tiene que ver con la representación, almacenamiento, organización y acceso a los elementos de información. “Es el proceso donde se accede a una información previamente almacenada, mediante herramientas informáticas que permiten establecer ecuaciones de búsqueda específicas”[2]. Esta información ha debido ser estructurada previamente a su almacenamiento.

Para esta labor se han creado los sistemas de recuperación de información, que permiten el almacenamiento, procesamiento, recuperación y difusión de la información, sea esta de tipo texto, imágenes, audio, vídeo, entre otros. Se puede recuperar a través de diferentes herramientas: bases de datos, Internet, tesauros, ontologías, mapas, etc.

Para la recuperación de información y documentos en la Web, se han desarrollado diferentes estrategias, como el uso de metadatos o la utilización de lenguajes semánticos basados en XML para indizar documentos web y representar el conocimiento incluido en ellos. Hoy es fácil recuperar la información precisa si se utiliza un lenguaje para representar el contenido semántico de los documentos y soportar las inferencias lógicas. “Los lenguajes de recuperación de información son lenguajes informáticos que constituyen un conjunto de órdenes, operadores y estructuras que, organizados según normas lógicas, permiten la consulta de fuentes y recursos de información electrónica” [3], estos son utilizados para recuperar información de sus almacenes.

Búsqueda de Información

La búsqueda de información es un conjunto de operaciones que tienen por objeto poner al alcance del usuario la información que de respuesta a sus preguntas ocasionales o permanentes. Por el gran volumen de información existente, es de vital importancia la utilización de herramientas que permitan su localización.

1.2 Herramientas web para la búsqueda y recuperación de información

Internet proporciona acceso a una gran cantidad de información sobre cualquier tema, la cual no está clasificada y organizada, como en una biblioteca o una base de datos. Buscar en Internet de manera rápida y eficaz es una tarea difícil. Es por ello que las herramientas de búsqueda y recuperación de información contribuyen a simplificar la búsqueda en la web.

1.2.1 Tipos de herramientas de búsqueda y recuperación de información

Motores de búsquedas

Los motores de búsquedas permiten buscar y recuperar información en Internet a través de la utilización de palabras claves y combinaciones de ellas. Son bases de datos creadas por indexación automática del texto completo de las páginas web y realizada por un programa llamado robot. Este robot lógico, o araña, explora de forma automática los servidores, extrayendo las palabras más significativas de cada página y creando un índice de búsqueda.

Metabuscadores

Los metabuscadores permiten buscar en varias herramientas de búsqueda a la vez y obtener una página con una compilación de los resultados. Es por ello que se usan para aquellas búsquedas en las que es interesante obtener el máximo de recursos disponibles en la red.

Directorios de Materias

Los directorios presentan una selección de recursos web organizados, siguiendo una estructura o clasificación jerárquica de materias que va de categorías más amplias a categorías más específicas. Los directorios se exploran mediante la navegación de una base de datos de documentos web compilados, recogidos y organizados manualmente por expertos. La búsqueda jerárquica sirve al usuario de guía, permitiendo acceder a la información en el contexto temático al que pertenece y en relación a otras áreas temáticas. También presentan un motor de búsqueda interno para localizar directamente recursos de la base de datos.

Guías de Materias

Las guías de materias son una selección de recursos de información relacionados con una materia, campo o disciplina que muestra de forma sistemática enlaces a páginas web relevantes para cada una de estas disciplinas. Estas son compiladas por expertos, especialistas, agencias y asociaciones que permiten su constante crecimiento, revisión y actualización.

Agentes Inteligentes

Los agentes inteligentes permiten localizar la información de forma automática, sólo necesita que se le defina un perfil de búsqueda y donde debe lanzarla (bases de datos, sitios web, etc) y automáticamente va presentando un informe sobre la nueva información que va surgiendo.

1.2.2 Criterios para la selección de la herramienta de búsqueda y recuperación

Se seleccionó un directorio de materia como herramienta de búsqueda y recuperación de información porque es esta la que más se adapta al directorio de software, ya que en los directorios se sistematiza la información de forma manual, o sea son expertos los que organizan toda la información recogida en una base de datos, se encargan de establecer las categorías y de presentar todo el contenido, que en nuestro caso sería responsabilidad de los especialistas en migración. Además los directorios permiten realizar la búsqueda manual mediante la navegación por categorías, así como una búsqueda directa mediante un buscador interno, lo que sería de mucha utilidad para una mayor precisión en la búsqueda de aplicaciones.

1.3 Análisis de los directorios de software existentes

1.3.1 Directorios de software existentes

✓ Free Software Directory¹

El Directorio de Software Libre es un proyecto de la Free Software Foundation (FSF) y las Naciones Unidas para la Educación, la Ciencia y la Cultura (UNESCO). Posee una serie de catálogos de software libre que son útiles, ya que se ejecuta en sistemas operativos libres GNU y sus variantes GNU/Linux. Posee más de 5 000 programas de software libre.

Características:

- La información está organizada por categorías y subcategorías.
- Posee un buscador de aplicaciones libres.
- No brinda alternativas a programas privativos.
- Ofrece información sobre cada aplicación como: la página oficial, licencia, la valoración de los usuarios.
- Permite evaluar las aplicaciones.

1 <http://directory.fsf.org/>

✓ **Directorio de aplicaciones libres de la Oficina de Software Libre de la Universidad de Zaragoza**²

El directorio pertenece a la Oficina de Software Libre de la Universidad de Zaragoza y este genera una lista de programas libres para distintos sistemas operativos (Windows, MACOS X, GNU/Linux) como alternativa a los programas privativos más comunes.

Características:

- No posee buscador.
- Está organizado por categorías y subcategorías.
- Brinda la posibilidad de evaluar las aplicaciones.
- Ofrece información sobre cada software como: licencia, autores, sistemas operativos compatibles, página web oficial, última versión recomendada.
- Muestra una imagen descriptiva de cada aplicación.
- Permite descargar el software.

✓ **Open Source as alternative**³

Open Source as alternative (osalt) está disponible en tres idiomas, el inglés y español entre ellos; es una web bastante intuitiva y muy básica, se centra en proveer de una lista de software privativo y libre, para así poder hacer una comparación y valoración entre ellos. La idea es encontrar el software libre que compite con determinado software privativo, así el usuario tiene la posibilidad de observar las cualidades de cada aplicación.

Características:

- Brinda aplicaciones equivalentes a programas para Windows, Mac y Linux.
- Permite buscar las aplicaciones libres, así como las alternativas a los programas privativos mediante un buscador.
- Está organizado por categorías.
- Sugiere las aplicaciones más populares en cada categoría.
- Posibilita la descarga de programas.
- Muestra información sobre cada aplicación como: sistemas operativos compatibles y página web oficial.
- Se puede sugerir o solicitar las aplicaciones ya sean de código abierto o privativas

2 <http://osluz.unizar.es/aplicaciones/>

3 <http://www.osalt.com/>

mediante un formulario de contacto.

✓ **Alternativas Libres**⁴

Alternativas Libres establece relaciones entre aplicaciones privativas y las alternativas libres que existen. Ha sido desarrollado exclusivamente usando software libre. La mayor parte del sitio ha sido escrito en PHP, utilizando como servidor web Apache. Todo el código ha sido escrito con Vim. El gestor de bases de datos usado es PostgreSQL. Para algunas tareas puntuales se ha utilizado Perl y la biblioteca LWP. Permite syndicar las diez últimas aplicaciones usando RSS una de las aplicaciones del RDF, estándar del W3C para la web semántica. El aspecto del sitio y la disposición se determina mediante una hoja de estilo CSS2.

Características:

- Posee un buscador de aplicaciones libres y otro para la obtención de las alternativas libres a los programas privativos.
- Está organizado por categorías.
- El sitio posee dos modos: edición y usuario. En modo edición se puede contribuir al sitio usando los enlaces de edición de cada página, mientras que el modo usuario es utilizado para consultar la información disponible.
- Brinda información sobre cada aplicación como: descripción, página web oficial, valoración de la aplicación, imagen de la aplicación.
- Permite contribuir al sitio añadiendo aplicaciones libres, aplicaciones privativas o mediante el completamiento de aplicaciones ya disponibles.

✓ **Freshmeat**⁵

Freshmeat mantiene el índice más grande de la web de Unix y de software multiplataforma. Posee miles de aplicaciones liberadas bajo licencia de código abierto, que son catalogados en la base de datos. Se actualiza continuamente con las últimas aplicaciones libres.

Características:

- Está organizado por categorías.
- Muestra información sobre cada aplicación como: descripción, página oficial, imagen de la aplicación, sistemas operativos compatibles, lenguajes de implementación, últimas

4 <http://www.freealts.com/>

5 <http://freshmeat.net/>

versiones, estadísticas gráficas, las dependencias de otras aplicaciones.

- Permite solicitar la propiedad de cada software, así como la realización de comentarios de los mismos.
- Posibilita descargar cada software.
- Posee un buscador para la localización directa de determinada aplicación.
- Ofrece la posibilidad de reportar los problemas relacionados con algún programa.
- Se pueden añadir aplicaciones.

✓ **Open Source Living an archive of the web's best open source software**⁶

El Archivo de Open Source Living es un proyecto que trata de crear una base de datos de aplicaciones bajo licencia libre. Su principal característica está en la colaboración, ya que está abierto a las personas que realicen proposiciones para ir enriqueciendo la lista de aplicaciones.

Características:

- Está organizado por categorías y subcategorías.
- Permite buscar las aplicaciones libres mediante un buscador.
- Refleja información sobre cada software como: licencia, autores, sistemas operativos compatibles, página web oficial, versión, descripción, imagen de la aplicación.
- Ofrece una valoración en cuanto a características, usabilidad e instalación de los programas.
- Brinda la posibilidad de hacer comentarios a las aplicaciones.
- Se pueden añadir programas.

✓ **Todo Programas**⁷

Todo Programas es un directorio de programas gratis que ofrece descargas según el sistema operativo: Windows, Mac, Linux y móviles.

Características:

- Está organizado por categorías y subcategorías.
- Posee un buscador para realizar búsquedas básicas y avanzadas de los programas.
- Sugiere aplicaciones equivalentes a programas para Windows, Mac y Linux.
- Ofrece información sobre cada aplicación como: licencia, autores, versión, descripción,

⁶ <http://osliving.com/>

⁷ <http://www.todoprogramas.com/>

imágenes de la aplicación, tamaño, idioma, fecha de actualización, estadísticas sobre visitas y descargas, valoración.

- Brinda la posibilidad de descargar cada software, realizar comentarios y votaciones.

1.3.2 Valoración del estudio realizado

Luego de concluir el estudio y análisis de algunos de los directorios de software existentes y en concordancia con las características que poseen; se puede concluir que como estos sistemas no están disponibles para su descarga, se tendrán en cuenta para el desarrollo del directorio algunas de sus características tales como:

- Organización por categorías y subcategorías.
- Información sobre cada aplicación como: descripción, versión, licencia, imagen de la aplicación, plataformas, enlace para la descarga, dirección en internet de la página oficial del proyecto de software en cuestión, comentarios de visitantes, idioma, tamaño, total de descargas.
- Búsqueda de aplicaciones libres y de alternativas libres a los programas privativos.
- Descarga de los programas.
- Evaluación de las aplicaciones.

1.4 Metodología de desarrollo

La selección correcta de la metodología de desarrollo de software; así como las herramientas, tecnologías y lenguajes de programación que soportarán al mismo es muy importante; la calidad del proceso y del producto final estarán condicionadas en gran medida por estos dos aspectos (herramientas y metodología).

Metodología de desarrollo de Software

Una metodología de desarrollo de software es un conjunto de pasos, técnicas y procedimientos que deben seguirse para desarrollar software [4].

Las metodologías de desarrollo de software facilitan la planificación, control, seguimiento y evaluación de los proyectos, optimizan la gestión de recursos y garantizan en la medida de lo posible la calidad del producto.

Existen dos vertientes fundamentales de metodologías de desarrollo de software, las cuales están dirigidas a beneficiar la labor de los desarrolladores de software: las metodologías ágiles y las metodologías tradicionales. Las metodologías ágiles proponen mejorar la calidad del producto de

software a través de la comunicación inmediata y directa con el cliente, mientras que las metodologías tradicionales proponen que sea a través del orden y la documentación.

1.4.1 Metodologías ágiles

Son metodologías aplicables a proyectos que necesitan solución rápida y no generan mucha documentación, pero resuelven los problemas de diseño y construcción de una forma bastante ágil respecto a las metodologías tradicionales. Su filosofía se centra en el individuo, la colaboración con el cliente, que éste sea partícipe del proceso de desarrollo del software, y el desarrollo incremental del producto software en sí con iteraciones cortas.

Metodología SXP

Metodología ágil desarrollada en la Universidad de las Ciencias Informáticas, que hace uso de SCRUM para la gestión del trabajo y toma de XP las mejores prácticas que guían el desarrollo del software, como la refactorización y pruebas continuas. Ofrece una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software, para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo, y cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto.

SXP consta de 4 fases:

- **Fase de Planificación-Definición:** Se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto. Cuenta con tres tareas principales: ingeniería de sistemas o de información, planificación del proyecto y análisis de los requisitos.
- **Fase de Desarrollo:** Se implementa un sistema listo para entregar en una serie de iteraciones. Cuenta con tres tareas fundamentales: diseño del software, generación de código y prueba del software.
- **Fase de Entrega:** Se despliega y se pone en marcha el producto software
- **Fase de Mantenimiento:** Se realiza el soporte para el cliente.

SXP es ideal para proyectos de corta duración con requisitos cambiantes o no bien definidos, donde prevalezca la retroalimentación entre el cliente y el equipo de trabajo. El desarrollo con SXP se realiza en iteraciones cortas (sprints) a lo largo de 3 fases, dándole cumplimiento a un grupo de actividades, de las que se generan una serie de artefactos, que documentan el proceso de

desarrollo, obteniendo un release del producto con nuevas funcionalidades [5].

1.5 Lenguajes, tecnologías y herramientas de desarrollo utilizadas

1.5.1 Lenguajes

➤ PHP

PHP (de sus siglas en inglés Hypertext Preprocessor) es un lenguaje de código abierto interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor [6]. Es posiblemente el lenguaje más usado para la programación de sitios web dinámicos, por su sencillez y prestaciones, aún más cuando su versión más reciente la PHP 5 (v5.05) tiene incorporado un grupo de mejoras en cuanto a POO (Programación Orientada a Objetos). Además de estar publicado bajo PHP Licence v3.01 (<http://www.php.net>) y considerada como software libre por la Free Software Foundation.

Características:

- Es un lenguaje multiplataforma.
- Dispone de una conexión propia a todos los sistemas de base de datos, tales como:
- MySQL, PostgreSQL, mSQL, Oracle, dbm, filepro, Hyperwave, Informix, InterBase y Sybase, entre otras.
- Se integra a los servidores web más utilizados, Apache e IIS (Internet Information Server).
- Posee alto rendimiento en la ejecución de los scripts.
- Posee un gran número de funciones predefinidas.

PHP fue creado como un complemento para el lenguaje PERL en 1994 por Rasmus Lerdorf, lo utilizó por primera vez en su propia página Web para monitorizar las visitas que recibía. Por la popularidad de sus comandos en sus páginas, Rasmus se decidió a lanzar la primera versión completa de PHP, conocida como Personal Home Page Tools (Herramientas para Páginas Personales). Poco a poco fueron aumentando las necesidades de gestión de su información personal, por lo que fue agregándole un conjunto de librerías escritas en lenguaje C, capaces de conectar con bases de datos y permitía a los usuarios escribir pequeñas aplicaciones Web dinámicas.

Inicialmente la sintaxis de las variables era muy similar a la de Perl y sólo se podían ejecutar algunas de las funcionalidades básicas de las que cuenta hoy en día; en 1993 surge la primera versión bajo el nombre de PHP 3, incorporando soporte para varias bases de datos, además de

convertirse en un lenguaje escalable, bajo el nombre de PHP, lo que eliminaba todo vestigio de uso personal.

En mayo de 2000, fue liberado PHP 4 con un nuevo motor (Zend) con mejoras en código base, haciéndolo modular y concebido para la ejecución de aplicaciones complejas. Se incluyeron nuevas características como soporte para múltiples servidores Web, manejos de sesiones de usuarios, buffers de salidas y formas más seguras de control de entradas de usuarios [7].

➤ **YML**

YAML (del inglés, *YAML Ain't Markup Language*) es un lenguaje de serialización de datos [8], es un lenguaje muy simple que se usa para describir los datos de una manera similar a XML, pero con una sintaxis más simple. Su uso se extiende a múltiples aplicaciones, principalmente en ficheros de configuración e intercambio ligero de información y como lenguaje para la representación del esquema de base de datos para el framework de ORM (del inglés, *Object Relational Mapping*) Doctrine.

➤ **XML**

XML, del acrónimo en inglés *eXtensible Markup Language* (lenguaje de marcas extensible), es un metalenguaje extensible de etiquetas, que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados. Surge de un lenguaje ideado por IBM, llamado GML (*Generalized Markup Language*), como necesidad de almacenar grandes cantidades de información.

XML es un lenguaje muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos [9] ya que permite la lectura de datos a través de diferentes aplicaciones. Sirve para estructurar, almacenar e intercambiar información.

➤ **JavaScript**

JavaScript es un lenguaje de programación ligero y orientado a objeto, que nos permite interactuar con el navegador de manera dinámica y eficaz, proporcionando a las páginas web dinamismo y vida [10]. El corazón del lenguaje se embebe en los navegadores para interpretar los códigos (scripts) escritos en las páginas. Estos códigos contribuyen a controlar el navegador o crear páginas completas HTML sin que se tenga que ir nuevamente al servidor.

➤ **XHTML**

XHTML (*Lenguaje de Marcado de Hipertexto Extensible*) es una versión más estricta y limpia de HTML[11]. XHTML extiende HTML 4.0 combinando la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos.

➤ CSS

CSS es el lenguaje de hojas de estilos (Cascade Style Sheet, en inglés), creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML [12]. Este lenguaje se utiliza para definir el aspecto de todos los contenidos, es decir, el color, tamaño y tipo de letra de los párrafos de texto, la separación entre titulares y párrafos, la tabulación con la que se muestran los elementos de una lista, etc. El W3C es el encargado de definir los estándares a utilizar por los navegadores y actualmente existen los estándares CSS1, CSS2.0, CSS2.1 y se trabaja en el estándar CSS3 ya soportado por algunos navegadores modernos (Mozilla Firefox, Opera y Safari). El uso de hojas de estilos permite separar los contenidos, su presentación y es imprescindible para crear páginas web complejas. Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

1.5.2 Tecnologías

➤ AJAX

AJAX es un acrónimo de Asynchronous JavaScript + XML, que se puede traducir como JavaScript asíncrono + XML [13]. No es una tecnología en sí mismo. En realidad, se trata de la unión de varias tecnologías que se desarrollan de forma autónoma y que se unen de formas nuevas y sorprendentes. El término fue anunciado por primera vez en el artículo "Ajax: A New Approach to Web Applications".

Las tecnologías que forman AJAX son:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML, XSLT y JSON, para el intercambio y la manipulación de información.
- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.

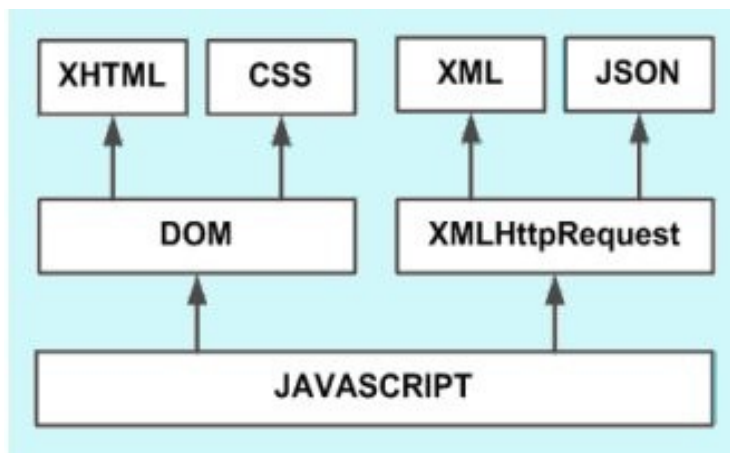


Figura 1: Tecnologías agrupadas bajo el concepto de AJAX

AJAX permite mejorar completamente la interacción del usuario con la aplicación ya que evita las recargas constantes de la página, y el intercambio de información con el servidor se produce en un segundo plano. Trata de hacer la creación de aplicaciones web más versátil e interactiva, permitiendo que las páginas web puedan realizar llamadas asíncronas al servidor de forma transparente mientras el usuario está trabajando. Las tecnologías AJAX se aplica en todos los navegadores web modernos, tales como Mozilla Firefox, Internet Explorer u Opera.

1.5.2.1 Framework para el lenguaje PHP

➤ **Symfony**

Symfony es un completo framework desarrollado con PHP 5 y diseñado para optimizar el desarrollo de las aplicaciones web. Está bajo licencia MIT convirtiéndolo en un producto libre y cuenta con una gran comunidad de desarrolladores y documentación precisa. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Entre sus principales características se destacan las siguientes:

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).
- Independiente del sistema gestor de bases de datos.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Su escalabilidad lo hace extensible al integrarse con plugins y librerías desarrolladas por otras personas.
- Hace uso de Propel como capa de ORM, que facilita su integración con los sistemas gestores de bases de datos relacionales más usados (MySQL, PostgreSQL, Microsoft SQL

Server y Oracle), además de integrarse con Doctrine ORM, a conveniencia del usuario.

- Symfony incluye su propio framework de pruebas (unitarias y funcionales) para la automatización de las mismas y cuenta con más de 9000 pruebas entre funcionales y unitarias sobre su mismo código, permitiendo que el mismo esté muy optimizado.
- Provee varios entornos (entorno de producción, de desarrollo y pruebas).

Symfony incluye un conjunto de tareas que automatizan las características de un proyecto web:

- La capa de presentación incluye plantillas de presentación globales (layouts) que constituye el maquetado del sistema, pudiendo diseñarlo sin necesidad de conocimientos del framework.
- Brinda una barra de depuración web para ayudar al programador a corregir errores en el código.
- Está basado en la reutilización de código a través de plugins y helpers para HTML, formularios, JavaScript y AJAX, que encapsulan una gran cantidad de código.
- Tiene incluido en su núcleo el framework Prototype, para trabajar con la capa de presentación de la aplicación, aunque se puede integrar perfectamente con JQuery, Mootools o ExtJS.
- Provee varios mecanismos de seguridad de la aplicación, como son las estrategias de escape de variables para evitar ataques contra el sistema y los mecanismos de gestión de sesión y credenciales para la autenticación de los usuarios.
- Disminuye el uso de ancho de banda y la carga del servidor al gestionar la caché de la aplicación.
- El sistema de enrutamiento que incluye permite el uso de URL limpias y más tomadas en cuenta por los buscadores.

Se utilizará como framework de desarrollo para PHP a Symfony porque reúne las mejores prácticas de desarrollo web e integra muchas otras librerías y herramientas que aportan fortaleza al producto final. Es compatible con todos los sistemas de gestión de bases de datos y provee varios mecanismos de seguridad que evitan los ataques al sistema. El uso de este framework reduce el tiempo de desarrollo de las aplicaciones mediante la incorporación de varias herramientas y clases.

1.5.2.2 Sistema gestor de base de datos

➤ PostgreSQL

PostgreSQL es un sistema de gestión de base de datos relacional orientada a objetos de código

abierto, es multiplataforma y está publicado bajo la licencia BSD. Posee soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados (en varios idiomas). Se incluye la mayoría de los tipos de datos de SQL. También soporta almacenamiento de objetos binarios grandes, como imágenes, sonidos o vídeo. Cuenta con interfaces nativas de programación para C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otros, y posee una buena documentación. Es altamente escalable, tanto en la enorme cantidad de datos que puede manejar y en el número de usuarios concurrentes que puede gestionar. Algunas características son:

- Lenguaje procedural propio denominado PL/PgSQL.
- Usa una arquitectura cliente/servidor de proceso por usuario.
- Utiliza la tecnología Multi-Version Concurrency Control (MVCC) para evitar el bloqueo innecesario.
- Ofrece soporte comercial a través de diferentes empresas de consultoría.
- Soporte para integridad referencial.
- Soporte nativo para SSL.
- Soporte multiusuario.
- Herencia entre tablas.

Se empleará PostgreSQL por ser un magnífico gestor de base de datos, lo cual se ha ganado por su desarrollo activo en los últimos años. Posee una gran escalabilidad y es multiplataforma. Tiene prácticamente todas las características de los demás gestores, siendo este una muy buena alternativa libre. Además nuestra universidad se encuentra inmersa en el proceso de migración a software libre en sus sistemas y PostgreSQL constituye la alternativa de código abierto y bajo licencia BSD a utilizar, como solución al problema planteado para la gestión de base de datos relacional.

1.5.2.3 Servidor de Aplicaciones Web

➤ Apache

Apache surge a partir de un servidor web de la NCSA (National Center for Super Computing Applications) como corrección a errores y mejoras importantes al producto inicial. Por los resultados logrados, es fundado el Grupo Apache, potenciador del nuevo producto al que se le hicieron mejoras en su arquitectura logrando modularidad. En 1999 los desarrolladores fundan la Apache Software Foundation, organización que da soporte legal al desarrollo de Apache y que lo mantiene actualmente bajo la licencia Apache Licence 2.0, que lo convierte en software libre.

Apache en su versión 2 es una solución más flexible y escalable, ofreciendo un conjunto de mejoras notables.

Características

- Soporte para el protocolo HTTP 1.1.
- Sencillo, con la configuración basada en archivos fáciles de configurar.
- Soporte para CGI (Common Gateway Interface).
- Soporte para FastCGI.
- Soporte de Host Virtuales.
- Soporte de autenticación HTTP.
- Perl integrado.
- Soporte de scripts PHP.
- Soporte de servlets de Java.
- Servidor proxy integrado.
- Estado del servidor y adaptación de registros.
- Soporte de Server Side Include (SSI).
- Soporte de Secured Socket Layer (SSL).
- Módulos multiproceso.
- Portable en tiempo de ejecución [14].

Para la solución al problema planteado en la presente investigación se hará uso de Apache 2.2.15 como servidor web, dadas las características favorables en cuanto a rendimiento, seguridad y escalabilidad. Además la Universidad de las Ciencias Informáticas lleva a cabo una estrategia para fomentar el uso de software libre en sus sistemas, y Apache constituye la alternativa libre en el hospedaje de aplicaciones para la web, con soporte nativo para PHP, lenguaje que fue seleccionado para el desarrollo de la solución propuesta. Es el servidor web por excelencia para sistemas operativos GNU/Linux.

1.5.2.4 Plugins y librerías utilizadas

➤ Plugin ckWebServicePlugin

El plugin ckWebServicePlugin permite construir una API de servicios web para las aplicaciones en Symfony. Viene con un potente y fácil generador de WSDL compatibles con WS-I, para un máximo de interoperabilidad con clientes PHP, .NET y Java.

➤ Librería phpmailer

PHPMailer es una clase escrita en PHP que facilita el envío de correo, posibilita el envío de adjuntos en formato HTML y con diferentes codificaciones, soporte para imágenes embebidas, headers personalizados, permite el envío a varios remitentes, autenticación por SMTP y además funciona con múltiples servidores de correo.

➤ **Libería JQuery UI**

jQuery UI es una librería de componentes para el framework jQuery que provee un conjunto de plugins, efectos visuales y widgets [15]. Su utilización permite construir aplicaciones web altamente interactivas.

1.5.2.5 Servicios Web

¿Qué es un servicio web?

Se denomina servicio web a cualquier servicio que se encuentra disponible a través de Internet, que utiliza el estándar XML para el sistema de mensajería, y no está ligado a ningún sistema operativo o lenguaje de programación [16]. Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario.

La arquitectura de un servicio web se define de la siguiente manera:

- **Service transport:** esta capa es responsable del transporte de mensajes entre aplicaciones a través de los protocolos HTTP, SMTP, FTP, y BEEP.
- **XML messaging:** esta capa es responsable de la codificación de mensajes en un formato común XML para que los mensajes se pueden entender por las aplicaciones que se comunican. Actualmente, esta capa incluye XML-RPC, SOAP y REST.
- **Service description:** esta capa es responsable de describir la interfaz pública del servicio web específico.
- **Service discovery:** esta capa es responsable de centralizar los servicios en un registro común, y proporcionando facilidad de publicación y búsqueda de los mismos.

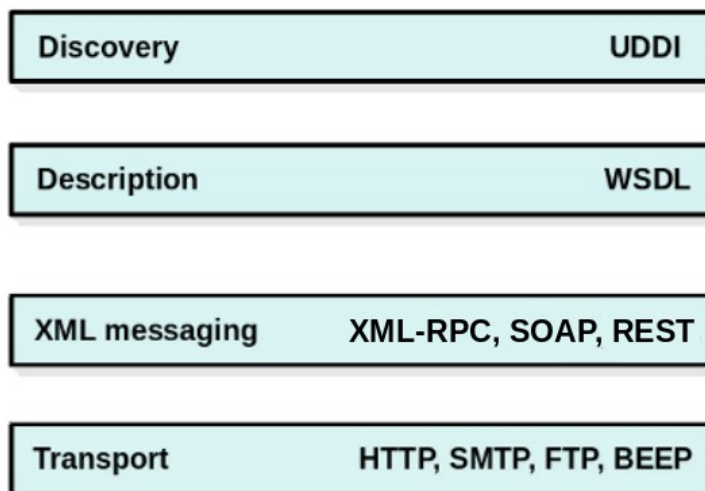


Figura 2: Pila de Protocolos para servicios web.

WSDL (Lenguaje de Descripción del Servicio Web)

WSDL son las siglas de Web Services Description Language, un formato XML que se utiliza para describir servicios Web [17]. La versión 1.1 está en estado de "propuesta de recomendación" por parte del W3C. WSDL describe la interfaz pública a los servicios Web. Está basado en XML y describe la forma de comunicación, es decir, los requisitos del protocolo y los formatos de los mensajes necesarios para interactuar con los servicios listados en su catálogo. Las operaciones y mensajes que soporta se describen en abstracto y se ligan después al protocolo concreto de red y al formato del mensaje.

UDDI (Universal Description, Discovery, and Integration)

UDDI son las siglas del catálogo de negocios de Internet denominado Universal Description, Discovery, and Integration [17]. Es uno de los estándares básicos de los servicios web cuyo objetivo es ser accedido por los usuarios y clientes y dar paso a documentos WSDL, en los que se describen los requisitos del protocolo y los formatos del mensaje solicitado para interactuar con los servicios web del catálogo de registros.

Variantes de la capa XML Messaging

➤ SOAP

SOAP (siglas de Simple Object Access Protocol) es un protocolo estándar creado por Microsoft, IBM y otros, está actualmente bajo el auspicio de la W3C que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. Es un marco extensible y descentralizado que permite trabajar sobre múltiples pilas de protocolos de redes

informáticas. PHP desde su versión 5.0, tiene soporte nativo para SOAP a través de una extensión.

SOAP auxiliándose del WSDL adquiere ventajas sobre REST y XML-RPC, al especificar un contrato entre el cliente y el proveedor del servicio. Cuando se desea implementar un servicio web desde un lenguaje de programación orientado a objetos, SOAP aprovecha las características de los tipos de datos complejos. Un mensaje SOAP, es un documento XML, por lo que la seguridad de los datos enviados, puede asegurarse a través de un conjunto de técnicas: Encriptación XML, Firma Digital XML, Certificados XKMS (XML Key Management Specification), SAML (Security Assertion Markup Language) y validación de datos. Integrado en lenguajes como PHP, JAVA y C# a través de APIs [18].

➤ XML-RPC

XML-RPC es un protocolo de llamada a procedimiento remoto usando HTTP para el transporte y XML para la codificación. Es una especificación y una serie de implementaciones que permiten a los software que se ejecutan en sistemas operativos diferentes realizar llamadas de procedimiento a través de Internet.

Está diseñado para ser tan simple como sea posible, permitiendo al mismo tiempo las estructuras de datos complejos para ser transmitida, procesada y devuelta.

➤ REST

REST son las siglas en inglés de Representational State Transfer. Es un estilo de arquitectura y no un estándar definido. Se basa en las URIs para identificar recursos, los cuales se retornan en XML puro. REST es una manera simplificada de ejecutar métodos remotos. Utiliza HTTP como protocolo de transporte y HTTPS para brindar seguridad.

Selección de tecnologías para implementar los servicios web

De acuerdo con el estudio realizado sobre las tecnologías relacionadas con los servicios web se escoge SOAP como protocolo de la capa de mensajería para la implementación de los servicios web, por las ventajas que ofrece en relación a la descripción del servicio, auxiliándose de un documento WSDL. Además de que el framework para php Symfony seleccionado para el desarrollo de esta aplicación se integra fácilmente con el plugin ckWebServicePlugin, el cual haciendo uso de SOAP aprovecha la capa de acceso a datos del framework y expone las acciones de la capa correspondiente al controlador provista por la arquitectura del framework como servicios web; así como la generación automática del documento WSDL que cumple con los lineamientos de la WS-I, de máxima interoperabilidad con clientes PHP, .NET y Java.

1.5.3 Herramientas de soporte al proceso de desarrollo de software

➤ **Netbeans IDE⁸**

Netbeans IDE (del inglés, Integrated Development Environment) es un entorno de desarrollo, una herramienta para escribir, compilar, depurar y ejecutar programas. Aunque está escrito en Java, posee soporte para PHP y características visuales para el desarrollo web. Es un producto libre, multiplataforma y gratuito sin restricciones de uso. Se integra muy bien con sistemas de control de versiones y sistema gestores de bases de datos (MySQL y PostgreSQL). A partir de su versión 6.8, integra soporte para el framework Symfony versión 1.2.

➤ **pgAdmin III**

pgAdmin III es una herramienta libre para administrar el gestor de bases de datos PostgreSQL. Está escrita en C++ usando la librería gráfica wxWidgets y es multiplataforma. Está diseñado para escribir consultas SQL simples, así como desarrollar bases de datos complejas, su interfaz gráfica soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados, soporte para el motor de replicación Slony-I y mucho más. La conexión al servidor puede hacerse mediante conexión TCP/IP o Unix Domain Sockets (en plataformas *nix), y puede encriptarse mediante SSL para mayor seguridad.

➤ **Visual Paradigm for UML**

Visual Paradigm for UML (VP-UML) es una herramienta de diseño UML y herramienta CASE UML diseñada para la ayuda al desarrollo de software. VP-UML soporta los principales estándares de la industria tales como Lenguaje de Modelado Unificado (UML), SysML, BPMN, XMI, etc. Ofrece un completo conjunto de herramientas a los equipos de desarrollo de software necesarias para los requisitos de la captura, software de planificación, la planificación de controles, el modelado de clases, modelado de datos, etc.

➤ **RapidSVN**

RapidSVN es un cliente de interfaz gráfica para la comunicación con servidores Subversion y facilita el versionado de ficheros, desde una interfaz sencilla e intuitiva. Está disponible para plataformas Windows, Linux, MAC OS y Solaris, y es distribuido bajo la licencia GNU General Public License v3. Es rápido y eficiente, está escrito en C++.

➤ **Firebug⁹**

8 <http://netbeans.org/>

9 <http://www.getfirebug.com>

Firebug es un complemento para el navegador Mozilla Firefox que permite inspeccionar y editar el código Javascript, XHTML y CSS de una página web así como analizar las peticiones y respuestas del cliente y servidor, monitorizar el tiempo de carga de las páginas, además de analizar las peticiones vía Ajax. Está distribuido bajo licencia GPL.

1.6 Conclusiones parciales

En el desarrollo de este capítulo se abordaron conceptos generales acerca de la búsqueda y recuperación de información, así como los sistemas existentes para la web de búsqueda y recuperación de información, con el objetivo de ubicar al lector en el dominio del problema. Son analizados algunos de los directorios de software, para documentar las fundamentales características que proveen y determinándose que es necesaria la implementación de un sistema web que esté acorde con las necesidades de la búsqueda de alternativas libres en la migración a Software Libre en Cuba.

Para ello, se abordan elementos de la metodología SXP que guiará el proceso de desarrollo de software, así como las tecnologías y lenguajes usados en la construcción del sistema, seleccionando Apache 2.2.15, PostgreSQL 8.4.5 y Symfony 1.4.6 para el desarrollo, y como Sistema Operativo la distribución GNU/Linux Nova 3.0.

Se describen conceptos relacionados con los servicios web, eligiéndose para la implementación de los mismos a SOAP como capa de mensajería en conjunto con XML para la codificación de los mensajes y WSDL para su descripción. Se definen además un conjunto de herramientas de soporte al ciclo de desarrollo.

Capítulo 2. Definición del directorio de software

En el presente capítulo se recogen las funcionalidades del directorio de software reflejadas en las Historias de Usuarios, así como la planificación de las iteraciones del ciclo de desarrollo.

2.1 Lista de Reserva del Producto (LRP)

La lista de reserva del producto es una lista priorizada de requisitos que define el trabajo que se va a realizar en el proyecto y determina el orden en que se le irá dando cumplimiento durante las iteraciones a cada requerimiento recogido según la prioridad establecida en cada uno; también abarca las características que el producto debe tener.

En la siguiente tabla se recogen en cada una de las filas los requisitos identificados del producto. Se muestra el nombre del desarrollador que dará cumplimiento al requisito, el número identificativo del requisito, la descripción del requisito, el tiempo estimado para su cumplimiento y el nombre del estimador respectivamente.

RF (Requisitos Funcionales)				
Prioridad	Ítem *	Descripción	Estimación	Estimado por
Subsistema: Directorio de software				
Alta				
Yanirys	1	Insertar software.	1 hora	Yanirys
	2	Editar un software dado su identificador, se recibirían los datos del nuevo software.	3 horas	
	3	Listar software por categoría dado su identificador.	5 horas	
	4	Eliminar un software dado su identificador.	1 hora	
	5	Insertar una versión de software dado el nombre del software, se recibirían los datos de la nueva versión del software.	1 hora	
	6	Listar todas las versiones de un software dado su identificador.	1 hora	
	7	Editar una versión de software dado su identificador, se recibirían los datos de la nueva	1 hora	

		versión del software.		
	8	Obtener la versión de un software dado su identificador.	1 hora	
	9	Eliminar una versión de software dado su identificador.	5 horas	
	10	Insertar una imagen para una versión de software dado su identificador.	2 horas	
	11	Listar las imágenes por cada versión de software dado su identificador.	3 horas	
	12	Eliminar imagen de un software dado su identificador.	1 hora	
	13	Listar los software dada su categoría por varios criterios (según cantidad de descarga, cantidad de visitas, cantidad de recomendaciones).	1 hora	
	14	Mostrar evolución del software privativo, en cuanto a los precios de sus versiones mediante la graficación de estos datos.	1 día	
Odaimy	15	Insertar una alternativa a un software determinado dado el identificador de la versión del software.	1 día	Odaimy
	16	Editar la alternativa de un software dado su identificador, se recibirían los datos de la nueva alternativa.	5 horas	
	17	Listar todas las alternativas de un software dado el identificador de la versión del software correspondiente.	3 horas	
	18	Eliminar una alternativa dado su identificador.	3 horas	
	19	Obtener alternativa dado su identificador.	2 horas	
Media				
Odaimy	20	Insertar categoría, se recibirían los datos de la nueva categoría.	1 día	Odaimy

	21	Editar una categoría dado su identificador, se recibirían los datos de la nueva categoría.	5 horas	
	22	Eliminar categoría dado su identificador.	3 horas	
	23	Mostrar las subcategorías pertenecientes a una categoría dada.	2 hora	
	24	Mostrar una categoría de software dado un identificador.	1 hora	
Javier	25	Insertar licencia, se recibirían los datos de la nueva licencia.	1 día	Javier
	26	Modificar una licencia dado su identificador, se recibirían los datos de la nueva licencia.	5 horas	
	27	Eliminar licencia dado su identificador.	3 horas	
	28	Mostrar todas las licencias de software.	3 horas	
	29	Mostrar una licencia de software dado un identificador.	1 hora	
Javier	30	Enviar listado por email.	1 día	Javier
Eider	31	Insertar plataforma, se recibirían los datos de la nueva plataforma.	1 día	Eider
	32	Modificar una plataforma dado su identificador, se recibirían los datos de la nueva plataforma.	5 horas	
	33	Eliminar una plataforma dado su identificador.	3 horas	
	34	Mostrar todos las plataformas.	3 horas	
	35	Mostrar una plataforma dado un identificador.	1 hora	
Baja				
Yanirys	36	Insertar comentario, se recibirían los datos de un nuevo comentario.	1 día	Yanirys
	37	Eliminar comentario dado su identificador.	3 horas	Yanirys
	38	Mostrar todos los comentarios asociados a una versión de software	3 horas	Yanirys

		dado su identificador.		
	39	Mostrar un comentario dado su identificador.	1 hora	Yanirys
	40	Editar comentario dado su identificador.	1 hora	Yanirys
Yanirys	41	Insertar listado, se recibirían los datos del nuevo listado.	1 día	Yanirys
	42	Eliminar listado dado su identificador.	3 horas	Yanirys
	43	Mostrar todos los listados creados por determinado usuario dado su identificador.	3 horas	Yanirys
	44	Editar listado dado su identificador.	1 hora	Yanirys
	45	Adicionar software a un listado.	1 hora	Yanirys
Eider	46	Insertar repositorio, se recibirían los datos del nuevo repositorio.	2 horas	Eider
	47	Modificar repositorio dado su identificador, se recibirían los datos del repositorio.	2 horas	Eider
	48	Eliminar repositorio dado su identificador.	1 hora	Eider
	49	Listar todos los repositorios.	1 hora	Eider
	50	Mostrar repositorio dado su identificador.	1 hora	Eider
Odaimy	51	Importar paquetes de un repositorio	2 horas	Odaimy
Odaimy	52	Mostrar paquetes de un repositorio	1 hora	Odaimy
Odaimy	53	Eliminar paquete dado su identificador	1 hora	Odaimy
Yanirys	54	Listar todos los idiomas de los software.	1 hora	Yanirys
Yanirys	55	Crear evaluación a una versión de software.	1 hora	Yanirys
Yanirys	56	Crear evaluación a una alternativa.	1 hora	Yanirys
Yanirys	57	Guardar descarga de un software.	1 hora	Yanirys
Yanirys	58	Listar todos los tipos de alternativas.	1 hora	Yanirys

Yanirys	59	Mostrar un tipo de alternativa dado su identificador.	1 hora	Yanirys
---------	----	---	--------	---------

2.2 Lista de Historias de Usuario

Las historias de usuario son la técnica utilizada en SXP para especificar los requerimientos del sistema, lo que sería semejante a los casos de uso en la metodología RUP. Las mismas guían el proceso de pruebas al sistema y son utilizadas para estimar el tiempo de desarrollo del producto. A continuación se definen las Historias de Usuarios y a las tareas de ingeniería asociadas a cada una de ellas.

2.2.1 Historia de Usuario Gestionar Versiones de Software

Historia de Usuario	
Número: 1	Nombre Historia de Usuario: Gestionar Versiones de Software
Modificación de Historia de Usuario Número: 1	
Usuario: Yanirys Ortíz Lima	Iteración Asignada: 1
Prioridad en Negocio: Alta	Puntos Estimados: 2 semanas
Riesgo en Desarrollo: Alto	Puntos Reales: 1 semana
<p>Descripción: Las versiones de software contienen las características de interés de cada software: Nombre del software del cual constituyen una versión, descripción, versión, licencia, imagen principal, plataformas, tamaño, precio, origen del precio, software similares (puede ser en la misma plataforma u otra), cantidad de visitas (contador de veces que se solicitó ver la versión de software en cuestión), URL oficial del software, si está visible en el directorio, fecha de agregación al directorio, usuario que lo agregó al directorio de software, cantidad de recomendaciones, cantidad de visitas, comentarios de visitantes, listado de software similares, idioma, total de descargas. Se gestionan las versiones de software, permitiendo insertar, modificar, eliminar y mostrar cada una de ellas.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> Para insertar una versión de software son necesarios los siguientes datos: nombre del software del cual constituyen una versión, descripción, versión, licencias, imagen principal, plataformas, categoría, tamaño, precio, origen del precio, URL oficial del software, URL de descarga para cada plataforma, si está visible en el directorio, fecha 	

de agregación al directorio, usuario que lo agregó al directorio de SW, idioma, cantidad de visitas (que constituye un contador de las visitas realizadas al software), cantidad de recomendaciones (que constituye un contador de las recomendaciones realizadas del software).

- x Insertar una versión de software dado el nombre del software implica que si no existe el software al cual se le va a adicionar una versión, este es creado junto a la nueva versión de software.
- Para **modificar** se selecciona la versión de software a modificar y se actualiza la misma.
- Para **mostrar** las versiones de software se selecciona el software al cual se desea encontrar las versiones que posee y también debe poderse mostrar los detalles de una versión de software en específico.
- Para **eliminar** se selecciona la versión de software y se confirma la eliminación.
 - x Eliminar una versión de software implica eliminar todas las relaciones de la misma.

Prototipo de interfaz:

The image shows a web form for creating a software version. The form is organized into two columns of input fields. The left column includes fields for 'Software', 'Versión', 'Descripción', 'URL oficial', and 'Imagen Principal' (with an 'Examinar' button). The right column includes fields for 'Precio', 'Origen del precio', 'Tamaño', 'Categoría' (a dropdown menu), and a 'Visible' checkbox. Below these fields is a license management section with three tabs: 'Licencias', 'Plataformas', and 'Idiomas'. The 'Licencias' tab is active, showing two boxes: 'Asignados' (empty) and 'Disponibles' (containing 'GPL', 'MIT', and 'LGPL'). A green plus sign is positioned between the boxes, and a red minus sign is below it. At the bottom right of the form are 'Enviar' and 'Cancelar' buttons.

Figura 1: Crear versión de software

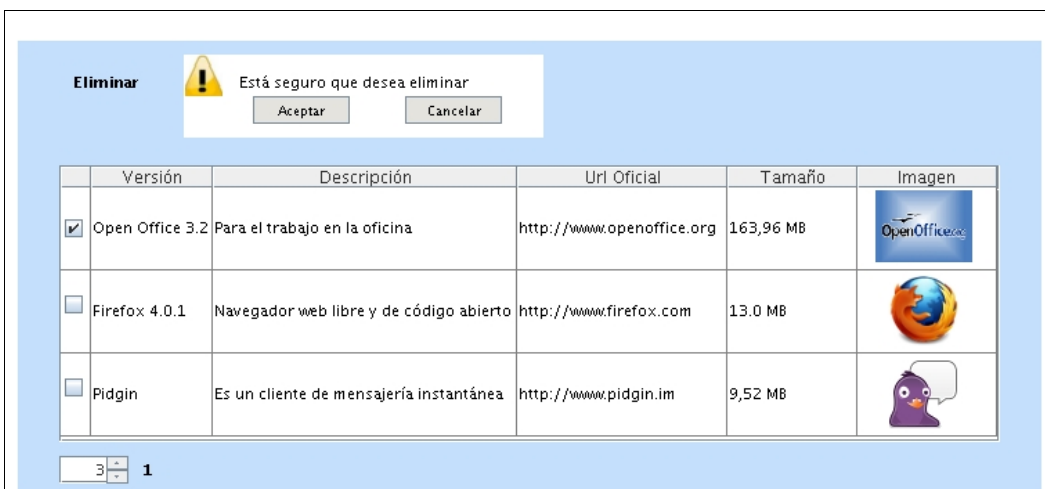


Figura 2: Eliminar versiones de software

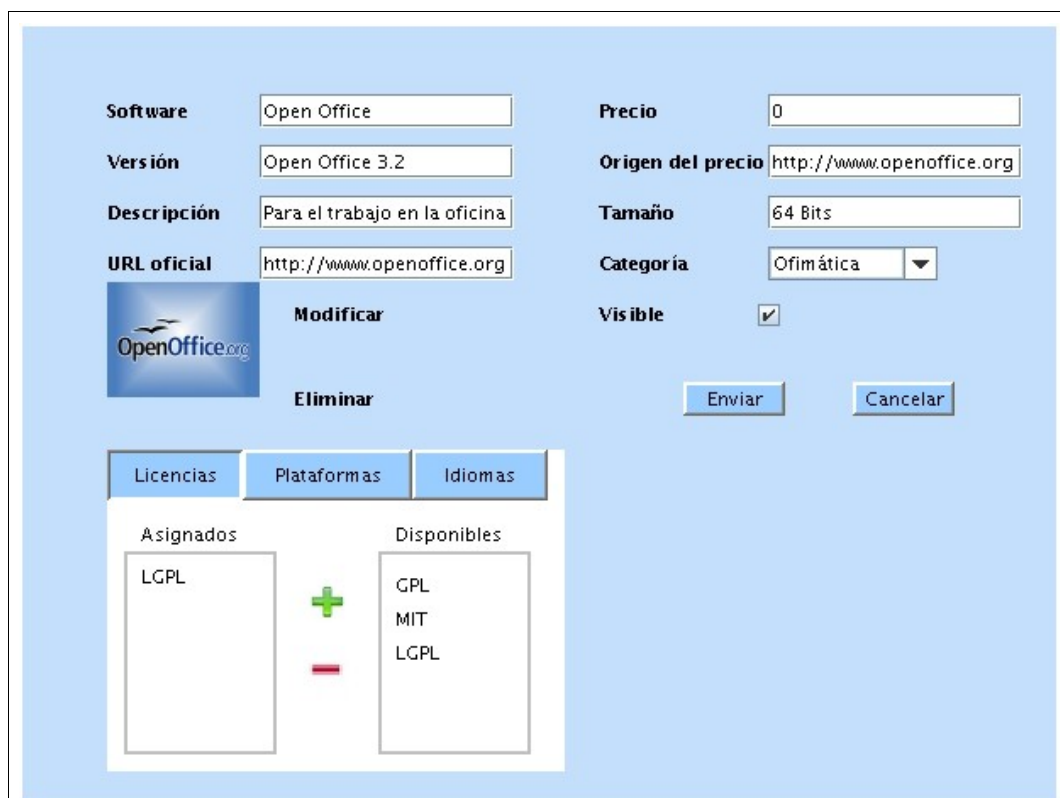





Figura 3: Editar versión de software

	Versión	Descripción	Url Oficial	Tamaño	Imagen
<input type="checkbox"/>	Open Office 3.2	Para el trabajo en la oficina	http://www.openoffice.org	163,96 MB	
<input type="checkbox"/>	Firefox 4.0.1	Navegador web libre y de código abierto	http://www.firefox.com	13.0 MB	
<input type="checkbox"/>	Pidgin	Es un cliente de mensajería instantánea	http://www.pidgin.im	9,52 MB	

3 1

Figura 4: Listar versiones de software

2.2.2 Historia de Usuario Gestionar Software

Historia de Usuario	
Número: 2	Nombre Historia de Usuario: Gestionar software
Modificación de Historia de Usuario Número: 1	
Usuario: Yanirys Ortíz Lima	Iteración Asignada: 1
Prioridad en Negocio: Alta	Puntos Estimados: 1 semana
Riesgo en Desarrollo: Alta	Puntos Reales: 1 semana
Descripción: Los software incluyen: nombre, categoría e imagen principal.	
Observaciones:	
<ul style="list-style-type: none"> Para insertar un software son necesarios los siguientes datos: nombre, categoría e imagen principal. <ul style="list-style-type: none"> x Un software es insertado solo cuando se adicione por primera vez una determinada versión del mismo. Para modificar se selecciona el software a modificar y se actualiza el mismo. Para mostrar los software se selecciona la categoría de la cual desea ver los software pertenecientes a esta, se debe tener un paginado de manera que la lista visualizada no sea demasiado extensa. 	

- Para **eliminar** se selecciona el software y se confirma la eliminación.
 - x Eliminar un software implica eliminar todas las versiones existentes del software, así como las relaciones del mismo.

Prototipo de interfaz: No tiene, es a través de servicios web.

2.2.3 Historia de Usuario Gestionar imágenes

Historia de Usuario	
Número: 3	Nombre Historia de Usuario: Gestionar imágenes
Modificación de Historia de Usuario Número: 1	
Usuario: Yanirys Ortíz Lima	Iteración Asignada: 1
Prioridad en Negocio: Alta	Puntos Estimados: 0.5 semanas
Riesgo en Desarrollo: Medio	Puntos Reales: 0.5 semanas
<p>Descripción: Las imágenes constituyen un nomenclador para el desarrollo de la gestión de las versiones de software. Incluye el identificador de la versión de software al cual se le va a crear la imagen, dirección y descripción de la misma. Se gestionan las imágenes; permitiendo insertar, eliminar y mostrar cada uno de ellas.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> • Para insertar una imagen son necesarios los siguientes datos: identificador de la versión de software al cual se le va a crear la imagen, dirección y descripción de la misma. • Para mostrar las imágenes se selecciona la versión de software del cual se desea ver las imágenes. • Para eliminar se selecciona la imagen y se confirma la eliminación. 	
Prototipo de interfaz: No tiene, es a través de servicios web.	

2.2.4 Historia de Usuario Gestionar alternativas a las versiones de software

Historia de Usuario	
Número: 4	Nombre Historia de Usuario: Gestionar alternativas a las versiones de software

Modificación de Historia de Usuario Número: 1	
Usuario: Odaimy Alvarez Acosta	Iteración Asignada: 1
Prioridad en Negocio: Alta	Puntos Estimados: 2 semanas
Riesgo en Desarrollo: Alto	Puntos Reales: 1 semana
<p>Descripción: Se le dará a cada software privativo una clasificación de alternativa que pueden ser: emulado, necesitado de desarrollo, recomendado y con un listado de aplicaciones alternativas. En caso de ser emulado se indicará el software para visualizarlo, en caso de que tenga un listado de aplicaciones alternativas se indicarán las mismas que cada una de ellas pueden constituir uno o un conjunto de aplicaciones y en el caso de ser recomendado necesitará de la aprobación del administrador para estar visible para los usuarios. En caso de tener necesidad de desarrollo no se indicará ninguna alternativa. Se gestionan las alternativas, permitiendo insertar, eliminar, modificar y mostrar cada uno de ellos.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> • Para insertar una alternativa son necesarios los siguientes datos: identificador de la versión de software a la cual se le crea la alternativa, identificador del tipo de alternativa, el nombre del usuario que crea la alternativa, los identificadores de las versiones de software que constituyen las alternativas, el tipo de alternativa, aprobación de la misma y descripción de la alternativa. • Para modificar se selecciona la alternativa a modificar y se actualiza la misma. • Para mostrar las alternativas se selecciona la versión de software al que se le quiere encontrar las alternativas y también debe poderse mostrar los detalles de una alternativa en específico. • Para eliminar se selecciona la alternativa y se confirma la eliminación. <ul style="list-style-type: none"> x Eliminar una alternativa implica eliminar todos las relaciones de la misma. 	
Prototipo de interfaz:	

Versión de Software
Descripción
Tipo de alternativa ▾
Aprobado

Alternativas

Filtro

Software:

Vers iones

Figura 5: Crear alternativa

Filtro

Software:

	Software	Versión	Categoría
<input type="checkbox"/>	Internet Explorer	7	Internet

1

Datos generales

Tipo de alternativa	Listado de Aplicaciones
Descripción	Para navegar en Internet
Usuario	admin
Aprobado	Sí

Alternativas

Firefox 3.5, Firefox 3.6

Figura 6: Listar alternativas

Versión de Software
Descripción
Tipo de alternativa
Aprobado

Alternativas

3.6

+
-

Filtro

Software:

Vers iones

3.6
3.5

Figura 7: Editar alternativa

	Software	Versión	Categoría
<input type="checkbox"/>	Internet Explorer	7	Internet

1

Datos generales

Tipo de alternativa	Listado de Aplicaciones
Descripción	Para navegar en Internet
Usuario	admin
Aprobado	Sí

Alternativas

Firefox 3.5, Firefox 3.6

Eliminar

Está seguro que desea eliminar

Figura 8: Eliminar alternativa

2.2.5 Historia de Usuario Gestionar Comentarios

Historia de Usuario	
Número: 5	Nombre Historia de Usuario: Gestionar Comentarios
Modificación de Historia de Usuario Número: 1	
Usuario: Yanirys Ortíz Lima	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas
<p>Descripción: Los comentarios constituyen un nomenclador para el desarrollo de la gestión de las versiones de software. Incluye el autor, asunto, texto del comentario, así como su aprobación. Se gestionan los comentarios, permitiendo insertar, eliminar, modificar y mostrar cada uno de ellos.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> • Para insertar un comentario son necesarios los siguientes campos: identificador de la versión de software al cual se le va a crear el comentario, nombre del autor, asunto y texto del comentario, aprobación (visible o no). • Para modificar se selecciona el comentario a modificar y se actualiza el mismo. • Para mostrar los comentarios se selecciona la versión de software de la cual desea ver los comentarios, se debe tener un paginado de manera que la lista visualizada no sea demasiado extensa y también debe poderse mostrar los detalles de un comentario en específico. • Para eliminar se selecciona el comentario y se confirma la eliminación. 	
<p>Prototipo de interfaz: No tiene, es a través de servicios web.</p>	

2.2.6 Historia de Usuario Gestionar listados de software

Historia de Usuario	
Número: 6	Nombre Historia de Usuario: Gestionar listados de software
Modificación de Historia de Usuario Número: 1	

Usuario: Yanirys Ortíz Lima	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas
<p>Descripción: Los listados constituyen listas que cada usuario puede crear y adicionarle software estas y así poder recomendar estas listas de software mediante email a un amigo. Incluye el usuario que lo crea, nombre del listado, así como su descripción. Se gestionan los listados permitiendo insertar, modificar, eliminar y mostrar cada una de ellas.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> • Para insertar un listado son necesarios los siguientes campos: identificador del usuario que crea el listado, nombre y descripción del listado. • Para modificar se selecciona el listado a modificar y se actualiza el mismo. • Para mostrar los listados se selecciona el usuario del cual se quieren ver los listados creados, se debe tener un paginado de manera que la lista visualizada no sea demasiado extensa y también debe poderse mostrar los detalles de un listado en específico. • Para eliminar se selecciona el listado y se confirma la eliminación. <ul style="list-style-type: none"> x Eliminar una listado implica eliminar todos las relaciones del mismo. 	
<p>Prototipo de interfaz: No tiene, es a través de servicios web.</p>	

2.2.7 Historia de Usuario Gestionar repositorios

Historia de Usuario	
Número: 7	Nombre Historia de Usuario: Gestionar repositorios
Modificación de Historia de Usuario Número: 1	
Usuario: Eider Ochoa Enríquez	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas
Riesgo en Desarrollo: Medio	Puntos Reales: 0.5 semanas
<p>Descripción: Los repositorios constituyen la fuente desde la cual se obtendrá las aplicaciones necesarias para la automatización de la creación de las versiones de software. Se gestionan los repositorios, permitiendo insertar, modificar, eliminar y mostrar cada uno de ellos.</p>	

Observaciones:

- Para **insertar** un repositorio son necesarios los siguientes datos: nombre y dirección del mismo.
- Para **modificar** se selecciona el repositorio a modificar y se actualiza el mismo.
- Para **mostrar** se debe tener un paginado de manera que la lista visualizada no sea demasiado extensa y también debe poderse mostrar los detalles de un repositorio en específico.
- Para **eliminar** se selecciona el repositorio y se confirma la eliminación.

Prototipo de interfaz:

Nombre

Dirección

Figura 9: Crear repositorio

Nombre

Dirección

Figura 10: Editar repositorio

	Nombre	Dirección
<input type="checkbox"/>	Nova	ftp://nova.f10.uci.cu
<input type="checkbox"/>	Ubuntu	http://ubuntu.uci.cu

Figura 11: Listar repositorio

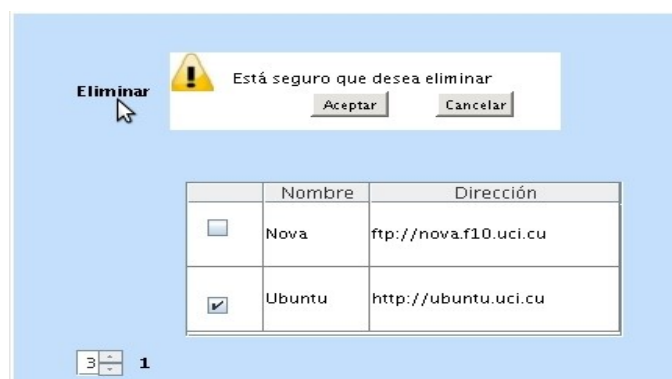


Figura 12: Eliminar repositorio

2.2.8 Historia de Usuario Gestionar Categorías de Software

Historia de Usuario	
Número: 8	Nombre Historia de Usuario: Gestionar Categorías de Software
Modificación de Historia de Usuario Número: 1	
Usuario: Odaimy Alvarez Acosta	Iteración Asignada: 2
Prioridad en Negocio: Media	Puntos Estimados: 1 semana
Riesgo en Desarrollo: Media	Puntos Reales: 0.5 semanas
<p>Descripción: Las categorías de software constituye un nomenclador para el desarrollo de la gestión de las versiones de software. Incluyen la categoría padre, el nombre, una breve descripción y una imagen de la misma. Se gestionan las categorías de software; permitiendo insertar, modificar, eliminar y mostrar cada una de ellas.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> • Para insertar una categoría son necesarios los siguientes datos: identificador de la categoría a la que pertenece, nombre, descripción e imagen de la misma (opcional). • Para modificar se selecciona la categoría a modificar y se actualiza la misma. • Para mostrar se selecciona la categoría de la cual se desea obtener las subcategorías, se debe tener un paginado de manera que la lista visualizada no sea demasiado extensa y también debe poderse mostrar los detalles de una categoría en específico. 	

- Para **eliminar** se selecciona la categoría y se confirma la eliminación.

Prototipo de interfaz:

Nombre

Descripción

Categoría padre Raiz ▼

Imagen

Figura 13: Crear categoría

Nombre

Descripción

Categoría padre Raiz ▼

Figura 14: Editar categoría

	Nombre	Descripción	Imagen
<input type="checkbox"/>	Desarrollo	Para el trabajo en la programación	
<input type="checkbox"/>	Internet	Para el trabajo en la web	
<input type="checkbox"/>	Ofimática	Para el trabajo en la oficina	

Figura 15: Listar categorías

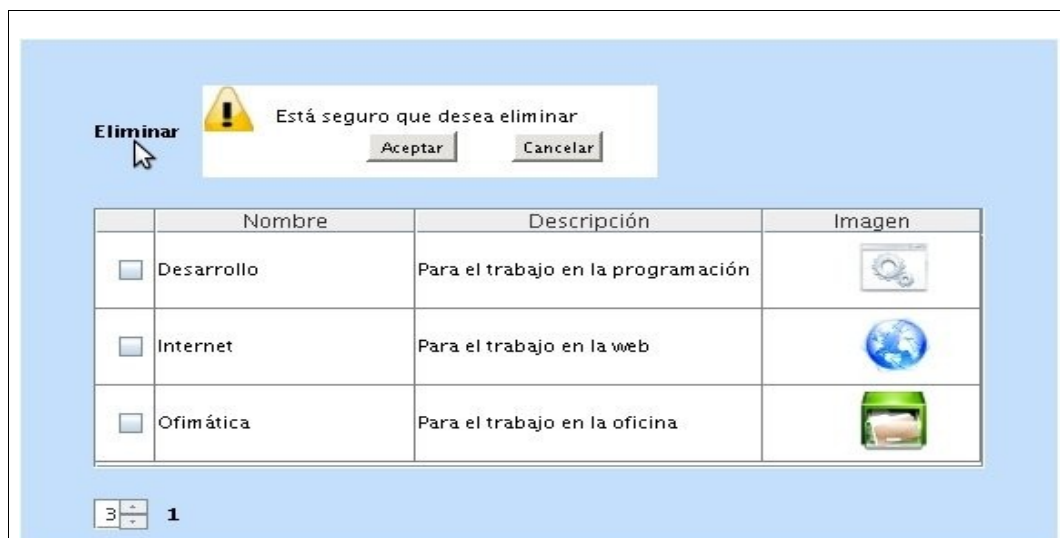


Figura 16: Eliminar categorías

2.2.9 Historia de Usuario Gestionar Licencias de Software

Historia de Usuario	
Número: 9	Nombre Historia de Usuario: Gestionar Licencias de Software
Modificación de Historia de Usuario Número: 1	
Usuario: Javier Méndez Pérez	Iteración Asignada: 2 semanas
Prioridad en Negocio: Media	Puntos Estimados: 1 semana
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas
<p>Descripción: Las licencias de software constituye un nomenclador para el desarrollo de la gestión de las versiones de software. Incluye el nombre de la licencia y una descripción de cada una de ellas. Se gestionan las licencias de software, permitiendo insertar, modificar, eliminar y mostrar cada una de ellas.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> • Para insertar una licencia deben completarse los siguientes campos: nombre y descripción de la misma. • Para modificar se selecciona la licencia a modificar y se actualiza la misma. 	

- Para **mostrar** se debe tener un paginado de manera que la lista visualizada no sea demasiado extensa y también debe poderse mostrar los detalles de una licencia en específico.
- Para **eliminar** se selecciona la licencia y se confirma la eliminación.

Prototipo de interfaz:

Nombre

Descripción

Figura 17: Crear licencia

Nombre

Descripción

Figura 18: Editar licencia

	Nombre	Descripción
<input type="checkbox"/>	GPL	General Public Licence
<input type="checkbox"/>	LGPL	LGPL
<input type="checkbox"/>	MIT	MIT

Figura 19: Listar licencia

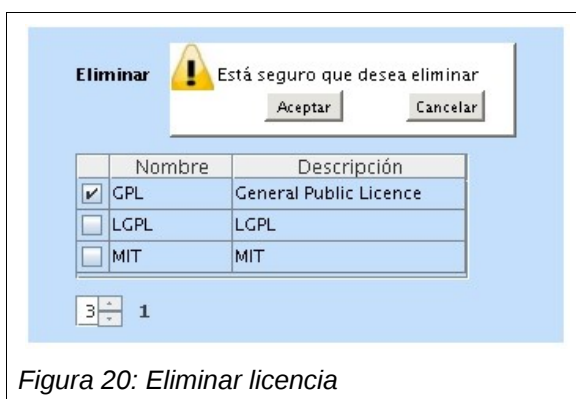


Figura 20: Eliminar licencia

2.2.10 Historia de Usuario Gestionar Plataformas

Historia de Usuario	
Número: 10	Nombre Historia de Usuario: Gestionar Plataformas
Modificación de Historia de Usuario Número: 1	
Usuario: Eider Ochoa Enriquez	Iteración Asignada: 2
Prioridad en Negocio: Media	Puntos Estimados: 1 semana
Riesgo en Desarrollo: Medio	Puntos Reales: 0.5 semanas
<p>Descripción: Las plataformas constituyen un nomenclador para el desarrollo de la gestión de las versiones de software. Incluye el nombre de la plataforma, una descripción, así como una imagen de cada uno de ellos. Se gestionan las plataformas; permitiendo insertar, modificar, eliminar y mostrar cada uno de ellas.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> • Para insertar una plataforma son necesarios los siguientes datos: nombre, descripción e imagen de la misma(es opcional). • Para modificar se selecciona la plataforma a modificar y se actualiza la misma. • Para mostrar se debe tener un paginado de manera que la lista visualizada no sea demasiado extensa y también debe poderse mostrar los detalles de una plataforma en específico. • Para eliminar se selecciona la plataforma y se confirma la eliminación. 	
Prototipo de interfaz:	

Nombre

Descripción

Imagen

Figura 21: Crear plataforma

Nombre

Descripción



Figura 22: Editar plataforma




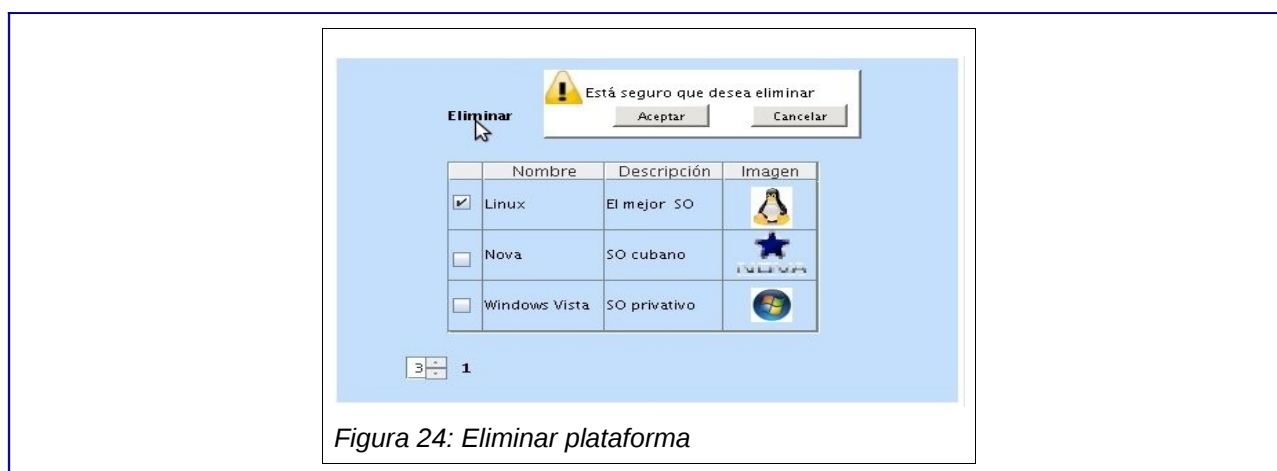
	Nombre	Descripción	Imagen
<input type="checkbox"/>	Linux	El mejor SO	
<input type="checkbox"/>	Nova	SO cubano	
<input type="checkbox"/>	Windows Vista	SO privativo	

Figura 23: Listar plataforma



2.2.11 Historia de Usuario Enviar e-mail

Historia de Usuario	
Número: 11	Nombre Historia de Usuario: Enviar e-mail
Modificación de Historia de Usuario Número: 1	
Usuario: Javier Méndez Pérez	Iteración Asignada: 2
Prioridad en Negocio: Media	Puntos Estimados: 2 semanas
Riesgo en Desarrollo: Media	Puntos Reales: 2 semanas
Descripción: El envío de email permite recomendar a un amigo un software o un listado de estos.	
Observaciones: <ul style="list-style-type: none"> Para enviar un email son necesarios los siguientes datos: dirección email del destinatario, dirección email del remitente, asunto del email y un listado de software que constituye el cuerpo del mensaje. 	
Prototipo de interfaz: No tiene, es a través de servicios web.	

2.2.12 Historia de Usuario Listar los tipos de alternativas

Historia de Usuario

Número: 12	Nombre Historia de Usuario: Listar los tipos de alternativas		
Modificación de Historia de Usuario Número: 1			
Usuario: Yanirys Ortíz Lima	Iteración Asignada: 3		
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas		
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas		
Descripción: Las tipos de alternativas constituyen un nomenclador para el desarrollo de las alternativas. Estos tipos de alternativas se encuentran definidas por el equipo de desarrollo.			
Observaciones:			
<ul style="list-style-type: none"> Para mostrar se listan todos los tipos de alternativas existentes y también debe poderse mostrar los detalles de un tipo de alternativa en específico 			
Prototipo de interfaz: No tiene, es a través de servicios web.			

2.2.13 Historia de Usuario Evaluar software y alternativas de software

Historia de Usuario			
Número: 13	Nombre Historia de Usuario: Evaluar software y alternativas de software		
Modificación de Historia de Usuario Número: 1			
Usuario: Yanirys Ortíz Lima	Iteración Asignada: 3		
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas		
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas		
Descripción: A los usuarios se le da la posibilidad de evaluar los software y las alternativas. Esta evaluación incluye los siguiente datos: el usuario que realiza la evaluación, el peso de la evaluación (1 hasta el 5) y el identificador de el software o de la alternativa a evaluar.			
Observaciones:			
<ul style="list-style-type: none"> Para insertar una evaluación son necesarios los siguientes datos: usuario que realiza la evaluación y peso de la evaluación. <ul style="list-style-type: none"> x En el caso de evaluar una versión de software se especifica además su 			

<p>identificador.</p> <p>x En el caso de evaluar una alternativa se especifica además su identificador.</p>
<p>Prototipo de interfaz: No tiene, es a través de servicios web.</p>

2.2.14 Historia de Usuario Listar idiomas

Historia de Usuario	
Número: 14	Nombre Historia de Usuario: Listar idiomas
Modificación de Historia de Usuario Número: 1	
Usuario: Yanirys Ortíz Lima	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas
<p>Descripción: Los idiomas constituyen un nomenclador para el desarrollo de las versiones de software. Estos idiomas se encuentran definidos por el equipo de desarrollo e incluyen el nombre de los mismos.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> Para mostrar se listan todos los idiomas existentes y se debe tener un paginado de manera que la lista visualizada no sea demasiado extensa. 	
<p>Prototipo de interfaz: No tiene, es a través de servicios web.</p>	

2.2.15 Historia de Usuario Guardar descargas

Historia de Usuario	
Número: 15	Nombre Historia de Usuario: Guardar descargas
Modificación de Historia de Usuario Número: 1	
Usuario: Yanirys Ortíz Lima	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas

<p>Descripción: Las descargas permiten al usuario obtener determinada versión de software. Estas incluyen identificador de la versión de software que se descargó, usuario que realizó la descarga y la fecha en que esta se realizó.</p>
<p>Observaciones:</p> <ul style="list-style-type: none"> • Para guardar la descarga son necesarios los siguientes datos: el usuario que realiza la descarga, el cual se encuentra autenticado en la plataforma y el identificador de la versión de software que se desea descargar.
<p>Prototipo de interfaz: No tiene, es a través de servicios web.</p>

2.2.16 Historia de Usuario Adicionar software a un listado

Historia de Usuario	
Número: 16	Nombre Historia de Usuario: Adicionar software a un listado
Modificación de Historia de Usuario Número: 1	
Usuario: Yanirys Ortíz Lima	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas
<p>Descripción: A los usuarios se le da la posibilidad de agregar 1 o varios software a las listas creadas por ellos.</p>	
<p>Observaciones:</p> <ul style="list-style-type: none"> • Para adicionar software a un listado se especifican las versiones de software a adicionar así como el listado que se desee llenar. 	
<p>Prototipo de interfaz: No tiene, es a través de servicios web.</p>	

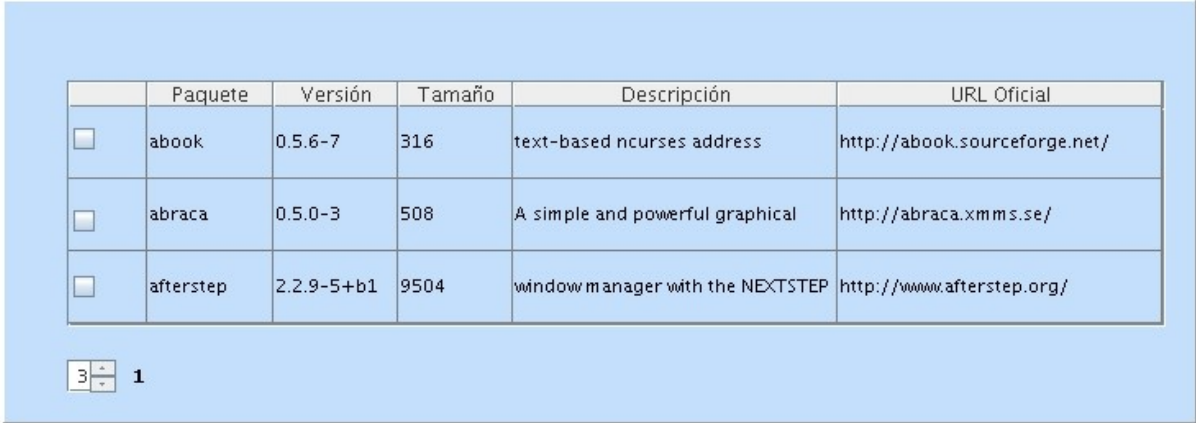
2.2.17 Historia de Usuario Importar paquetes de un repositorio

Historia de Usuario	
Número: 17	Nombre Historia de Usuario: Importar paquetes de un repositorio
Modificación de Historia de Usuario Número: 1	

Usuario: Odaimy Alvarez Acosta	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas
Descripción: A partir de los repositorios creados se permite importar paquetes, los cuales son guardados para luego ser añadidos al directorio como nuevas versiones de software.	
Observaciones: <ul style="list-style-type: none"> Para importar paquetes de un repositorio son necesarios los siguientes datos: identificador del repositorio del cual se importan los paquetes, así como todas las características que poseen los paquetes. 	
Prototipo de interfaz:	
<p><i>Figura 25: Importar paquetes de un repositorio</i></p>	

2.2.18 Historia de Usuario Mostrar paquetes de un repositorio

Historia de Usuario	
Número: 18	Nombre Historia de Usuario: Mostrar paquetes de un repositorio
Modificación de Historia de Usuario Número: 1	
Usuario: Odaimy Alvarez Acosta	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas

Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas
Descripción: Cada repositorio esta constituido por un grupo de paquetes. Los paquetes Incluyen el nombre del mismo, versión, tamaño, una breve descripción y la url oficial.	
Observaciones: <ul style="list-style-type: none"> Para mostrar paquetes de un repositorio se debe tener un paginado de manera que la lista visualizada no sea demasiado extensa. 	
Prototipo de interfaz:	
	
<p>Figura 26: Listar paquetes</p>	

2.2.19 Historia de Usuario Eliminar paquetes de un repositorio

Historia de Usuario	
Número: 19	Nombre Historia de Usuario: Eliminar paquetes de un repositorio
Modificación de Historia de Usuario Número: 1	
Usuario: Odaimy Alvarez Acosta	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0.5 semanas
Riesgo en Desarrollo: Bajo	Puntos Reales: 0.5 semanas
Descripción: Los paquetes son eliminados cuando se adicionan al directorio como una nueva versión de software o manualmente a partir de la lista de paquetes existentes.	
Observaciones:	

- Para **eliminar** se selecciona el paquete y se confirma la eliminación.

Prototipo de interfaz:

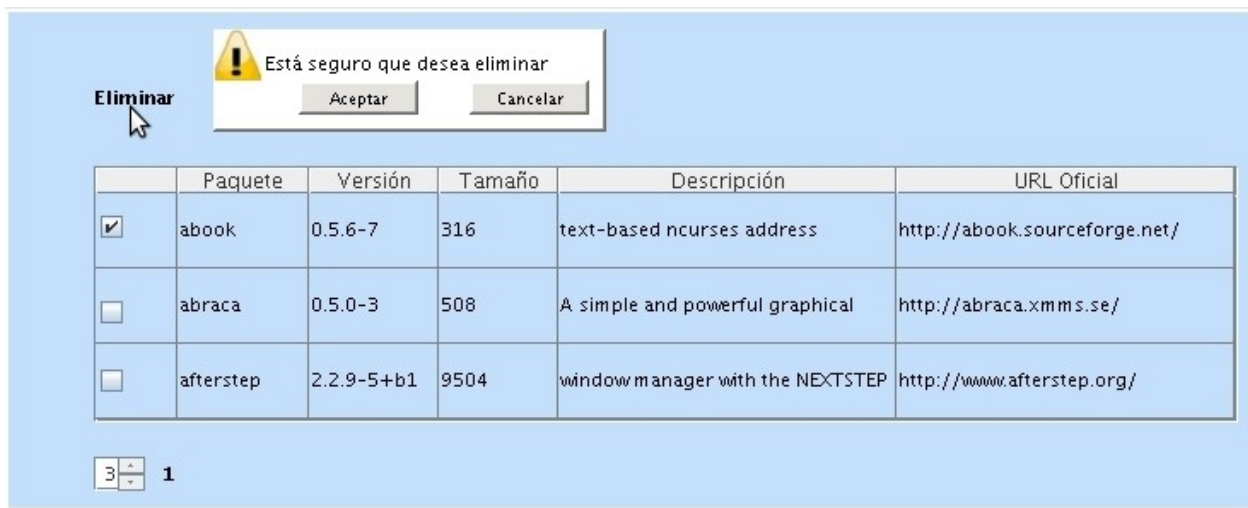


Figura 27: Eliminar paquetes de un repositorio

2.3 Plan de Releases

De acuerdo a las funcionalidades descritas en las historias de usuarios y en concordancia con la prioridad asignada para su realización, se planificaron 4 iteraciones que recogen el desarrollo del directorio de software.

Release	Descripción de la iteración	Orden de la HU a implementar	Duración total
1	Desarrollo de las Historias de Usuario de alta prioridad.	01, 02, 03, 04	6 semanas
2	Desarrollo de las Historias de Usuario que tienen prioridad media e integrar los componentes desarrollados en la iteración anterior.	08, 09, 10, 11, 12	4 semanas
3	Desarrollo de las Historias de Usuario de prioridad baja e integrar los componentes desarrollados en	05, 06, 07, 13, 14, 15, 16, 17, 18, 19, 20	3 semanas

	iteraciones anterior.		
4	Integrar el producto con el núcleo de la plataforma.	11	2 semanas

2.4 Conclusiones parciales

En este capítulo se identificaron los requerimientos que el sistema debe cumplir desglosados en Historias de Usuario y estableciéndose un orden de prioridad para su realización, además de ser asignadas a un desarrollador que para su cumplimiento debe llevar a cabo una serie de tareas o actividades. Se establecieron los ciclos iterativos en los que serán desarrolladas o mejoradas las funcionalidades de la solución propuesta.

Capítulo 3. Desarrollo del directorio de software e integración con la plataforma de migración

Este capítulo aborda los elementos de diseño y la arquitectura del directorio de software, describiendo el patrón arquitectónico Modelo-Vista-Controlador (MVC), en la variante implementada por el framework Symfony; así como los patrones de diseño empleados y los mecanismos de seguridad que provee el framework. Se muestra el modelo de diseño, se describe la estructura del directorio de software a través del diagrama de componentes, así como la estrategia utilizada para el diseño del modelo de datos y el estándar de código usado en la implementación. Se plantea todo lo referente a la creación del módulo de interfaz de usuario del directorio de software, mostrando el modelo de diseño del mismo.

3.1 Desarrollo del directorio de software

El directorio de software es un producto que funciona por sí solo, los servicios web que este brinda pueden ser consumidos por cualquier otro sistema que necesite las funcionalidades que este ofrece, característica conocida como interoperabilidad con otras aplicaciones.

3.1.1 Arquitectura del directorio de software

Para darle cumplimiento al problema planteado se creó un proyecto Symfony que consta de la siguiente aplicación:

frontend: Es el directorio de software en sí, la cual cuenta con un total de 9 módulos que recogen todas las funcionalidades requeridas por el cliente, brindando servicios web al Núcleo de la Plataforma de Migración a código abierto.

3.1.1.1 Modelo de diseño

Un modelo de diseño representa los requisitos funcionales en un lenguaje técnico lo más cercano posible al lenguaje de programación y sirve como una abstracción del código fuente. Este brinda una vista estática del sistema. "El modelo de diseño proporciona detalles acerca de las estructuras de datos, las arquitecturas, las interfaces y los componentes del software que son necesarios para implementar el sistema" [21].

En la metodología SXP no se enfatiza en la definición temprana de una arquitectura estable para el sistema, sino que esta se asume de forma evolutiva reducir los inconvenientes con dicha

arquitectura se utiliza el diseño con metáforas que da como resultado el modelo de diseño.

El diseño del directorio de software se ajusta a la arquitectura definida para la interacción con los demás subsistemas que componen la plataforma y la utilización del patrón arquitectónico MVC.

A continuación se representa como fue diseñado el sistema:

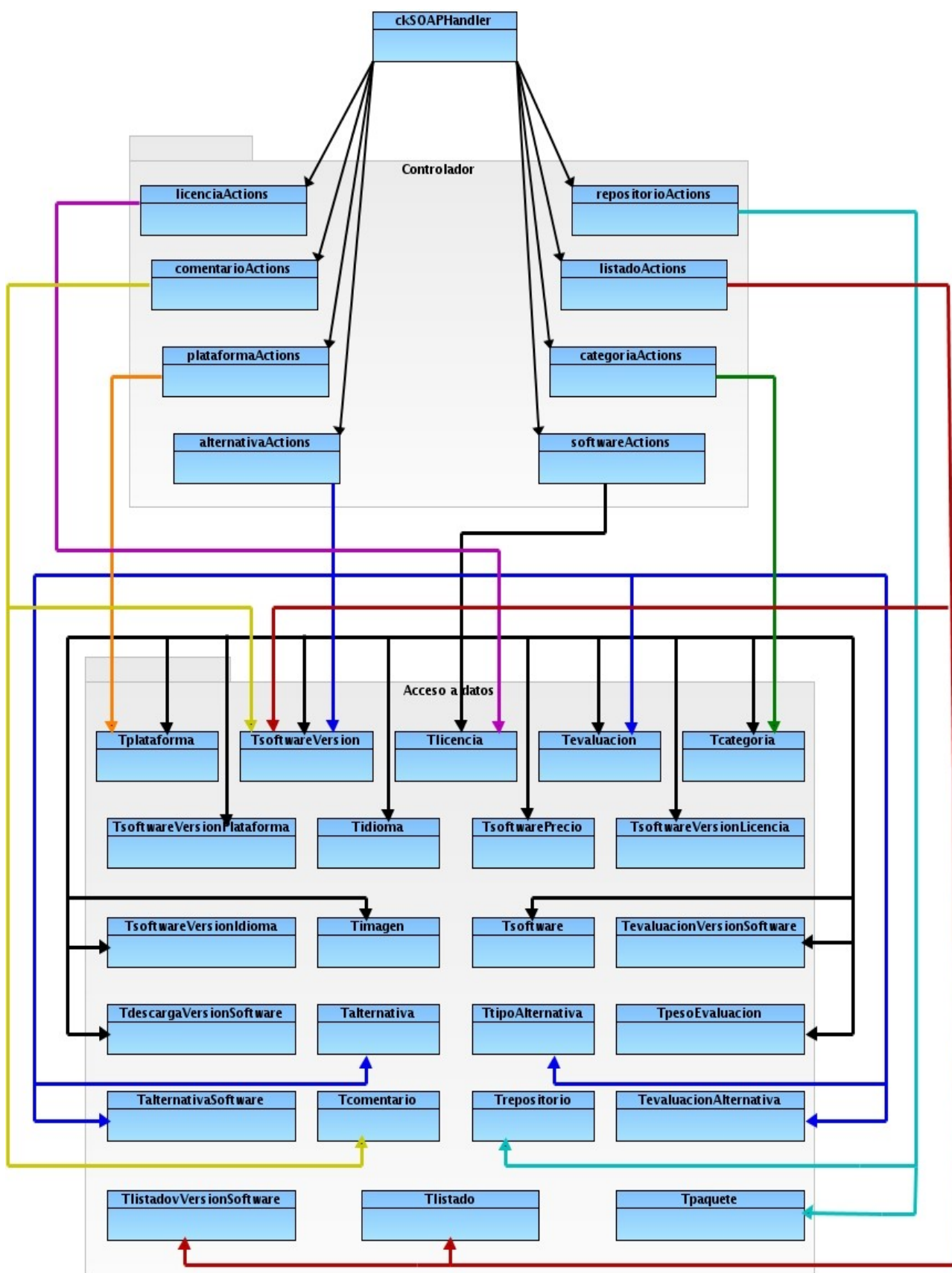


Figura 1: Modelo de diseño del directorio de software

3.1.1.2 Diagrama de componentes

Un diagrama de componentes representa la vista de implementación estática de un sistema. “La vista de implementación muestra el empaquetado físico de las partes reutilizables del sistema en unidades sustituibles, llamadas componentes. Una vista de implementación muestra la implementación de los elementos del diseño (tales como clases) mediante componentes, así como sus interfaces y dependencias entre componentes. Los componentes son las piezas reutilizables de alto nivel a partir de las cuales se pueden construir los sistemas” [22].

Los componentes y paquetes en el directorio de software mantienen una organización física estrechamente relacionada con la estructura lógica de este sistema. En la aplicación frontend se encuentran los archivos de configuración, así como todos los módulos y las librerías que estos necesitan y esta se relaciona con los ficheros propios del framework Symfony, así como con todos los plugins utilizados.

A continuación se describen los componentes de la solución propuesta, así como su interacción con otros componentes:

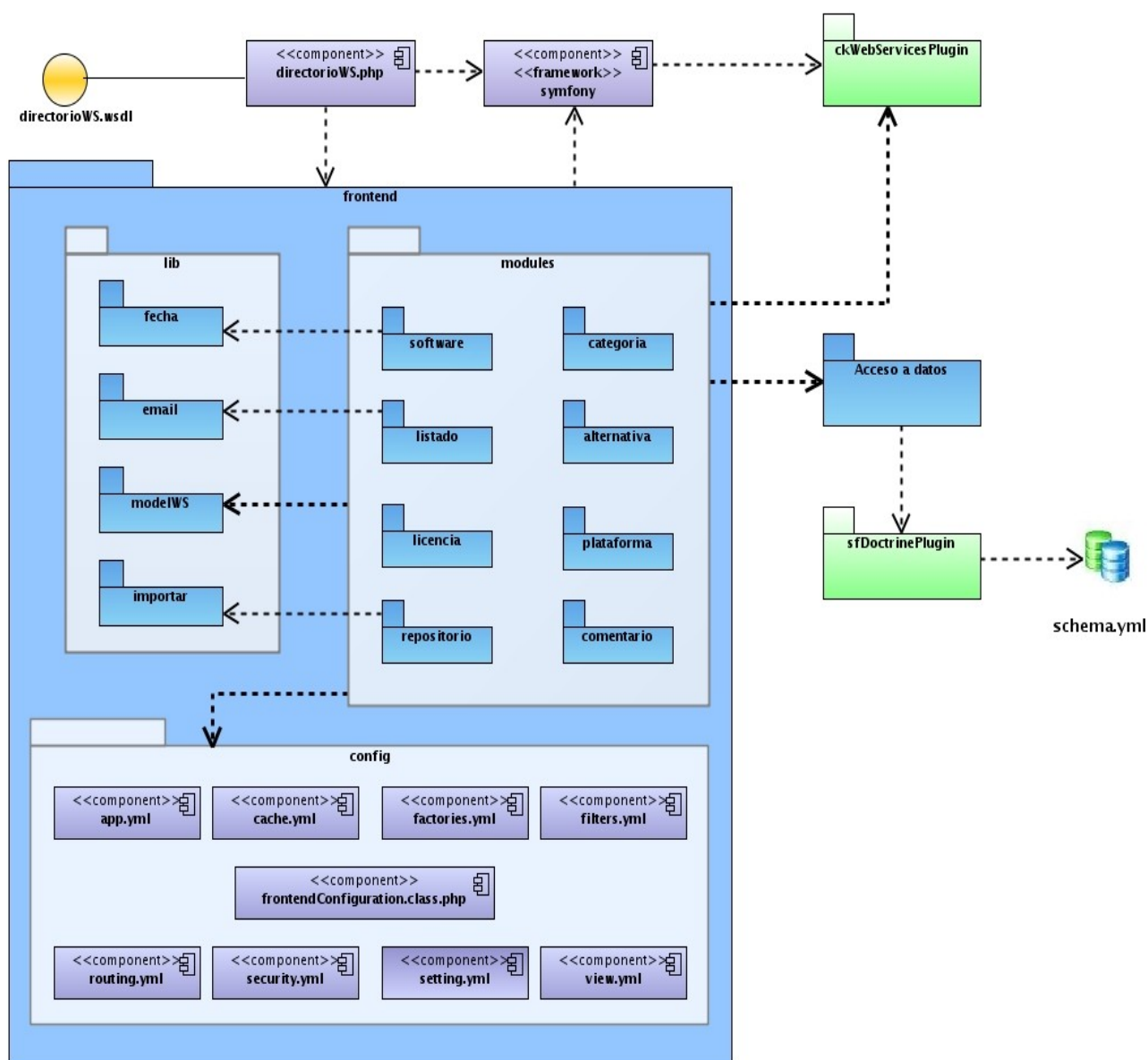


Figura 2: Diagrama de componentes del directorio de software

3.1.1.3 Esquema de la base de datos utilizada en la solución propuesta

A continuación se presenta el modelo de datos utilizado por la aplicación y la estrategia seguida para la obtención del mismo. Para la construcción del diagrama entidad-relación fue utilizada la herramienta CASE Visual Paradigm for UML versión 6.4. Luego se realizaron las transformaciones necesarias para obtener el modelo relacional, y por último se normalizó parcialmente (debido a requerimientos del cliente) el esquema obtenido hasta la tercera forma

normal. Posteriormente se escribió en el archivo `schema.yml` el esquema final que utiliza la librería Doctrine para la generación automática de las clases modelo y formularios y la inserción de la base de datos en el Sistema Gestor de Bases de Datos PostgreSQL (Ver Anexo II).

3.1.2 Estándar de codificación utilizado

Una de las buenas prácticas propuesta por las metodologías ágiles es la adopción de un estándar de codificación por parte del equipo de desarrollo; asegurando que el código exprese claramente el propósito del mismo y agilice el proceso de refactorización. Se trata de que los miembros del equipo lo apliquen y se sientan cómodos.

Para darle cumplimiento a la solución propuesta se escoge el estándar de código propuesto para el desarrollo de Symfony y se adicionan algunas reglas que hacen más legible el código escrito y con las que el equipo de desarrollo está familiarizado (Ver Anexo III).

3.2 Desarrollo del módulo de interfaz de usuario del directorio de software

Para darle cumplimiento a la propuesta de integración de los componentes de la plataforma, se crea el módulo de interfaz de usuario del directorio de software, este proporciona una vista sencilla e intuitiva donde se consumen los servicios web que el directorio de software ofrece que permiten la administración del mismo.

3.2.1 Requisitos para la integración.

Durante la entrevista con el cliente se acordó que es indispensable desarrollar la interfaz de administración del directorio de software y para ello se definieron los siguientes requisitos:

1. Crear interfaz sencilla e intuitiva desde la cual se acceda a todas las funcionalidades necesarias para la administración del directorio de software.
2. El intercambio de información debe ser mediante servicios web.
3. Debe ser implementada en lenguaje PHP.
4. La interfaz del directorio debe estar acorde con la identidad del proyecto SIMAYS.
5. Deben utilizarse tecnologías libres para el desarrollo del directorio.

3.2.2 Modelo de diseño

El diseño del módulo de interfaz del directorio de software traduce los requisitos definidos en una

representación estática del sistema, y para ello se hace uso de las clases definidas en el sistema de manejo de interfaces implementado para la integración de los sistemas de la Plataforma de Migración a Código Abierto.

Dentro del paquete *dirSW* se crea una clase por cada módulo al que se le va a construir la interfaz de usuario, así como un paquete de igual nombre que contendrá tantos ficheros PHP como funcionalidades tenga el módulo al cual está asociado cada clase.

A continuación se representa como fue diseñado el módulo de interfaz del directorio:

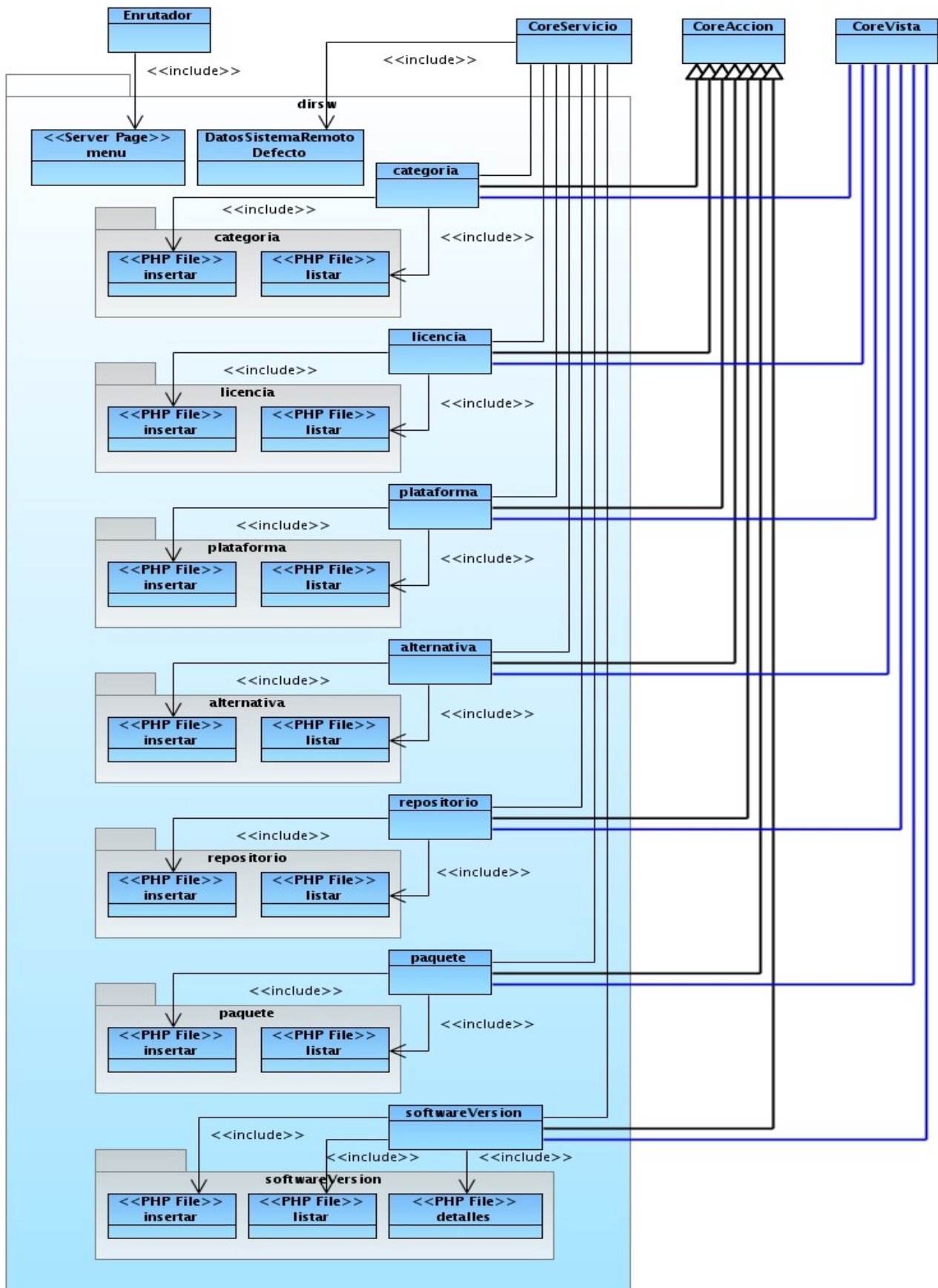


Figura 3: Diagrama de clases del diseño del módulo interfaz del directorio de software

3.3 Conclusiones parciales

En este capítulo se han descrito los elementos esenciales de la arquitectura del directorio de software, los componentes desarrollados y su interacción con los demás subsistemas que componen la plataforma. El sistema cumple con las funcionalidades requeridas por el cliente, basado en los patrones de diseño y el patrón arquitectónico que implementa el framework Symfony, además de los mecanismos de seguridad que este provee. Describe los servicios web que brinda el directorio de software; convirtiéndolo en una aplicación interoperable, capaz de compartir funcionalidades y datos con otras aplicaciones, independientemente del lenguaje de programación utilizado o plataforma sobre la cual estén desplegadas.

Capítulo 4. Análisis de los resultados

Las pruebas que se le realizan a un software son vitales para ganar en la seguridad y la calidad del mismo, además de encontrar errores previos a la entrega del sistema al usuario final, de modo que una vez entregado al cliente, el producto posea el menor número de inconvenientes posibles. Es recomendado realizar pruebas al software desde el mismo momento en que se implementa una nueva funcionalidad, logrando así un entorno de desarrollo satisfactorio.

Para la realización de las pruebas se debe tener en cuenta que:

- La prueba es un proceso de ejecución de un programa con la intención de descubrir errores.
- Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.
- Una prueba tiene éxito si descubre un error no detectado hasta entonces.

Existen varios tipos de pruebas que pueden ser realizadas a un sistema para así lograr la obtención de un producto funcional y fiable. Sin embargo, si se trata de desarrollo ágil con pruebas constantes, es interesante destacar la utilización de herramientas que automatizan esta actividad.

La automatización de pruebas es uno de los mayores avances en la programación desde la invención de la orientación a objetos. Concretamente en el desarrollo de las aplicaciones web, las pruebas aseguran la calidad de la aplicación incluso cuando el desarrollo de nuevas versiones es muy activo [20].

4.1 Pruebas funcionales

Las pruebas funcionales no solo validan la transformación de una entrada en una salida, sino que validan una característica completa [20]. Estas son necesarias para que la aplicación funcione como un todo y determinar si el producto desarrollado cumple con las funcionalidades requeridas por el cliente; específicamente en un ambiente web verifican que todos los enlaces funcionen correctamente, es decir, al hacer clic sobre vínculos y botones, se muestren las páginas y mensajes esperados. Como este proceso se hace manualmente, se torna aburrido y es muy propenso a cometer errores. Es por ello que Symfony incluye su propio framework de pruebas (Lime) además de una clase que representa un navegador web, capaz de automatizar estas tareas (subframework de pruebas funcionales). Estas pruebas simulan el funcionamiento del navegador, enviando peticiones y recibiendo las respuestas del servidor, asegurando que cada funcionalidad haga lo que se supone debe hacer.

4.1.1 Pruebas funcionales realizadas al sistema

Fueron diseñadas y ejecutadas un conjunto de pruebas funcionales a todos los módulos del directorio de software, enfatizando en las funcionalidades que no poseen interfaz. Las mismas permitieron corregir los errores detectados para así lograr el correcto funcionamiento del sistema. (Ver Anexo IV)

4.2 Impacto y aporte de la solución propuesta

Toda investigación o producto que posea como tarea principal solucionar determinado problema debe tener un impacto, ya sea social o económico. El impacto que este produce y aporte para el desarrollo de la humanidad demuestra la veracidad de la importancia que represente. Al término de la finalización de la presente investigación se obtuvo una aplicación web libre que automatiza la obtención de alternativas de una forma más eficiente, flexible y adaptable ante cambios. El sistema permite la gestión de software con sus versiones, las licencias de los mismos, así como las plataformas en que estos pueden funcionar correctamente. Con el uso de la aplicación obtenida el trabajo de los especialistas en migración se hace más fácil, ya que logran hacer su trabajo de una manera más sencilla.

Además el directorio de software brinda un conjunto de servicios web a través de los cuales el sistema es capaz de compartir datos y funcionalidades con otras aplicaciones, independientemente de su lenguaje de implementación o sistema operativo en el que están desplegadas.

4.3 Conclusiones parciales

Mediante la automatización de las pruebas funcionales que fueron realizadas al producto se facilita el proceso de desarrollo de una manera ágil y fiable, influyendo en la rápida corrección de errores en las funcionalidades implementadas. Garantizando la calidad y eficiencia de la aplicación realizada, cumpliendo con el levantamiento de requisitos.

Conclusiones

En la presente investigación se realizó un estudio del estado del arte de las herramientas de búsqueda y recuperación de información en la web, enfatizando precisamente en los directorios de materias y sintetizando las características que proveen.

Fueron analizados algunos directorios de software existentes para identificar y documentar sus principales características, este estudio permitió determinar la necesidad de desarrollar un directorio de software capaz de automatizar la búsqueda de alternativas libres y fuese lo suficientemente eficiente y adaptable ante cambios, para de esta manera lograr suplir una de las necesidades que existe en el proceso de migración a código abierto llevado a cabo en nuestro país, además de optimizar el empleo de tiempo y fuerza de trabajo.

Se realizó un análisis de la implementación de servicios web, cuestiones que posibilitaron apreciar el papel importante que juegan en la comunicación entre sistemas, así como las principales herramientas, tecnologías y lenguajes de programación usados en la construcción de aplicaciones web dinámicas. El proceso de desarrollo estuvo guiado por la metodología ágil SXP, se obtiene un producto completamente libre, a través de un ciclo de desarrollo ágil, permitiendo responder rápidamente a cambios en los requisitos.

Fueron implementadas todas las funcionalidades del sistema mediante servicios web, posibilitando la obtención de un producto que satisface las necesidades del cliente. Se desarrolló además el módulo interfaz de usuario del directorio para la administración del mismo. El diseño y ejecución de un conjunto de pruebas funcionales realizadas al culminar cada nueva funcionalidad, junto a la realización de los Casos de Pruebas de Aceptación al módulo interfaz del directorio aseguraron la calidad del producto final.

Recomendaciones

Al concluir la presente investigación, se recomienda:

- Validar el uso del directorio de software en un proyecto de migración real.
- Implementar una aplicación que permita a los usuarios que no trabajan en labores administrativas del directorio publicar comentarios, crear listados de software para recomendar a la comunidad y verificar las alternativas libres a las aplicaciones privativas.
- Agregar un buscador que automatice la búsqueda de software por varios criterios.

Referencias bibliográficas

- [1]. Dr. Cs. Carlos Alvarez de Zayas. METODOLOGIA DE LA INVESTIGACION CIENTIFICA. [SANTIAGO DE CUBA], 1995.
- [2]. BUSQUEDA Y RECUPERACIÓN DE INFORMACIÓN. Disponible en:
<http://www.mariapinto.es/e-coms/recu_infor.htm#ri1>.
- [3]. SeRQL y SPARQL - Recuperacion y organizacion de la informacion. Disponible en:
<<http://serqlsparql.50webs.com/>>.
- [4]. Metodologías de desarrollo de software. Disponible en:
<http://www.rhernando.net/modules/tutorials/doc/ing/met_soft.html>.
- [5]. Gladys Marsi Peñalver Romero, y Abel Meneses Abad. SXP, metodología ágil para proyectos de software libre. 2009.
- [6]. Procesador de hipertexto | Autorneto. Disponible en:
<<http://autorneto.com/tecnologia/software/procesador-de-hipertexto/>>.
- [7]. PHP: Historia de PHP - Manual. Disponible en:
<<http://www.php.net/manual/es/history.php.php>>.
- [8]. Oren Ben-Kiki, Clark Evans, y Brian Ingerson. YAML Ain't Markup Language (YAML TM) Version 1.1. Diciembre 2004.
- [9]. Guía Breve de Tecnologías XML. Disponible en:
<<http://www.w3c.es/divulgacion/guiasbreves/tecnologiasxml>>.
- [10]. Manual de JavaScript. Tutorial de JavaScript. WebEstilo. Disponible en:
<<http://www.webestilo.com/javascript/>>.
- [11]. Guía Breve de XHTML. Disponible en:
<<http://www.w3c.es/divulgacion/guiasbreves/XHTML>>.
- [12]. Javier Eguíluz Pérez. Introducción a CSS. Mayo 2009.
- [13]. Javier Eguíluz Pérez. Introducción a AJAX. Febrero 2008.
- [14]. Mohammed J, Kabir. Servidor Apache 2.
- [15]. Introducción a jQuery UI. Disponible en:
<<http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=jQueryUI>>.
- [16]. Ethan Cerami, Web Services Essentials, 2002.

- [17]. Visión General de Servicios Web XML. Disponible en:
<<http://es.scribd.com/doc/8739651/Vision-General-de-Servicios-Web-XML>>.
- [18]. R. Richards, "Web Distributed Data Exchange, XML-RPC, Representational State Transfer (REST), SOAP," Pro PHP XML and Web Services, Appress, 2006, pp. 567-750.
- [19]. Patrones de diseño. Análisis y Diseño. Ingeniería del Software. Disponible en:
<<http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php>>.
- [20]. A Gentle Introduction to symfony | Chapter 2 - Exploring Symfony's Code | symfony | Web PHP Framework. Disponible en: <http://www.symfony-project.org/gentle-introduction/1_4/en/02-Exploring-Symfony-s-Code>.
- [21]. Roger S. Pressman, y Ph. D. Software Engineering a practitioner's approach. 7ma.
- [22]. J. Rumbaugh, I. Jacobson, and G. Booch, "Vistas Físicas," El Lenguaje Unificado de Modelado. Manual de Referencia, Addison Wesley.

Bibliografía

- [1]. BUSQUEDA Y RECUPERACIÓN DE INFORMACIÓN. Disponible en:
<http://www.mariapinto.es/e-coms/recu_infor.htm#ri1>.
- [2]. Indización y Clasificación de hipertextos. Disponible en:
<<http://www.hipertexto.info/documentos/indizacion.htm#listas>>.
- [3]. Apuntes. Ingeniería del software. Sistemas Informaticos. Nivel de madurez software. Informatica Aplicada a la Gestión Pública. 2005/06-2. Universidad de Murcia. Rafael Barzanallana. Disponible en: <<http://www.um.es/docencia/barzana/IAGP/lagp2.html>>.
- [4]. Utilizando Framework de programación en las Empresas - QDiario AplicacionesEmpresariales.com. Disponible en:
<<http://www.aplicacionesempresariales.com/utilizando-framework-de-programacion-en-las-empresas.html>>.
- [5]. PgAdmin III - Guía Ubuntu. Disponible en: <http://www.guia-ubuntu.org/index.php?title=PgAdmin_III>.
- [6]. UML CASE tool for software development. Disponible en: <<http://www.visual-paradigm.com/product/vpuml/>>.
- [7]. PostgreSQL: The world's most advanced open source database. Disponible en:
<<http://www.postgresql.org/>>.
- [8]. Guía Breve de Servicios Web. Disponible en:
<<http://www.w3c.es/divulgacion/guiasbreves/ServiciosWeb>>.
- [9]. XML-RPC Home Page. Disponible en: <<http://www.xmlrpc.com/>>.
- [10]. symfony | Web PHP Framework. Disponible en: <<http://www.symfony-project.org/>>.
- [11]. SOA - Web Services — ibermatica.com. Disponible en:
<<http://www.ibermatica.com/ibermatica/integracionssii/soa>>.
- [12]. Zend Framework: Documentation: Zend_Search_Lucene - Zend Framework Manual. Disponible en: <<http://framework.zend.com/manual/en/zend.search.lucene.html>>.
- [13]. Plugins | ckWebServicePlugin | 4.0.0 | symfony | Web PHP Framework. Disponible en:
<<http://www.symfony-project.org/plugins/ckWebServicePlugin>>.
- [14]. PHP: Historia de PHP - Manual. Disponible en:

- <<http://www.php.net/manual/es/history.php.php>>.
- [15]. Introducción a PHPMailer | unijimpe. Disponible en: <<http://blog.unijimpe.net/introduccion-a-phpmailer/>>.
- [16]. Visión General de Servicios Web XML. Disponible en: <<http://es.scribd.com/doc/8739651/Vision-General-de-Servicios-Web-XML>>.
- [17]. jQuery UI - Home. Disponible en: <<http://jqueryui.com/>>.
- [18]. Las pruebas en aplicaciones Web, conceptos y clasificación | Zona Webmaster. Disponible en: <<http://www.web-mix.ws/webmaster/las-pruebas-en-aplicaciones-web-conceptos-y-clasificacion-fuente-las-pruebas-en-aplicaciones-web-conceptos-y-clasificacion/>>.

Glosario de Términos

Framework: Es un marco de trabajo que expone un conjunto de librerías que no forman parte de una aplicación específica y que han sido creadas para ser utilizados por cualquier aplicación. Sus componentes pueden ser usados para propósitos generales y están ampliamente probados para funcionar en gran variedad de escenarios.

Helpers: Funciones simples que empaquetan bloques de código con el fin de hacer más legible y sencillo.

Indizar: Comprende la tarea de catalogar y describir los recursos a través de palabras clave que forman parte de un índice terminológico o vocabulario controlado.

Internet: Conjunto descentralizado de redes de comunicación interconectadas, garantizando que las redes físicas que la componen funcionen como una red lógica única, de alcance mundial.

Metadato: Los metadatos son datos altamente estructurados que describen información de otros datos.

Ontologías: En informática hace referencia a la formulación de un exhaustivo y riguroso esquema conceptual dentro de uno o varios dominios dados; con la finalidad de facilitar la comunicación y el intercambio de información entre diferentes sistemas y entidades.

ORM (Object Relational Mapping): Es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional.

Plugin: Es una aplicación que se integra con otra para aportarle una función nueva y generalmente muy específica.

Release: Se refiere a una versión funcional de un producto software.

Refactorización: Reestructuración y reutilización de un código previamente escrito, alterando su estructura interna sin cambiar su comportamiento externo.

Software: Programas del ordenador. Se refiere a instrucciones que se ejecutan por medio de un equipo de cómputo. Comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas.

Tesauros: Lista que contiene los "términos" empleados para representar los conceptos, temas o contenidos de los documentos, con miras a efectuar una normalización terminológica que permita mejorar el canal de acceso y comunicación entre los usuarios y las Unidades de Información.

UML (Unified Modeling Language): Lenguaje Unificado de Modelado, es un lenguaje de modelado de sistemas de software. Permite la especificación, visualización, construcción y documentación de elementos de la Ingeniería del Software.

W3C (World Wide Web Consortium): Organismo internacional que tiene por fin establecer normas para el desarrollo y uso de la web.