Universidad de las Ciencias Informáticas

"Facultad 5"

Título: Plataforma colaborativa para la gestión de activos de tipo Media.

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

Autores:

Naidiley Vargas Pérez.

Leonarkis Acosta Suárez.

Tutor:

Ing. Jaime González Campistruz.

Co-Tutor:

Ing. Julio César Espronceda Pérez.

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 ____.

Naidiley Vargas Pérez

_____

Firma del Autor

Leonarkis Acosta Suárez

_____

Firma del Autor

Ing. Jaime Campistruz

_____

Firma del Tutor

Ing. Julio César Espronceda Pérez

_____

Firma del Co-Tutor

DATOS DE CONTACTO

Tutor: Jaime González Campistruz.

Graduado de: Ingeniero en Ciencias Informáticas.

Años de experiencia: 7 años.

Teléfono de contacto: 837 3173.

E-mail: jgonzalezc@uci.cu

Co-Tutor: Julio César Espronceda Pérez.

Graduado de: Ingeniero en Ciencias Informáticas.

Años de experiencia: 2 años.

Teléfono de contacto: 837 2745.

E-mail: jcespronceda@uci.cu

AGRADECIMIENTOS

A la Revolución, por haberme brindado la posibilidad de estudiar en esta universidad donde me formé como profesional y poder ostentar hoy, el título de Ingeniera en Ciencias Informáticas.

A todos los profesores que han contribuido en mi formación durante los cinco años, transmitiéndome sus conocimientos.

A mis tutores por sus consejos y asesoramiento que hicieron posible la realización de este trabajo.

A todos mis amigos, compañeros, conocidos que compartieron conmigo momentos buenos y malos durante mi estancia en la universidad. En especial esos amigos que cuando de verdad los necesitas están ahí, gracias Crespo por estar incondicionalmente cuando lo necesité.

A mi otra gran familia. Mi suegra, mi cuñada, que han sido siempre tan buenas conmigo.

A mi novio y compañero de tesis, por malcriarme, por estar conmigo en todo momento y darme ánimos cuando lo necesité.

A mi familia, especialmente a mi abuelita, a mi hermano y a mis padres, por ser los principales responsables de lo que he logrado en la vida.

Naidiley

Agradecimientos

A la Revolución y a la UCI por darme la oportunidad de materializar mis sueños y formarme profesionalmente.

A todos los profesores que me han apoyado y me han aportado sus conocimientos a lo largo de la carrera.

A mis tutores por el apoyo y la confianza que depositaron en mí.

A todos mis compañeros, los que me acompañaron en esta trayectoria de aprendizaje y conocimiento, compartiendo buenos y malos momentos.

A mi otra familia por recibirme siempre con los brazos abiertos y ser para mí unos padres más.

A mi novia, compañera de tesis y la mujer que amo, por luchar conmigo por el mismo sueño y ser siempre tan incondicional.

A mi familia, especialmente a mis padres, mi abuela, mis tíos maternos y mi hermana, por todo el amor y la ayuda que me han brindado.

Leonarkis

A todas las personas que de una forma u otra nos apoyaron en la elaboración de este trabajo incluyendo al tribunal por todas sus sugerencias, les agradecemos de todo corazón. A todos Muchas Gracias.

DEDICATORIA

A mi abuelita, por apoyarme en todo momento, darme buenos consejos y transmitirme esa mente positiva ante todas las adversidades.

A mi hermanito, que siempre lo tengo presente y lo quiero tanto.

A mi papá, por toda la ayuda y la confianza que siempre me ha brindado en la lucha por alcanzar mi sueño.

A mi mamá, por su preocupación y constante cuidado sobre mí, por ser mi alma gemela y la mejor madre del mundo.

Naidiley

A mi abuela, por todo el amor que me ha dado.

A mi hermanita, que aún siendo más joven que yo, ha hecho mucho por mí.

A mi mamá porque creyó en mí siempre, por ser mi amiga incondicional y por haber vivido este sueño cada segundo junto conmigo.

A toda mi familia por apoyarme y haber confiado en mí.

Leonarkis

RESUMEN

El número creciente de información disponible a través de internet hace necesarias herramientas para la selección de los datos a los cuales se desea acceder. Por otra parte la posibilidad de compartir datos e información permite el desarrollo de sistemas para el soporte de la colaboración entre usuarios. El presente proyecto denominado Plataforma colaborativa para la gestión de activos de tipo Media consiste en una aplicación web que implementa un sistema de recuperación de información para la manipulación, búsqueda, organización e incremento del contenido de activos de tipo media (modelos, texturas, audios y videos) que se genera en el proceso de producción de software de realidad virtual del centro VERTEX- Entornos Interactivos 3D, que pertenece a la facultad 5 de la Universidad de las Ciencias Informática (UCI). Esta solución permite la administración, centralización y reutilización de artefactos de diseño. Además el desarrollo de los productos de diseño no se verá comprometido por la falta de un servidor de media profesional.

Palabras clave: activos de tipo Media, plataforma colaborativa, productos de diseño, VERTEX.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El desarrollo acelerado de las Tecnologías de la Informática y las Comunicaciones (TIC) ha influido de forma positiva en cada esfera de la sociedad. Llevar a cabo la informatización en diferentes sectores de la actividad humana ha sido una labor en la cual se han realizado intensos esfuerzos, con el objetivo de lograr una mejor gestión y control de la información que se genera en una determinada entidad.

El uso de la informática en la actividad diaria del hombre ha permitido que los grandes volúmenes de información que se manejan hayan sido digitalizados y que muchos de los procesos actuales se ejecuten de forma automática. El desarrollo acelerado de las TIC ha conducido a grandes reestructuraciones en los espacios de trabajo, donde la producción masiva de documentación, la necesidad de acceder y gestionar la misma son planes fundamentales en el desarrollo de la ciencia y la tecnología.

Esta abundancia informativa ha impulsado el desarrollo de aplicaciones informáticas que faciliten y proporcionen la gestión de la misma. El número creciente de información hace necesarias herramientas para la selección de los datos a los cuales se desea acceder. Por otra parte la posibilidad de compartir datos y archivos permite el desarrollo de sistemas para el soporte de la colaboración entre usuarios. Dichas herramientas son identificadas como plataformas de recuperación de información colaborativa (1), estas han influido gradualmente en la automatización, manipulación, búsqueda y organización de la información en las distintas esferas de desarrollo, según sus necesidades.

Por consiguiente, se hace indispensable la creación de sistemas de recuperación de información que aceleren el proceso de informatización en las diferentes empresas cubanas de manera que la sociedad pueda ir logrando insertarse con gran velocidad en la era de la información y el conocimiento. Cuba se inserta y avanza en el mundo de la informática, puesto que se ha visto inmersa en este desarrollo imparable de la informatización. Tanto es así que se han informatizado todos los centros educacionales, científicos y de trabajo de forma tal que casi toda la información se ha logrado almacenar de forma digital. Esto ha provocado la necesidad de crear sistemas de recuperación para esta información.

La Universidad de las Ciencias Informáticas se incluye dentro del proceso de informatización del país y promueve toda labor social y cultural acerca de las TIC. Esta universidad ha alcanzado un gran prestigio en todo el mundo, debido a que forma profesionales altamente capacitados y contribuye a potenciar el desarrollo del software cubano. Como parte de la estrategia para dar cumplimiento a sus objetivos se crearon varios centros especializados para el desarrollo de software en diferentes líneas de trabajo. El centro VERTEX- Entornos Interactivos 3D que pertenece a la facultad 5 de esta universidad, tiene como tarea el

desarrollo de software de Realidad Virtual y Entornos 3D. Este centro cuenta con un grupo de diseñadores encargados del diseño visual de los productos en dos dimensiones (2D) y en tres dimensiones (3D). Actualmente el proceso de producción del grupo de diseñadores presenta ciertas deficiencias que se mencionan a continuación:

- Los artefactos generados por este grupo tales como modelos, texturas, videos y audios se encuentran dispersos en sus distintos puestos de trabajo ya que no existe un repositorio centralizado para almacenar dichos resultados; esto provoca además que no se puedan reutilizar en un nuevo proyecto.
- La pérdida de tiempo al tener que destinar varios recursos humanos o el mismo varias veces en la elaboración de un artefacto común.
- No hay seguridad con los activos de diseño puesto que se encuentran únicamente en la máquina del usuario que creó el producto, si la computadora presenta algún problema técnico que no permita recuperar la información almacenada los artefactos pueden perderse.
- Existe gran lentitud en el proceso de solicitud de un producto de diseño a un diseñador, porque es personal la mayoría de las veces, o a través de un proceso jerárquico.

Dados estos antecedentes es necesario contar con una herramienta que cumpla con las expectativas y tendencias actuales en cuanto a tratamiento y gestión de la información se refiere y que brinde a sus usuarios una interfaz simple y agradable que facilite la gestión de activos de tipo media (modelos, texturas, videos y audios).

Ante la **situación problemática** anteriormente descrita se define como **problema de la investigación**:

- ¿Cómo contribuir a la centralidad y seguridad de la información en el proceso de gestión de activos de tipo media en el centro VERTEX?

A partir de este problema se establece como **objeto de estudio** del trabajo:

- Proceso de gestión de activos de tipo media.

Se propone como **campo de acción**:

- Herramientas de gestión de activos de tipo media en el centro VERTEX.

Para darle solución al problema formulado se plantea como **objetivo general**:

- Desarrollar una plataforma colaborativa que contribuya a mejorar la centralidad y seguridad de la información en el proceso de gestión de activos de tipo media en el centro VERTEX.

Para dar cumplimiento al objetivo de la investigación se definieron las siguientes **tareas investigativas:**

- Elaboración del marco teórico de la investigación a partir del estado del arte existente sobre el tema.

- Selección de metodología, herramientas de software a utilizar y lenguajes de programación para el desarrollo del sistema informático con calidad.
- Identificación de los requisitos de software para determinar las funcionalidades y restricciones que el sistema debe tener.
- Puntualización del diseño del software para la definición de la arquitectura.
- Implementación de los elementos necesarios para cumplir con las funcionalidades identificadas.
- Valoración y prueba de los resultados obtenidos para asegurar la calidad de la herramienta.

En la investigación se esperan como posibles resultados:

1. Catalogación de activos de tipo Media (visualización de activos de media en vista previa).
2. Gestión de comentarios de los activos.
3. Motivación de los implicados, haciendo uso de elementos motivacionales.
4. Posea un módulo de Administración de usuarios.
5. Posea un módulo de Administración de activos de media (modelos, videos, texturas y audios).
6. Posea un módulo de Reportes.

Para obtener los conocimientos necesarios con la finalidad de hacer posible el cumplimiento del objetivo trazado en el trabajo, se llevó a cabo una investigación en la que se utilizaron algunos de los métodos científicos existentes, tanto teóricos como empíricos.

Los métodos teóricos son aquellos que permiten estudiar las características del objeto de investigación que no son observables directamente, contribuyendo al desarrollo de las teorías científicas.

Métodos Teóricos:

- Analítico-sintético: Se utiliza con el objetivo de analizar y aumentar los conocimientos respecto al tema en cuestión, a partir de consultar la bibliografía científica correspondiente, para después, haciendo uso de la síntesis, lograr resumir y exponer los elementos que se relacionan con el proceso de gestión de activos de tipo media (modelos, texturas, videos, audios).
- Análisis Histórico-Lógico: Permite describir cómo ha evolucionado el desarrollo de las plataformas colaborativas en el proceso de gestión de activos de tipo media (modelos, texturas, videos, audios), y cómo se han incrementado las funcionalidades en los mismos.

Los métodos empíricos, representan un nivel de investigación cuyo contenido procede de la experiencia y es sometido a cierta elaboración racional.

Métodos Empíricos:

- Entrevista: Se realizaron entrevistas a varios trabajadores del centro VERTEX para tener un mayor conocimiento sobre la situación problemática existente. Para comprender cómo se lleva a cabo el proceso de gestión de los activos de diseño desarrollado por el personal. Además para verificar que el sistema cumpla con los requisitos propuestos por el cliente.
- Observación: Utilizado con la finalidad de caracterizar y analizar detalladamente cómo se realiza actualmente el proceso de gestión de activos de tipo media (modelos, texturas, videos, audios), identificando los mecanismos para desarrollar un sistema acorde con las necesidades. Se realizó un estudio sobre plataformas colaborativas con el objetivo de conocer funcionalidades y características de las mismas para lograr la implementación de un sistema que cumpla con los estándares de desarrollo establecidos. Utilizada también para contrastar los resultados de la entrevista.
- La consulta de información en todo tipo de fuentes: Utilizado para llevar a cabo la elaboración del marco teórico de la investigación, así como para abordar sobre las herramientas, lenguajes y metodología a utilizar.

La presente investigación está estructura en tres capítulos, los cuales se detallan a continuación:

Capítulo 1: "Fundamentación teórica". En este capítulo se plantean los elementos teóricos que sustentan la investigación, especificando los conceptos que permitan un mejor entendimiento de la situación problemática planteada. Se realiza un estudio y análisis de las tecnologías y las herramientas existentes para el desarrollo de aplicaciones web. Se describen y definen los lenguajes de programación y la metodología que guiará el proceso de desarrollo del software.

Capítulo 2: "Características y diseño del sistema". En este capítulo se abordan las fases de Exploración, Planificación e Iteraciones, todas estas etapas propias de la metodología de desarrollo utilizada. Se describen los artefactos obtenidos según la metodología. Se realizan las historias de usuario y se especifican los requisitos funcionales y no funcionales que deberá cumplir la solución. También se definen los patrones de diseño y el patrón de arquitectura a utilizar en el desarrollo de la solución.

Capítulo 3: "Implementación y prueba". En este capítulo se abordan los aspectos relacionados con la implementación del sistema, donde se incluyen los diagramas de despliegue. Además se valida la solución propuesta mediante las pruebas de aceptación.

1. CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

En este capítulo se describen los fundamentos teóricos que sustentan la presente investigación y se abordan términos que servirán para lograr una mejor comprensión del trabajo. También se realiza un estudio del marco de trabajo, ofreciendo una descripción de las principales características de los lenguajes, tecnologías, herramientas y metodología de desarrollo de software que se utilizarán en la implementación de la plataforma propuesta, con el propósito de seleccionar la mejor opción para el desarrollo de dicho sistema. Se realiza además un análisis de otras soluciones existentes con características similares al prototipo que se pretende desarrollar.

1.1. Conceptos asociados al dominio del problema

Con el avance de las Tecnologías de la Información y las Comunicaciones está el surgimiento de varios conceptos asociados a dichas tecnologías. El término **plataforma** forma parte de los conceptos antes mencionados, dichas plataformas *"son básicamente soportes de contenido, que administran la entrega de información e incorporan recursos como el chat, las listas de interés y similares"* (2). También son conocidas como una combinación de hardware y software usada para ejecutar aplicaciones; en su forma más simple, consiste únicamente de un sistema operativo, arquitectura, o una combinación de ambos (3).

Las tecnologías disponibles para el intercambio de información permiten el desarrollo de sistemas de colaboración en los cuales es posible atender requerimientos tanto individuales como de grupo, de manera tal que un usuario no se vea obligado a participar activamente en un entorno de colaboración. *"Los entornos de trabajo colaborativo podrían ser utilizados antes de entrar a definir una plataforma específica"* (2).En esta solución se propone la adición del factor de colaboración en el proceso de recuperación de activos de tipo media, sin que esto implique incrementar el esfuerzo cognitivo necesario para el uso del sistema propuesto.

En el Diccionario de la Lengua Española se define **colaboración** como la *"realización conjunta de un trabajo o tarea*" (4). El término antes mencionado se conoce además como el "*apoyo o contribución al logro de un objetivo*" (5).

Según los conceptos de plataforma y colaboración planteados anteriormente, se define como **plataforma colaborativa** a la herramienta informática que centraliza todas las funcionalidades ligadas a la conducción de un proyecto, la gestión de conocimientos y el funcionamiento de una organización, poniendo las mismas a disposición de los diferentes actores involucrados. La presente investigación consiste en la construcción

de una plataforma colaborativa que permita centralizar y administrar los activos de tipo media que se generan en el proceso de producción de software de realidad virtual del centro VERTEX. Por esta razón se requiere del estudio de estos sistemas, para de ser posible, utilizar funcionalidades similares y tener conocimiento sobre que posibles herramientas y tecnologías utilizar en la aplicación.

Los instrumentos o herramientas que suelen integrar dichas plataformas son (6):

- Herramientas del tipo foro digital, páginas de discusión, chats: las mismas permiten una mayor comunicación e interacción entre los usuarios, posibilitan además consolidar bases de conocimiento para consulta y apoyo técnico. Se utilizan foros de discusión para organizar la información que puede ser consultada por todos los visitantes.
- Sistema que permita compartir y gestionar archivos y recursos (cliente-servidor): disponen de un sistema de almacenamiento y gestión de archivos que permite realizar operaciones básicas sobre ellos, como visualizarlos, organizarlos en carpetas (directorios) y subcarpetas, copiar, pegar, eliminar, comprimir, descargar o cargar archivos en el sistema. La herramienta tiene la capacidad de integrar en un mismo entorno todos los recursos con los que se trabaja, permitiendo que los usuarios puedan reutilizar archivos creados por otros, con esto se logra ampliar los niveles de colaboración.
- Sistema de archivo colectivo, y de páginas digitales personales: El primero permite que todos los recursos compartidos estén disponibles para los usuarios. El segundo permite que cada usuario disponga de una zona privada para administrar (cargar, publicar, eliminar) sus archivos.
- Sistema de calificación (rating) y/o de votación: Este instrumento permite que los usuarios califiquen o evalúen un archivo, en dependencia del impacto que tenga el elemento a calificar. Un sistema de votación es otra forma de ver el nivel de aceptación que tiene un objeto o fichero en los individuos.
- *Weblogs* o equivalentes, por proyecto y/o por temática: El *weblog* es una publicación en línea de historias publicadas con una periodicidad muy alta, que son presentadas en orden cronológico inverso, es decir, lo más reciente que se ha publicado es lo primero que aparece en la pantalla. Es muy frecuente que los *weblogs* dispongan de una lista de enlaces a otros *weblogs*, a páginas para ampliar información, citar fuentes o hacer notar que se continúa con un tema que empezó otro *weblog.* También suelen disponer de un sistema de comentarios que permiten a los lectores establecer una conversación con el autor y entre ellos acerca de lo publicado.
- Calendario: Incluye un calendario de eventos y actividades de tipo personales, compartidas y globales.

- Reportes estadísticos: Se incluyen reportes detallados de acceso a la herramienta (historia, cantidad y frecuencia) y de participación en los diversos espacios de la aplicación.
- Aprobación de contenidos: Existen mecanismos centralizados de regulación y aprobación de la información (bajo responsabilidad del rol administrador).
- Manejo de roles y privilegios: Permite definir perfiles, roles y grupos, y asociar permisos de acceso a distintas herramientas y espacios.
- Autenticación: Requiere autenticación mediante contraseña y un cierre de sesión seguro (*logout*). Permite integración con sistemas de autenticación existentes (LDAP, Kerberos, NIS)
- Un índice o base-listado con las tareas pendientes y con las ya cumplidas.
- Base de conocimientos estructurados conteniendo guías o métodos de trabajo en grupo, para mejorar la comunicación, la producción, y la coordinación al interior del grupo.

Características de las plataformas colaborativas (2):

- Una característica fundamental debe de ser la interactividad. Esta es determinante para lograr maximizar los resultados y promover la colaboración. Interacción implica una acción recíproca de tal manera que cada participante pueda comunicarse con su semejante con vista de alcanzar un objetivo común.
- Una interfaz amigable y clara, que permita generar en el usuario la confianza necesaria para ubicarse con facilidad en todas las áreas que conforman la plataforma y crear un ambiente orientado al aprovechamiento de los contenidos y alcanzar los objetivos previstos.
- Un manejo ágil de las inscripciones y perfiles de entrada de los usuarios del sistema.
- Requerimientos mínimos del sistema, de modo que el acceso se haga sin mayor dificultad desde cualquier computadora. Esto implica entornos compatibles con diferentes navegadores y disponibilidad de equipos actualizados.
- Ambiente de trabajo colaborativo en el que se reflejen aspectos como confianza, compromiso, comunicación, coordinación, y complementariedad.
- Entorno en el cual todos los participantes del proyecto trabajan, colaboran, y se ayudan, para la realización del proyecto.

1.1.1. Gestión

El concepto de gestión hace referencia a la acción y al efecto de gestionar o administrar. A través de una gestión se llevarán a cabo diversas diligencias, trámites, las cuales conducirán al logro de un objetivo determinado.

En términos generales, *"gestión es la capacidad de una institución u organización para definir, alcanzar y evaluar sus propósitos administrando los recursos disponibles"* (7).

Gestionar es realizar diligencias conducentes al logro de un negocio o de un deseo cualquiera. El término gestión por lo tanto implica al conjunto de trámites que se lleva a cabo para resolver un asunto o concretar un proyecto. La gestión es también la dirección o administración de una empresa o de un negocio (8).

De acuerdo con lo expresado anteriormente, se define como gestión, a la actividad profesional que permite organizar el trabajo y los recursos que se posee, planificar tareas y comprobar la evolución de una labor.

1.1.2. Media

"Los medios de comunicación son un instrumento o forma de contenido por el cual se realiza el proceso comunicacional o comunicación. Algunos de los principales medios de comunicación como la televisión y la radio transmiten datos (generalmente ficheros de audio y video) mediante los cuales se establece dicha comunicación con las personas. Por lo que se define como media a todo el contenido de tipo video y sonido usado por los medios de trasmisión para divulgar la información" (9).

1.1.3. Activo de Tipo Media

Se denomina activo de tipo media a los artefactos producidos durante el desarrollo de un software de realidad virtual y entornos 3D (modelos, texturas, videos y audios), que son potencialmente reutilizables. Se pueden llamar además como una colección de partes de un software 2D o 3D, que se estructuran y modelan de una manera ordenada para producir los productos de una línea de productos.

1.1.4. Seguridad

"La seguridad informática consiste en garantizar que los recursos del sistema de información (material informático o programas) de una organización sean utilizados de la manera que se decidió y que el acceso a la información allí contenida así como su modificación solo sea posible a las personas que se encuentren acreditadas y dentro de los límites de su autorización" (10).

"A grandes rasgos se entiende que mantener un sistema seguro consiste básicamente en garantizar tres aspectos: confidencialidad, integridad y disponibilidad. Algunos estudios integran la seguridad dentro de una propiedad más general de los sistemas, la confiabilidad, entendida como el nivel de calidad del servicio ofrecido. Consideran la disponibilidad como un aspecto al mismo nivel que la seguridad y no como parte de ella, por lo que dividen esta última en solo las dos facetas restantes, confidencialidad e integridad" (10) .

Se dice que un sistema es confidencial cuando es accedido únicamente por los individuos autorizados a él; la integridad significa que los objetos solo pueden ser modificados por elementos autorizados, y de una manera controlada, y la disponibilidad indica que los objetos del sistema tienen que permanecer accesibles a elementos autorizados.

La información de una entidad determinada se puede ver afectada por muchos factores, incidiendo básicamente en los aspectos de confidencialidad, integridad y disponibilidad de la misma.

Desde el punto de vista del usuario, uno de los problemas más importantes puede ser el que está relacionado con el delito o crimen informático, por factores externos e internos. Una persona no autorizada podría: clasificar y desclasificar los datos, filtrar, borrar o alterar documentos, hojear información clasificada, entre otros (10).

1.1.5. Catalogación

"Teniendo en cuenta nuevamente el concepto de la Real Academia de la Lengua Española que conceptualiza catalogación como la acción de apuntar, registrar ordenadamente o clasificar un objeto según normas de ordenación, formando catálogo de ellos, se puede asumir que la catalogación es todo el procedimiento que se realiza para almacenar ordenadamente y clasificar un conjunto de contenidos" (9).

Al hablar de la catalogación como proceso, con independencia del contexto en que se sitúe, se está en presencia de una actividad propia de la organización de la información (11).

"Registro ordenado de libros, documentos o material semejante de acuerdo con unas normas" (12).

El desarrollo armónico de la catalogación, exige establecer, previamente, lo que se define como política de catalogación. Es decir, ante qué documento, que tipo de usuarios habitualmente van a acceder a él y por lo tanto, a qué nivel de profundidad amplitud se va a catalogar. Del estudio previo de esta política, dependerá que los usuarios puedan encontrar más o menos información en los distintos catálogos.

"El catálogo es el instrumento clásico generado por las bibliotecas para organizar el almacenamiento y difusión de la información que conforma la colección de documentos. El catálogo fue siempre el principal

instrumento para acercar la colección al usuario. La automatización del catálogo permitió integrar en una sola herramienta los diferentes catálogos y multiplicar los puntos de acceso a la información referencial, mejorando las posibilidades de búsqueda y cercándolo a las necesidades del usuario" (13).

Principios de la catalogación:

Con el término principios en la literatura más especializada se alude tanto a los objetivos de un sistema bibliográfico como a los fundamentos estructurales que dan coherencia, unidad y eficacia a un código de catalogación y al lenguaje documental sobre los que se apoya un sistema bibliográfico dado. Conforme a la teoría general de sistemas, dos principios son comunes a cualquiera de ellos (14):

- Principio de razón suficiente, por el que todos los elementos que lo constituyen deben tener fundado un tipo de relación con el resto; así se garantiza que los resultados estén justificados por un tipo de relación dado.
- Principio de brevedad, por el que es posible escoger entre varias opciones para alcanzar una meta, siempre se opta por el procedimiento más simple y corto.
- Principio de uso adecuado, por el que las descripciones se hacen pensando en las necesidades de los usuarios.
- Principio de representación, por el que toda descripción se basará en una entidad que se describe a sí misma.
- Principio de suficiencia y necesidad, por el que las descripciones obedecerán a objetivos concretos y no incluirán elementos no derivados de estos propósitos.
- Principio de normalización, por el que las descripciones serán normalizadas tanto en extensión como en nivel.
- Principio de integración, por el que las descripciones para las diversas categorías de materiales se fundamentarán en un conjunto común de reglas, que deberá ser lo más completo posible.

Principios relacionados con los primeros, y con una categoría menor serían:

- Principio de uso común, por el que las descripciones emplearán un vocabulario normalizado ampliamente aceptado por los usuarios a los que se dirige tal sistema.
- Principio de precisión, por el que las descripciones representarán fielmente a entidades distintas.
- Principio de significación, por el que las descripciones incluirán solo aquellos elementos que posean contenido informativo a efectos bibliográficos.

Pilares fundamentales de la catalogación (15):

- Estandarización de la información.
- Homogeneidad.
- Respeto por las peculiaridades.
- Flexibilidad en lo accesorio.
- Organización de la información por categorías.
- Aplicabilidad en diversos soportes.
- Se organiza por niveles.

Tipos de catalogación (15):

- Particular: Se realiza para una organización o una persona con fines de uso solo para esa organización o esa persona.
- Cooperativa: Se realiza en varias organizaciones similares o unidas por algo en común, en diversas unidades autónomas, pero aportando un producto en común (catálogo colectivo, común o cooperativo).
- Centralizada: Es realizada en una organización con varios organismos distintos, alejados entre sí, en una unidad central adoptada por las mismas, pero con distribución a sus unidades dependientes.

La catalogación es un proceso que es aplicado para darle solución a la gestión de activos de tipo media. Este proceso es de suma importancia cuando se cuenta con un gran cúmulo de información, en este caso archivos tales como modelos, texturas, audios y videos. La catalogación ayuda de forma directa a colocar donde se determine los ficheros con una descripción previa, para luego acceder a los mismos de forma organizada. Cuando existen varios recursos informáticos bien catalogados es más fácil trabajar con ellos, tanto a la hora de administrarlos como de usarlos. La catalogación es el módulo principal en la concepción de la plataforma desarrollada, la determinante funcionalidad de dicho módulo lo convierte en un componente imprescindible para el adecuado trabajo de cualquier aplicación donde se necesite almacenar y reutilizar recursos informáticos, pues aminora la labor humana sobre la máquina proporcionando disminuir el tiempo de búsqueda de la información requerida.

1.2. Estudio de tendencias

La presente investigación tiene entre sus premisas el estudio de tendencias para la implementación del módulo de Reporte, el cual permite la generación de reportes gráficos a partir de las evaluaciones de indicadores. *“Tendencia es cuando se observa que los datos estudiados presentan preferencia a estar de*

una forma u otra, es decir, cuando los datos tienden a elevarse en el gráfico esa es una tendencia al aumento en largo plazo" (16).

Las tendencias son información que se representan en forma de gráficas, diagramas y tablas, donde se muestran los datos y su comportamiento en el tiempo. Sirven para revisar el comportamiento de una variable o grupo de variables y pronosticar su tendencia y así facilitar una correcta toma de decisiones por parte de las organizaciones. La tendencia es un gráfico histórico que se va actualizando permanentemente o en tiempo real según cambie la variable graficada.

Mediante la visualización gráfica de los datos se puede detectar la variabilidad y consistencia de los mismos en el tiempo, permitiendo anticiparse a futuros acontecimientos y tomar las decisiones correctas. Conocer el estado y tendencia real de los diferentes parámetros en régimen estable o transitorio que determinan el perfecto desempeño de los sistemas, permite prevenir y evitar fallos inesperados en el sistema informático, bloqueo de los sistemas de procesamiento de datos y pérdida de información.

En la actualidad las empresas requieren que su información sea procesada, para ello necesitan herramientas que le permitan generar reportes dinámicos que muestren la información actualizada y requerida. El presente trabajo introduce una solución informática que permite adaptar un sistema de análisis y reportes a través de la observación de diferentes variables, y sobre las cuales se pueden tomar decisiones que apoyen el proceso de producción. Como resultado final se contará con gráficos estadísticos para analizar las tendencias.

1.3. Elementos Motivacionales

La motivación ha estado siempre presente en todas las épocas de la historia del hombre. La misma es un fenómeno complejo, contradictorio y diverso, que incluye a todos los factores internos y externos que organizados de alguna manera, realizan la motivación del sujeto y determinan la dirección de su comportamiento.

La estimulación es un sentimiento provocado por diferentes razones, producto del deseo de hacer u obtener algo, el cual muchas veces nos puede llevar a actuar en forma positiva y otras veces en forma negativa.

En el presente trabajo se considera que para lograr impulsar la motivación de los trabajadores del centro VERTEX en el proceso de producción de activos de tipo media, se hace necesario la creación de un módulo que permita aumentar los niveles motivacionales en dicho centro. Los mecanismos a utilizar en dicho módulo son:

Utilización de reportes a través de ranking (rango) que incluyen:

- Representación de los usuarios que más activos aportan a la red sin consumir.
- Representación de los usuarios que más consumen activos de la red sin aportar.
- Representación de los usuarios que más aportan activos a la red y a su vez, tienen alto índice de reutilización.
- Representación de los usuarios que menos aportan activos a la red y a su vez, tienen más bajo índice de reutilización.
- Representar los activos de diseño más reutilizados.
- Representar los activos de diseño con más votos obtenidos.
- Representar los activos de diseño más comentados.

Aquellas entidades proveedoras de activos que resulten tener un alto aporte a la red, serán estimuladas según las políticas de un sistema de recompensas establecido previamente por la entidad responsable del repositorio de activos de tipo media, en virtud de estimular la publicación de activos en el repositorio y garantizar su intercambio al entorno de reutilización establecido en la organización. Para estimular un mejor intercambio en un entorno de reutilización de activos de diseño, sería apropiado establecer un sistema de recompensas que como consecuencia de la propia estimulación, aumente la publicación de activos y a su vez fortalece la reutilización. Este mecanismo actúa como herramienta para la asignación de premios virtuales para aquellas entidades destacadas en el proceso de publicación de artefactos de diseño. Se define como premio virtual al reconocimiento virtual que se le hace a una persona por su labor y aporte a la publicación de activos en el servidor de medias.

1.4. Análisis de soluciones existentes

Para determinar las características y funcionalidades que debe cumplir la aplicación a realizar es necesario hacer un análisis sobre los sistemas que existen en la actualidad y las tendencias actuales que se emplean en situaciones similares tanto en el ámbito internacional o nacional. En el capítulo se abordan los sistemas: Blendswap, Thingiverse, Cubehero, 3dm3 y la aplicación web InterNos existente en la Universidad de las Ciencias Informáticas (UCI), los cuales tienen puntos comunes con el sistema a desarrollar. A continuación se hará una breve descripción de cada sistema y sus limitaciones en cuanto a su utilización, con el objetivo de estudiar sus ventajas y desventajas de forma tal, que contribuya a crear una nueva aplicación ajustable a las necesidades.

1.4.1. Blendswap:

Es una aplicación web colaborativa para subir y descargar modelos 3D realizados con Blender. Todos los modelos están disponibles en extensión *.blend* y han sido liberados bajo licencia por sus autores. La colección puede considerarse como una de las más grandes, ya que cuenta con más de 13 242 modelos organizados en categorías. Además este sitio es gratuito y permite que se envíen modelos para incluirse en la colección.

El sistema permite acceder de forma rápida a la información de un modelo determinado mostrando un resumen con los datos relevantes del mismo, además permite comentar los ejemplares así como añadirlos a la biblioteca personal de cada usuario. Las búsquedas que se realizan son básicas pero también es relevante el nivel de detalle para filtrar los resultados que se alcanza en las búsquedas dado por la cantidad de parámetros a tener en cuenta en la misma.

Entre las principales limitaciones que se evidencian en el sistema se deben mencionar que se limita la subida de modelos en otros formatos que no sean *.blend* y solo permite el trabajo con modelos, lo que excluye otros activos de media tales como videos, texturas y audios.

1.4.2. Thingiverse:

Otra aplicación web que se puede encontrar internacionalmente es Thingiverse. Es una comunidad en línea en la que los usuarios registrados idean y comparten sus diseños 3D más originales, acompañados de notas y explicaciones donde describen detalladamente los materiales y técnicas utilizadas y, sobre todo la finalidad del aparato construido.

Esta es una plataforma para el intercambio de modelos 3D donde es posible descargar objetos gratis una vez que el usuario es miembro de esta comunidad. Adquirir modelos gratis del catálogo de este espacio es fácil ya que posee en la fecha actual más de 2000 ejemplares. Esta aplicación en su página principal tiene etiquetados para acceder de forma rápida a: los últimos diseños publicados, los más destacados, los de mayor popularidad y además permite realizar búsquedas. También posee todos los modelos organizados por categorías de forma tal que se puede encontrar el ejemplar que se busca de manera más eficiente.

Independientemente de las ventajas que pueda propiciar esta plataforma, se ve limitada para realizar diversas funciones como son:

El tipo de contenido que maneja está limitado a modelos solamente lo que excluye otros tipos de activos de tipo media. No se visualiza información necesaria con los detalles de los modelos almacenados en el sistema por lo que el usuario recibe toda la información del objeto hasta que lo descarga e incluso la

ausencia de datos correspondientes a los diseños impide la ejecución de filtros de búsquedas, ya que no se cuentan con aspectos para el filtrado de los resultados.

1.4.3. Cubehero:

En el ámbito internacional existe la aplicación web Cubehero**,** es otra plataforma más pequeña creada para ayudar a los usuarios que diseñan modelos 3D, los cuales interactúan y colaboran en proyectos que se realizan. Esta aplicación realiza procesos como: la publicación y descarga de modelos, compartir un objeto de diseño para que los demás miembros de la comunidad puedan disfrutar del mismo, permite realizar búsquedas basadas en el título, además de listar todos los objetos de una determinada categoría. Todo esto permite el acceso fácil y rápido desde cualquier lugar y el intercambio de modelos entre usuarios de todo el mundo.

A pesar de lo ventajoso de este producto y de poseer más de 1000 modelos hasta la fecha actual, tiene sus deficiencias en cuanto a: las búsquedas que se pueden realizar se ven limitadas a obtener un resultado según el criterio definido por la aplicación, ya que no posee filtros para su ejecución, no permite realizar búsquedas avanzadas lo que dificulta la recuperación de un determinado ejemplar. Se ve limitado al manejo de modelos solamente lo que excluye otros tipos de objetos de diseño 3D.

1.4.4. 3dm3:

Es una de las plataformas web más conocidas internacionalmente, donde los usuarios pueden cargar y descargar modelos 3d, texturas y videos de forma gratuita. Todos los activos de diseño se encuentran organizados por categorías de forma tal que se puede acceder fácilmente al objeto que se busca. La plataforma permite realizar comentarios de cada uno de los artefactos que posee, tiene un foro donde los miembros de la comunidad pueden intercambiar opiniones e ideas sobre un determinado tema. El sistema permite acceder de forma rápida a la información de un objeto determinado mostrando un resumen con los datos relevantes del mismo, las búsquedas que se realizan son básicas y utilizan como criterio el nombre del elemento.

Independientemente de la facilidad de información que provee este portal, posee deficiencias en cuanto a: no permite realizar búsquedas avanzadas ni posee filtros para su ejecución, por lo que los usuarios no pueden encontrar con facilidad lo que desean obtener. En las vistas detalladas de cada activo no se muestra la información necesaria para que el usuario conozca más detalles del elemento seleccionado.

1.4.5. InterNos:

En la intranet cubana, específicamente en la Universidad de las Ciencias Informáticas (UCI), existe una plataforma para la distribución de contenidos audiovisuales llamada: InterNos, donde se encuentran varias publicaciones de diferentes materiales audiovisuales. Este sitio web representa un espacio ideal para estudiantes, profesores y trabajadores, en el mismo pueden disfrutar de distintas publicaciones para su entretenimiento. En el portal se publican teleclases docentes, canales de la televisión en vivo las 24 horas del día, series, películas y documentales. Esta aplicación posee todos los materiales organizados por categorías, lo que permite que se pueda localizar con facilidad al ejemplar buscado. En su página principal posee etiquetas para acceder de forma rápida a los videos más votados y a los que tienen mayor cantidad de visitas, y además permite realizar búsquedas.

Esta aplicación tiene algunas desventajas: no permite la publicación de contenidos audiovisuales por parte de los usuarios ajenos a la administración del sitio, se ve limitado solo a la manipulación de videos, lo que excluye otros tipos de activos de tipo media tales como, audios, texturas y modelos 3D. Otra dificultad es que la recuperación de los contenidos audiovisuales resulta algo complejo porque los resultados obtenidos no pueden ser filtrados ni se pueden ejecutar búsquedas avanzadas.

La gestión de activos de tipo media es un proceso muy importante dentro de toda entidad que trabaje con este tipo de archivos. En la actualidad la mayoría de las aplicaciones que se encuentran en el mercado de software y que tienen algo que ver con este proceso, son privativas y además pertenecen a empresas privadas; y las que se han implementado en software libre no engloban todas las funcionalidades que se requieren actualmente. La mayoría de estas aplicaciones se centran solo en la gestión de modelos 3D, o en más artefactos de diseño, pero no en la gestión de todo tipo de activos de diseños como modelos, texturas, videos y audios. Luego de realizarse un detallado análisis de las herramientas existentes en el mercado del software se puede llegar a la conclusión de que no existe un sistema que automatice el proceso de gestión de activos de tipo media.

1.5. Metodologías de desarrollo

Se define como metodología al conjunto de estrategias, métodos o actividades intencionadas, organizadas, secuenciadas e integradas, que permitan el logro de aprendizajes significativos y de calidad (17) .

Entre los principales objetivos de las metodologías de desarrollo de software se encuentran cumplir con los requisitos iniciales del problema, minimizar la pérdida de tiempo, minimizar riesgos e incrementar las

posibilidades de éxito en el desarrollo de los productos. La metodología define “Quién” debe hacer “Qué”, “Cuándo” y “Cómo” debe hacerlo. Este es un proceso que debe ser configurable a las características de cada proyecto, ya que no son universales.

En los últimos tiempos se han desarrollado dos corrientes en lo referente a los procesos de desarrollo: los llamados métodos pesados y los métodos ágiles. La diferencia fundamental entre ambos es que mientras los primeros intentan conseguir el objetivo común por medio de orden y documentación, los segundos tratan de mejorar la calidad del software por medio de una comunicación directa e inmediata entre las personas que intervienen en el proceso.

A continuación se detallarán algunas características de las Metodologías Programación Extrema (conocido por sus siglas en inglés XP) y Proceso Unificado de Racional (famoso por sus siglas en inglés RUP), a razón de que hoy día son de las más utilizadas, en vista de una mayor comprensión de la selección realizada.

1.5.1. Proceso Unificado de Racional (RUP)

El Proceso Unificado de Desarrollo de Software (*Rational Unified Process* en inglés, habitualmente resumido como RUP) es el más apropiado para el desarrollo de grandes proyectos, ya que requiere un equipo de trabajo capaz de administrar un proceso complejo en varias etapas.

RUP no es un sistema que tiene firmes pasos establecidos, sino un conjunto de metodologías adaptables a las necesidades de cada organización. Tiene una forma disciplinada para asignar tareas y responsabilidades dentro de una organización de desarrollo pues persigue como objetivo asegurar la producción de software de alta calidad para satisfacer los requerimientos de los usuarios finales (respetando cronograma y presupuesto).

Es un proceso iterativo e incremental que se encarga de dividir el trabajo en partes más pequeñas o en mini proyectos, permitiendo que se logre un equilibrio entre la arquitectura y casos de uso durante cada mini proyecto. En cada iteración del proyecto o mini proyecto se obtiene un incremento, provocando un aumento del producto (18).

El proceso de desarrollo (RUP) se divide en ciclos, el mismo está definido por 4 fases fundamentales (19):

- Conceptualización (Concepción o Inicio): Se describe el negocio y se delimita el proyecto describiendo sus alcances con la identificación de los casos de uso del sistema.
- Elaboración: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.

- Construcción: Se adquiere un producto listo para su utilización que está documentado y tiene un manual de usuario. Se obtienen varios entregables del producto y se ponen a consideración de un conjunto de usuarios.
- Transición: El entregable ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

RUP se caracteriza por ser dirigido por casos de uso, permitiendo a través de estos definir lo que el usuario desea a partir de la captura de requisitos. Es centrado en la arquitectura, lo cual brinda una visión completa del sistema, a través de la descripción de los procesos del negocio que son más importantes, para comprenderlo, desarrollarlo y producirlo de una forma eficaz. Iterativo e Incremental donde cada fase se desarrolla en iteraciones, de forma tal que se pueda dividir en pequeños proyectos mejorando su comprensión y desarrollo.

Otras de sus características más importantes son (20):

- Propone un flujo de trabajo específicamente para el Análisis y Diseño del sistema, donde se especifican los requisitos identificados y se describen en términos de diseño el modo de implementar el sistema.
- Provee amplia documentación al equipo de trabajo para documentar bien el software (planillas).
- Describe la manera de obtener, organizar y documentar todas las funcionalidades y restricciones del sistema.
- En cada una de sus fases obtiene una serie de artefactos que sirven para comprender mejor tanto el Análisis como el Diseño del sistema.

1.5.2. Programación Extrema (XP)

La Programación Extrema XP por sus siglas en inglés (E*xtreme Programming*) es una metodología formulada por Kent Beck en 1999. La Programación Extrema (XP) es una metodología ágil que intenta reducir la complejidad del software por medio de un trabajo orientado al objeto, basado en las relaciones interpersonales. Para lograr el éxito en un proyecto es necesario la capacitación de los desarrolladores, agradable ambiente de trabajo y participación del cliente en el equipo de desarrollo.

XP es adecuada para proyectos con requisitos imprecisos y muy cambiantes, donde existe un alto riesgo técnico. Además es aplicable a proyectos donde se puedan emplear prácticas adaptativas y flexibles en vez de predictivas, donde sea factible realizar un producto inicial entregable con un conjunto básico de prestaciones, para luego ir añadiendo funcionalidades que satisfagan todas las necesidades del cliente.

Entre las principales características de esta metodología se encuentran:

- Permite introducir nuevos requisitos o cambiar los anteriores ágilmente.
- Adecuado para proyectos pequeños y medianos.
- Adecuado para proyectos con alto riesgo.
- Su ciclo de vida es iterativo e incremental.
- Cada iteración dura entre una y tres semanas.
- No produce demasiada documentación acerca del diseño o la planificación.

Un proyecto XP tiene éxito cuando el cliente selecciona el valor de negocio a implementar basado en la habilidad del equipo para medir la funcionalidad que puede entregar a través del tiempo. El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos (21):

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos.

De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración.

El ciclo de vida ideal de XP consiste de seis fases: Exploración, Planificación de la Entrega, Iteraciones, Producción, Mantenimiento y Muerte del Proyecto (22).

1.5.3. Selección de la metodología de desarrollo

Luego de analizar las características de las metodologías mencionadas anteriormente y basándose en las particularidades del desarrollo del software a implementar, se selecciona XP para que guíe el proceso de desarrollo, debido a que:

- Es la que mejor se ajusta a las necesidades del proyecto en cuanto a recursos técnicos, humanos y tiempo de desarrollo.

- Se consiguen productos usables con mayor rapidez.
- El cliente forma parte del equipo de trabajo y esto trae consigo aumentar la probabilidad de éxito.
- Se atienden las necesidades del usuario con mayor exactitud, lográndose gracias a que se les ofrecen continuas versiones.
- Es adecuada para proyectos con requisitos imprecisos y cambiantes.
- Se consigue tener un equipo de desarrollo más motivado. Las razones son, por un lado la XP no permite excesos de trabajo (se debe trabajar 40 horas a la semana), y por otro la comunicación entre los miembros del equipo que consigue una mayor integración entre ellos.
- Permite que se pueda regresar a operaciones anteriores sin afectar el ciclo de vida del software en cualquier momento del proceso de desarrollo de la aplicación.
- Que el software funcione es más importante que la documentación exhaustiva. Desarrollar un software funcional es lo principal, los documentos deben ser cortos y centrarse en lo fundamental.

1.6. Lenguaje Unificado de Modelado (UML)

Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables (7).

El Lenguaje Unificado de Modelado (UML: *Unified Modeling Language*) constituye el estándar internacionalmente más aceptado. Además permite modelar sistemas de software orientado a objetos, incluye aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. Fue pensado para ser usado por sistemas desarrollados en diferentes lenguajes y plataformas.

Es el resultado del trabajo realizado por Grady Booch, James Rumbaugh e Ivar Jacobson en busca de un lenguaje no solo para comunicar las ideas a otros desarrolladores sino también para servir de apoyo en los procesos de análisis (23).

A continuación se mencionan las principales características de UML:

- Se pueden especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.

- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas y versiones).
- Cubre las cuestiones relacionadas con el tamaño propio de los sistemas complejos y críticos.
- Es un lenguaje muy expresivo que cubre todas las vistas necesarias para desarrollar y luego desplegar los sistemas.
- Ampliamente utilizado por la industria desde su adopción.
- Modela estructuras complejas.
- Las estructuras más importantes que soporta tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables si es necesario.

1.7. Estándares de programación web

1.7.1. Hojas de Estilo en Cascada (CSS)

Las Hojas de Estilos en Cascada (*Cascading Style Sheets* [CSS]) es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos.

CSS se utiliza para dar estilo a documentos HTML y XML y separa el contenido de la presentación. Los Estilos definen la forma de mostrar los elementos HTML y XML. CSS permite a los desarrolladores web controlar el estilo y el formato de múltiples páginas web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento (24).

CSS es un conjunto de reglas que se suelen aplicar como elementos dentro de la cabecera de la página web o se suelen ubicar en archivos externos. Su objetivo es recopilar todas las reglas que serán aplicadas a las etiquetas de una página web en un mismo espacio de forma tal que sea mucho más simple su modificación y mantenimiento. CSS es imprescindible para crear páginas web complejas (25).

La principal ventaja de CSS en HTML es que la presentación del marcado del estilo puede ser totalmente independiente del contenido. Mediante la separación de la capa de presentación, CSS permite que todos los estilos se mantengan en un número limitado de hojas permitiendo guardar el estilo de 10000 archivos HTML aproximadamente y de esta manera el ahorro global en el ancho de banda es medible. Con el uso

del CCS se logra un control centralizado de la presentación de un sitio web completo, agilizando en el mismo la actualización de sus páginas (26).

1.7.2. Lenguajes de Marcado de Hipertexto (HTML)

El Lenguaje de Marcación de Hipertexto o HTML (en inglés *Hypertext Markup Language*), es el lenguaje predominante para la construcción de páginas web. Su propósito fundamental es definir la estructura y apariencia básica de documentos de tal manera que puedan ser manejados de forma rápida y fácil por un usuario en la red para verlos en diferentes dispositivos. Unas de sus principales ventajas es su facilidad de uso porque pude ser editado desde cualquier editor de texto. Tiene dos características esenciales: hipertexto y universalidad. La primera significa que puede crear un vínculo en una página web hacia otra mientras que la segunda significa que, puesto que los documentos HTML se guardan como archivos de solo texto, cualquier ordenador puede leer una página web.

HTML es una implementación del estándar SGML (*Standard Generalized Markup Language*), estándar internacional para la definición de texto electrónico independiente de dispositivos, sistemas y aplicaciones. Es un metalenguaje para definir lenguajes de diseño descriptivos; proporciona un medio de codificar documentos hipertexto cuyo destino sea el intercambio directo entre sistemas o aplicaciones (27).

HTML 5 es una colección de estándares para el diseño y desarrollo de páginas web. Esta colección representa la manera en que se presenta la información en el explorador de internet y la manera de interactuar con ella. Permite una mayor interacción de las páginas web con contenidos de reproducción digital (videos, audio, juegos, entre otros) y facilidad a la hora de maquetar una página web (27).

1.8. Lenguaje de programación

1.8.1. JavaScript

Javascript es un lenguaje de escritura (en inglés *script*) desarrollado por Netscape para incrementar las funcionalidades del lenguaje HTML. Se utiliza embebido en el código HTML y XHTML, entre las etiquetas <script> y </script>. JavaScript es un lenguaje interpretado en el cliente por el navegador al momento de cargarse la página, es multiplataforma, orientado a eventos con manejo de objetos, cuyo código se incluye directamente en el mismo documento HTML.

Hasta entonces ya se usaba HTML y Java, pero la aparición del JavaScript produjo una importante revolución, ya que dio al usuario la posibilidad de crear aplicaciones en línea (on-line) (27).

Los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. Se utiliza integrado en un navegador web permitiendo el desarrollo de interfaces de usuario mejoradas y páginas web dinámicas. Es soportado por la mayoría de los navegadores como Internet Explorer, Netscape, Opera y Mozilla Firefox y gracias a esto es el más utilizado por parte de los clientes.

Dentro de sus características más importantes se pueden encontrar (28):

- Es un lenguaje interpretado, es decir, no requiere compilación. El navegador del usuario se encarga de interpretar las sentencias Javascript contenidas en una página HTML y ejecutarlas adecuadamente.
- Es un lenguaje orientado a eventos. Cuando un usuario presiona el cursor sobre un enlace o mueve el puntero sobre una imagen se produce un evento. Mediante este lenguaje se puede desarrollar Scripts que ejecuten acciones en respuesta a estos eventos.
- Es un lenguaje orientado a objetos. El modelo de objetos de Javascript está reducido y simplificado, pero incluye los elementos necesarios para que los Scripts puedan acceder a la información de una página y puedan actuar sobre la interfaz del navegador.

Ventajas:

- Los programas JavaScript tienden a ser pequeños y compactos, no requieren mucha memoria ni tiempo adicional de transmisión.
- Funciona en cualquier navegador que soporte JavaScript independientemente de la plataforma o el sistema operativo.
- El código JavaScript es gestionado en el navegador.

1.9. Herramienta CASE

En las últimas décadas se ha trabajado en el área de desarrollo de sistemas para encontrar técnicas que permitan incrementar la productividad y el control de calidad en cualquier proceso de elaboración de software.

Hoy en día la Ingeniería de Software Asistida por Ordenador (*Computer Aided Software Engineering*), CASE por sus siglas en inglés, reemplaza al papel y al lápiz por el ordenador para transformar la actividad de desarrollar software en un proceso automatizado.

Las herramientas CASE son un conjunto de programas y ayuda que dan asistencia a los analistas, ingenieros de software y desarrolladores durante todos los pasos del ciclo de vida de desarrollo de un software (29).

Estas herramientas son de gran utilidad pues intervienen en todos los aspectos del ciclo de vida del desarrollo del software, permitiendo a los desarrolladores menos complejidad a la hora de realizar diferentes actividades del proceso de software como son la realización del diseño del proyecto, la documentación o detección de errores entre otros.

Entre los principales objetivos que se buscan al utilizar una herramienta CASE están (30):

- Mejorar la productividad en el desarrollo y mantenimiento del software.
- Aumentar la calidad del software.
- Reducir el tiempo y coste de desarrollo y mantenimiento de los sistemas informáticos.
- Mejorar la planificación de un proyecto.
- Aumentar la biblioteca de conocimiento informático de una empresa ayudando a la búsqueda de soluciones para los requisitos.
- Automatizar el desarrollo del software, la documentación, la generación de código, las pruebas de errores y la gestión del proyecto.
- Ayuda a la reutilización del software, portabilidad y estandarización de la documentación.
- Gestión global en todas las fases de desarrollo de software con una misma herramienta.
- Facilitar el uso de las distintas metodologías propias de la ingeniería del software.

Las herramientas CASE se pueden clasificar teniendo en cuenta los siguientes parámetros:

- Las fases del ciclo de vida del desarrollo de sistemas que cubren.
- La arquitectura de las aplicaciones que producen.
- Su funcionalidad.

1.9.1. Visual Paradigm

Es una herramienta CASE que brinda sustento al desarrollo de software orientado a objetos, para ello hace uso del lenguaje de modelado UML. Es un software potente con gran facilidad de uso. Ofrece soporte para el modelado y visualización de componentes necesarios durante todo el ciclo de vida del software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Esta herramienta es de gran ayuda para la construcción de una aplicación con alta calidad y a un menor costo. La misma está diseñada para una

amplia gama de usuarios, incluidos los ingenieros de software, analistas de sistemas y analistas de negocio, o para cualquiera que esté interesado en la construcción de forma fiable a gran escala de sistemas de software, utilizando un enfoque orientado a objetos (31).

Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Posee capacidad para realizar ingeniería inversa con diferentes lenguajes de programación orientados a objetos. Facilita la interoperabilidad con otras herramientas CASE como el Rational Rose y puede ser integrada con herramientas Java. Brinda facilidades para la exportación e importación de componentes. En los últimos tiempos se ha convertido en la herramienta de modelado por excelencia para el desarrollo de sistemas en plataformas libres, ya que a pesar de su licencia comercial su versión Community posee licencia gratuita. Es una herramienta multiplataforma, diseñada para el trabajo tanto en Windows como en Linux (9).

Características (32):

- Producto de calidad.
- Soporta aplicaciones web.
- Las imágenes y reportes generados, no son de muy buena calidad.
- Generación de código para Java y exportación como HTML.
- Fácil de instalar y actualizar.
- Compatibilidad entre ediciones.

1.9.2. Rational Rose

Es una herramienta CASE diseñada para brindar soporte al proceso de desarrollo de software. Rational Rose cubre todo el ciclo de vida de una empresa: concepción y formalización del modelo, construcción de los componentes, transición a los usuarios y certificación de las distintas fases y entregables. Una de sus grandes ventajas es que utiliza la notación estándar en la arquitectura de software (UML), la cual permite a los arquitectos de software y desarrolladores visualizar el sistema completo utilizando un lenguaje común, además los diseñadores pueden modelar sus componentes e interfaces en forma individual y luego unirlos con otros componentes de la empresa (33).

Posee facilidades para la generación de códigos a partir de diagramas previamente elaborados. Facilita la ingeniería inversa de códigos fuentes en distintos lenguajes de programación basados en el paradigma de programación orientado a objetos. Es una herramienta de desarrollo basada en modelos que se integra con

las bases de datos y los entornos de desarrollo integrado de las principales plataformas del sector. Todos los productos de Rational Rose dan soporte a UML, pero no son compatibles con las mismas tecnologías de implementación. Rational Rose Enterprise es un entorno de modelado que permite generar código a partir de modelos Ada, ANSI C++, C++, Java/J2EE, Visual C++ y Visual Basic. Ofrece un lenguaje de modelado común que agiliza la creación del software (34).

Esta herramienta permite especificar, analizar, diseñar el sistema antes de codificarlo. Tiene varias características que lo distinguen como son:

- Admite como notaciones: UML, OMT y Booch.
- Mantiene consistencia de los modelos del sistema del software.
- Permite el desarrollo multiusuario.
- Permite la ingeniería inversa (crear modelo a partir del código).
- Genera documentación del sistema.
- Genera código a partir de los modelos.

1.9.3. Selección de la herramienta CASE para modelado del software

Para el desarrollo del sistema propuesto se selecciona la herramienta CASE Visual Paradigm por los siguientes motivos: posee gran facilidad de uso y proporciona calidad a la hora de desarrollar una aplicación. Acelera el desarrollo de aplicaciones, ya que sirve de puente visual entre arquitectos, analistas y diseñadores de sistemas de información. Además es sencilla a la hora de dibujar diagramas y generar código. Dicha herramienta es multiplataforma y colaborativa, diseñada tanto para el sistema operativo Windows como para las diferentes distribuciones de Linux, facilitando la importación y exportación de componentes, ya sea como imagen o como archivos XML. Apoya a un conjunto de idiomas tanto en la generación de código como en la ingeniería inversa de los mismos.

1.10. Sistemas de Gestión de Bases de Datos (SGBD)

Los Sistemas de Gestión de Bases de Datos o SGBD (en inglés *Database Management System*, abreviado DBMS) se pueden definir como un paquete de software, dedicado a servir de interfaz entre la base de datos, la aplicación que hace uso de esta y los usuarios. Las principales funciones que deben cumplir estos SGBD son: creación y mantenimiento de la integridad de los datos de la base de datos, la manipulación de datos

de acuerdo con las necesidades del usuario, control de accesos y privacidad de los datos y el cumplimiento de las normas de tratamiento de datos.

De forma general se destacan entre ellos las siguientes ventajas (7):

- Gran velocidad de procesamiento en muy poco tiempo.
- Integridad referencial el terminar los registros.
- Independencia del tratamiento de información.
- Facilidad de manejo de grandes volúmenes de información.
- No hay duplicidad de información, comprobación de información en el momento de introducir la misma.
- Seguridad de la información (acceso a usuarios autorizados), protección de información, de modificaciones, inclusiones, consulta.

Existen muchas empresas y sitios web que necesitan mantener de forma eficiente un gran volumen de datos. La solución en la que muchos confían es optando por una variante como PostgreSQL o MySQL ya que son los gestores más conocidos en la actualidad, los cuales presentan las características que se detallan a continuación.

1.10.1. MySQL

MySQL es un sistema de gestión de base de datos relacional capaz de distribuir una gran cantidad de datos. Gracias a su diseño multihilo le permite eficientemente soportar una gran carga, es multiusuario y posee una arquitectura cliente servidor. Es soportado por muchos lenguajes de programación que se usan en la actualidad y utiliza el lenguaje de consulta estructurado SQL *(Structured Query Language*). SQL es un lenguaje normalizado para consultar y actualizar los datos. El SGBD MySQL es gratuito y de código abierto por lo que cualquier usuario puede acceder al código fuente, es decir, al código de programación de MySQL. Posee un sistema de privilegios y contraseñas que es muy flexible y seguro permitiendo la verificación basada en el host (35).

En los últimos años este sistema con más de 6 millones de instalaciones se convirtió en propietario y hoy en día está patrocinado por una empresa privada que posee el derecho de autor de la mayor parte del código. Este gestor de bases de datos, es probablemente el más usado en el mundo del software libre, y esta gran aceptación es debido a sus ventajas (7):

- Velocidad al realizar las operaciones, lo que le hace uno de los gestores con mejor rendimiento.

- Bajo costo en requerimientos para la elaboración de bases de datos, ya que debido a su bajo consumo puede ser ejecutado en una máquina con escasos recursos sin ningún problema.
- Facilidad de configuración e instalación.
- Soporta gran variedad de Sistemas Operativos.
- Baja probabilidad de corromper datos, incluso si los errores no se producen en el propio gestor, sino en el sistema en el que está.
- Conectividad y seguridad.

1.10.2. PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo la licencia BSD (*Berkeley Software Distribution*) y con su código fuente disponible libremente. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Se ejecuta en varios sistemas operativos como por ejemplo: Linux, Unix, Mac OS y Windows (36). PostgreSQL se desempeña muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Es adaptable a las necesidades del cliente y posee una variada documentación en diferentes idiomas debido fundamentalmente a sus numerosas comunidades de desarrollo.

Posee características avanzadas como consultas complejas, llaves foráneas, vistas, integridad transaccional, control de concurrencia y disparadores. Sus características técnicas la hacen una de las bases de datos más potentes del mercado.

Ventajas de PostgreSQL (37):

- Instalación ilimitada, multiplataforma.
- Mejor soporte que los proveedores comerciales.
- Ahorros considerables en costos de operación.
- Estabilidad y confiabilidad legendarias.
- Extensible.
- Diseñado para ambientes de alto volumen.
- Herramientas gráficas de diseño y administración de bases de datos.

1.10.3. Selección del sistema gestor de base de datos

Después de analizar los principales gestores de base de datos que existen en la actualidad y que dan cumplimiento a las necesidades técnicas para el desarrollo de la solución que se pretende, se ha llegado a la conclusión, que MySQL es el más conveniente para el desarrollo de la base de datos que exige la aplicación. MySQL proporciona un servidor de base de datos SQL muy rápido, multiusuario y robusto. Cuenta con tentadoras ventajas que han sido la razón principal de la selección, como son: su bajo costo, alto rendimiento, facilidad de configuración y aprendizaje.

1.11. Marco de trabajo

Marco de trabajo o *framework* (palabra inglesa que significa marco de trabajo) es un esquema (un esqueleto, un patrón) para el desarrollo y/o la implementación de una aplicación. Son unas librerías de código que contienen procesos o rutinas ya listas para usar. Los programadores lo utilizan para no tener que desarrollar ellos mismos las tareas más básicas. En el propio *framework* ya hay implementaciones que están probadas, funcionan y son reutilizables. Además ayuda a desarrollar aplicaciones de manera rápida, fácil, modular y sencilla, ahorrando tiempo y esfuerzo.

En el desarrollo de software, un *framework* o infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con módulos de software concretos, con base a la cual otro proyecto de software puede ser más fácilmente organizado y desarrollado. Estos persiguen acelerar el proceso de desarrollo, reutilizar código ya existente y brindar buenas prácticas de desarrollo como el uso de patrones. Constituyen una herramienta de extrema importancia para la implementación de aplicaciones en la actualidad (38).

Ventajas (39):

- El programador no necesita plantearse una estructura global de la aplicación, sino que el *framework* le proporciona un esqueleto que hay que "rellenar".
- Facilita la colaboración. Se conoce lo difícil que resulta el entendimiento del código de otra persona por lo que todo lo que sea definir y estandarizar va a ahorrar tiempo y trabajo a los desarrollos colaborativos.
- Es más fácil encontrar herramientas (utilidades, librerías) adaptadas al *framework* concreto para facilitar el desarrollo.

- Un *framework* simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes y además facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

A continuación se analizarán algunos de los marcos de trabajo más usados actualmente para llegar a la selección del más apropiado para el desarrollo de la aplicación.

1.11.1. Symfony

Es un *framework* diseñado para optimizar el desarrollo de aplicaciones web gracias a sus diversas características. Separa las reglas de negocio de la aplicación, la lógica del servidor y las vistas de presentación, lo cual constituye un aspecto importante a la hora de desarrollar cualquier solución. Cuenta con muchas herramientas y clases orientadas a disminuir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes para que el desarrollador pueda ocuparse completamente de los aspectos específicos de cada aplicación. Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales. Es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft (40).

A continuación se hace mención de las principales características del *framework*:

- Gran facilidad de instalación y configuración en la mayoría de las plataformas de Linux y Windows.
- Independiente del manejador de base de datos: utiliza Propel, una capa de abstracción que le permite interactuar con varias bases de datos.
- Sencillo de usar en la mayoría de los casos y lo suficientemente flexible para adaptarse a escenarios complejos.
- Se rige por las mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas de arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Manejo de memoria caché: lo cual reduce el uso del ancho de banda y la carga en servidor.
- Contiene código fácil de leer, el cual incluye comentarios de phpDocumentor y formas de adaptarlo en forma sencilla.
- Mecanismos de autentificación y credenciales: que facilitan la creación de secciones restringidas y la gestión de seguridad de usuarios.

- URLs inteligentes: que permiten que las direcciones de las páginas web sean parte de la interfaz y resulten amigables a los motores de búsqueda.
- Fácil de extender, lo que posibilita integrarse con librerías desarrolladas por terceros.
- Incorpora herramientas que facilitan la prueba y depuración de aplicaciones: como unidades de generación de código, pruebas del funcionamiento del *framework,* panel de depuración, interfaz por línea de comandos y configuración en tiempo real.

1.11.2. Yii

Yii es un *framework* para PHP de alta performance (rendimiento) basado en componentes web para desarrollar aplicaciones de gran escala. Permite una máxima reusabilidad en la programación web y puede acelerar significativamente el proceso de desarrollo. El nombre Yii (pronunciado/i:/) proviene de facilidad, eficiencia y extensión.

Es un *framework* genérico para programar web que puede ser utilizado para desarrollar virtualmente cualquier tipo de aplicaciones web. Ya que es liviano y está equipado con las soluciones más sofisticadas, está especialmente diseñado para trabajar con aplicaciones web de tráfico alto, como portales, foros, sistemas de administración de contenidos (CMS), sistemas de comercio electrónico (*e-commerce*), y otros.

Este implementa el patrón de diseño Modelo Vista Controlador *(Model-View-Controller* MVC) el cual es adoptado ampliamente en la programación web. MVC tiene por objeto separar la lógica del negocio de las consideraciones de la interfaz de usuario para que los desarrolladores puedan modificar cada parte más fácilmente sin afectar a la otra. En MVC, el modelo representa la información (los datos) y las reglas del negocio; la vista contiene elementos de la interfaz de usuario como textos, formularios de entrada; y el controlador administra la comunicación entre la vista y el modelo. Más allá del MVC, Yii también introduce un *front-controller* llamado aplicación, el cual representa el contexto de ejecución del procesamiento del pedido. La aplicación resuelve el pedido del usuario y las despacha al controlador apropiado para tratamiento futuro.

Las principales características del *framework* Yii se mencionan a continuación:

- Fácil de instalar.
- Puede correr en modo debug o en modo producción, acorde al valor de la constante YII_DEBUG, por defecto toma modo producción.
- Liviano de correr y equipado con soluciones de cacheo sofisticadas.

- Permite máxima reutilización.
- Ha sido probado en Apache HTTP server en Windows y en Linux. Pero también deberá correr en otros servidores web o plataformas que soporten PHP 5.

1.11.3. Selección del marco de trabajo para la manipulación de PHP

Se elige como marco de trabajo del lado del servidor a Yii debido a que destaca sobre los otros *frameworks* PHP, por su eficiencia y su rica librería de funcionalidades así como también su clara documentación. Yii ha sido diseñado cuidadosamente desde el principio para el trabajo con sistemas de tráfico alto y utilizado para desarrollar virtualmente cualquier tipo de aplicaciones web. Es sencillo de instalar y permite una máxima reutilización. Como la mayoría de los *framework* para PHP, este sigue el patrón de diseño MVC. Yii es el resultado de la amplia experiencia que tienen sus autores en el desarrollo de aplicaciones web.

1.12. Entorno de Desarrollo Integrado (IDE)

Un Entorno de Desarrollo Integrado (IDE) es un programa compuesto por una serie de herramientas que utilizan los programadores para desarrollar código. Ofrece un entorno para programar, en uno o varios lenguajes de programación, mediante un grupo de herramientas que así lo permiten. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, se compone de un editor de texto, un compilador, un intérprete, un depurador, un cliente; algunos brindan un sistema de control de versiones y son factibles para apoyar en la construcción de interfaces gráficas de usuario. Los IDE pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes (41).

1.12.1. NetBeans

Netbeans es una aplicación gratuita y de código abierto pensada para escribir, compilar, depurar y ejecutar programas. Es un entorno de desarrollo hecho principalmente para Java aunque puede ser utilizado para cualquier otro lenguaje, así como PHP, JavaScript, Ajax y otros.

Tiene soporte para crear interfaces gráficas de forma visual, crear aplicaciones para móviles y desarrollar aplicaciones web con elevado contenido gráfico. Todas estas funcionalidades mencionadas pueden ser extendidas mediante la instalación de paquetes adicionales.

El IDE NetBeans está apoyado por una comunidad de desarrolladores y ofrece una amplia documentación y recursos de capacitación, así como una gran cantidad de plugins de terceros. Es fácil de instalar y utilizar, además, se ejecuta en muchas plataformas, incluyendo Windows, Linux, Mac y Solaris. Este IDE proporciona plantillas de código, sugerencias de codificación y herramientas de refactorización. Brinda diferentes vistas de los datos y herramientas útiles para la creación de aplicaciones y una gestión eficiente de las mismas, lo que le permite profundizar en sus datos de forma rápida y sencilla (41).

1.12.2. Zend Studio

Zend Studio o Zend Development Environment es un completo entorno de desarrollo integrado para el lenguaje de programación PHP. Está escrito en Java, y está disponible para las plataformas Microsoft Windows, Mac OS X y GNU/Linux. Incluye editor, análisis, depuración, optimizadores de código y herramientas de base de datos. Zend Studio fue diseñado para usarse con el lenguaje PHP; sin embargo ofrece soporte básico para otros lenguajes web, como HTML, Javascript y XML. Junto con su contraparte *Zend Platform*, son la propuesta de *Zend Technologies* para el desarrollo de aplicaciones web utilizando PHP, actuando Zend Studio como la parte cliente y *Zend Platform* como la parte servidora (42).

Zend Studio muestra varias características que se citan a continuación (42):

- No requiere la instalación previa de PHP ni del entorno de ejecución de Java.
- Soporte para PHP 4 y PHP 5, con phpDoc integrado.
- Resaltado de sintaxis, autocompletado de código, ayuda de código y lista de parámetros de funciones y métodos de clase.
- Inserción automática de paréntesis y corchetes de cierre.
- Detección de errores de sintaxis en tiempo real.
- Instalación de barras de herramientas para Internet Explorer y Mozilla Firefox.
- Soporte para gestión de grandes proyectos de desarrollo.
- Manual de PHP integrado.
- Soporte para control de versiones usando CVS o Subversión.
- Cliente FTP integrado.
- Soporte para navegación en bases de datos y ejecución de consultas SQL.

1.12.3. Selección del entorno de desarrollo integrado

Se decide hacer uso de NetBeans como entorno de desarrollo integrado por ser una herramienta que le ofrece facilidades al desarrollador a la hora de programar ya que brinda una amplia documentación y recursos de capacitación. Además es un IDE muy potente, de código abierto, y disponible para Windows, Linux y otros sistemas operativos. NetBeans es muy fácil de usar y cuenta con el autocompletado y el resaltado de código. También se tuvo presente su integración con el *framework* seleccionado. Estas características convierten a NetBeans en el IDE más indicado para el desarrollo del sistema.

Consideraciones Finales

El análisis de los sistemas homólogos permitió determinar que ninguno soluciona totalmente la problemática planteada, se analizaron las potencialidades que pudiesen ser incorporadas a la solución y se detectaron las tendencias actuales en el desarrollo de este tipo de software. Se realizó un estudio del marco teórico en el que se fundamenta el desarrollo de la plataforma informática propuesta, además de las herramientas y tecnologías que se utilizarán, quedando escogido el Visual Paradigm para modelar el sistema, UML como lenguaje de modelado, el uso de la metodología de desarrollo de software XP, entre los lenguajes a utilizar se pueden encontrar HTML, JavaScript, CSS y PHP para la programación del lado del servidor. También se utilizó NetBeans para la gestión de código fuente, como marco de trabajo Yii 1.1.13 y Apache 2 como servidor web.

2. CAPÍTULO 2. CARACTERÍSTICAS Y DISEÑO DEL SISTEMA

El presente capítulo estará dirigido a la elaboración de la propuesta de diseño del sistema a desarrollar. Se profundizará en cada una de las etapas de Exploración, Planificación e Iteraciones propias de la metodología XP que es utilizada. En la primera etapa se definirán las historias de usuario las cuales describirán las funcionalidades que brindará el sistema. En la fase que le sigue se describen los requisitos funcionales y no funcionales que deberá cumplir la solución, además se especifica el plan de iteraciones y el plan de entrega. En la etapa de iteraciones se utilizan técnicas como las tarjetas CRC (clase, responsabilidad y colaboración) para diseñar las aplicaciones.

2.1. Exploración

En la fase de exploración es donde se realiza el proceso de identificación de las Historias de Usuario (HU o UH, del inglés *User Histories*), estas constituyen uno de los artefactos más significativos que se generan en la metodología, pues son la forma en que se especifican los requisitos del sistema. En esta etapa los clientes definen sus necesidades a través de las HU, y los desarrolladores se familiarizan con las herramientas y tecnologías que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. Dependiendo del tamaño del proyecto y la experiencia que tengan los programadores con las tecnologías esta etapa toma de pocas semanas a pocos meses.

2.1.1. Historias de Usuario

Las HU son la técnica que utiliza la metodología XP para especificar los requisitos del sistema. Estas se elaboran en forma de tabla, desde la perspectiva del cliente, el cual describe concretamente las características que el software debe tener. Aunque los desarrolladores pueden ofrecer su ayuda en la identificación de las mismas. Tienen la misma finalidad que los casos de usos pero con la diferencia de que contienen algunas líneas escritas por el cliente en un lenguaje no técnico, sin centrarse en los detalles, donde expresan de modo sencillo las necesidades del sistema. El tratamiento de las HU es muy dinámico y flexible, ya que estas pueden ser modificadas o eliminadas, reemplazarse por otras más generales; o añadirse nuevas. Las HU son descompuestas en tareas de programación para que los programadores puedan implementarla durante una iteración. XP no propone un formato específico para las HU pero existen varias planillas sugeridas con respecto a la información contenida en las mismas (22).

Se obtiene un total de 55 HU, estas se realizan en 4 iteraciones donde cada una de ellas estará dando respuesta a las dísimiles funcionalidades que el cliente ha solicitado. Son elegidas las HU de mayor importancia. Teniendo en cuenta la amplitud del sistema propuesto, se seleccionan 6 de estas HU.

Tabla 1: *HU1 Autenticar usuario.*

Historia de Usuario	
Número: 1	**Usuario:** Todos
Nombre de Historia de Usuario: Autenticar usuario.	
Prioridad en Negocio: Muy Alta	**Iteración Asignada:** 1
Riesgo en Desarrollo: Alto	**Puntos Estimados:** 2 días
Descripción: El sistema debe permitir la autenticación de usuario para acceder al entorno de trabajo del mismo mediante usuario y contraseña. Se debe poner en práctica el Protocolo Ligero de Acceso a Directorios (LDAP) de la universidad, para la verificación de usuarios en el dominio. Automáticamente al usuario autenticado se le asigna el rol de usuario.	
Prototipo de Interfaz 	

Tabla 2: *HU11 Listar las categorías de un activo de tipo media.*

Historia de Usuario	
Número: 11	**Usuario:** Todos
Nombre de Historia de Usuario: Listar las categorías de un activo de tipo media.	
Prioridad en Negocio: Muy Alta	**Iteración Asignada:** 1
Riesgo en Desarrollo: Medio	**Puntos Estimados:** 2 días
Descripción: El sistema debe permitir listar las categorías de los activos. Si se trata de un modelo ir al menú y presionar la opción Modelo, donde encontrará el listado de todas las categorías.	

Prototipo de Interfaz

Tabla 3: *HU14 Subir modelo.*

Historia de Usuario	
Número: 14	**Usuario:** Proveedor, Publicador y Administrador
Nombre de Historia de Usuario: Subir modelo.	
Prioridad en Negocio: Alta	**Iteración Asignada:** 2
Riesgo en Desarrollo: Alto	**Puntos Estimados:** 2 días
Descripción: El sistema debe permitir subir modelos. Para ello dar en la opción subir modelo y mostrará la interfaz que se expone a continuación.	
Prototipo de Interfaz 	

Tabla 4: *HU30 Buscar activo de media.*

Historia de Usuario	
Número: 30	**Usuario:** Todos
Nombre de Historia de Usuario: Buscar activo de media.	
Prioridad en Negocio: Media	**Iteración Asignada:** 3
Riesgo en Desarrollo: Medio	**Puntos Estimados:** 2 días
Descripción: El sistema debe permitir la búsqueda de cualquier activo de media, como modelo, textura, video y audio. Para ello poner el nombre del activo o el usuario que lo subió.	

Prototipo de Interfaz

Tabla 5: *HU32 Aceptar activo de media.*

Historia de Usuario	
Número: 32	**Usuario:** Publicador y Administrador
Nombre de Historia de Usuario: Aceptar activo de media.	
Prioridad en Negocio: Media	**Iteración Asignada:** 3
Riesgo en Desarrollo: Medio	**Puntos Estimados:** 2 días
Descripción: El sistema debe permitir aceptar cualquier activo de media, como modelo, textura, video y audio. Para ello el usuario permitido debe dar clic en la opción *Aprobar archivo compartido.*	
Prototipo de Interfaz 	

Tabla 6: *HU39 Evaluar activo de media según el gusto.*

Historia de Usuario	
Número: 39	**Usuario:** Todos
Nombre de Historia de Usuario: Evaluar activo de media según el gusto.	
Prioridad en Negocio: Media	**Iteración Asignada:** 3
Riesgo en Desarrollo: Media	**Puntos Estimados:** 3 días
Descripción: El sistema debe permitir evaluar un activo de media según el gusto del usuario. Para ello presionar la opción *Clic si te gusta la textura.*	

Prototipo de Interfaz

2.2. Planificación

En esta fase el cliente establece la prioridad de cada HU, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente (22).

Para realizar una acertada estimación de la aplicación es necesario estimar el tiempo que les tomará la codificación de cada una de las HU. El tiempo de estimación ideal de las HU del sistema, puede extenderse un poco más debido a actividades imprevistas y la no dedicación de tiempo completo. Cada una de las historias se implementarán a partir de su prioridad en sus correspondientes iteraciones.

2.2.1 Lista de reservas del producto

La lista de reserva del producto es un artefacto que recoge los requisitos funcionales (RF) y no funcionales (RNF) de una aplicación, los mismos se muestran a continuación.

2.2.1.1 Requisitos funcionales

RF1: Autenticar usuario (usuario, contraseña).

RF2: Modificar rol de usuario (solapín, nombre, apellidos, rol, usuario, correo, fecha_de_creación).

RF3: Eliminar Usuario.

RF4: Listar usuarios.

RF5: Mostrar datos del usuario.

RF6: Buscar usuario (solapín, nombre, apellidos, rol, usuario, correo).

RF7: Descargar reporte en pdf de usuarios registrados.

RF8: Adicionar categoría de un activo de tipo media (nombre, subir_imagen).

RF9: Modificar la categoría de un un activo de tipo media (nombre, subir_imagen).

RF10: Eliminar la categoría de un un activo de tipo media.

RF11: Listar las categorías de un activo de tipo media.

RF12: Mostrar datos detallados de la categoría de un activo de tipo media.

RF13: Cerrar sesión.

RF14: Subir modelo (nombre_del_archivo, herramienta_de_desarrollo, categoría_del_modelo, descripción_del_archivo, subir_archivo, subir_imagen).

RF15: Descargar modelo.

RF16: Listar todos los modelos.

RF17: Mostrar datos detallados del modelo.

RF18: Subir video (nombre_del_video, categoría_del_video, descripción_del_video, subir_video).

RF19: Descargar video.

RF20: Listar todos los videos.

RF21: Mostrar datos detallados del video.

RF22: Subir textura (nombre_de_la_textura, categoría_de_la_textura, subir_comprimido, imagen_representativa).

RF23: Descargar textura.

RF24: Listar todas las texturas.

RF25: Mostrar datos detallados de las texturas.

RF26: Subir audio (nombre_del_audio, categoría, subcategoría, subir_audio).

RF27: Descargar audio.

RF28: Listar todos los audios.

RF29: Mostrar datos detallados de los audios.

RF30: Buscar activo de media (nombre_del_activo, usuario).

RF31: Filtrar búsqueda de un activo de tipo media (categorías, estados).

RF32: Aceptar activo de media.

RF33: Denegar activo de media.

RF34: Enviar notificación.

RF35: Añadir comentario de un activo de media.

RF36: Aceptar comentario de un activo de media.

RF37: Eliminar comentario de un activo de media.

RF38: Listar comentarios de un activo de media.

RF39: Evaluar activo de media según el gusto.

RF40: Generar reporte de activos más descargados.

RF41: Generar reporte de activos más visitados.

RF42: Generar reporte de activos con más votos.

RF43: Generar reporte de activos más comentados.

RF44: Generar reporte de usuarios que más aportan activos sin descargar.

RF45: Generar reporte de usuarios que más descargan activos sin aportar.

RF46: Generar reporte de usuarios que más aportan activos y a la vez tienen alto índice de descargas.

RF47: Generar reporte de usuarios que menos aportan activos y a la vez tienen bajo índice de descargas.

RF48: Generar reporte de usuarios que más publican activos.

RF49: Generar reporte de usuarios que más comentan activos.

RF50: Generar reporte de usuarios que publican todo tipo de activos.

RF51: Añadir solicitud (nombre_de_la_solicitud, descripción_de_la_solicitud).

RF52: Listar solicitudes.

RF53: Mostrar datos detallados de la solicitud.

RF54: Responder a la solicitud.

RF55: Eliminar solicitud.

2.2.1.2 Requisitos no funcionales

Los requisitos no funcionales, son las propiedades o cualidades que el producto debe tener, es decir, las características del producto a desarrollar. A continuación se exponen los requerimientos:

Usabilidad:

- Permitir una navegación sencilla, esto se logrará a partir de una estructura de la información correcta, en

todo momento los usuarios tendrán conocimiento del lugar donde se encuentra en el sistema a través del uso de títulos, subtítulos. Los enlaces y botones serán fáciles de asociar con las operaciones que realizan.

Confiabilidad:

- Ante cualquier evento, ya sea correcto o incorrecto, se mostrará un mensaje informativo.

Restricciones de diseño:

- Lenguaje de programación a utilizar PHP bajo el IDE NetBeans.

Rendimiento:

- Los tiempos de respuesta y velocidad de procesamiento de la información serán rápidos, no mayores de 3 segundos para las actualizaciones y 8 para las recuperaciones.

Software:

- **Para el cliente:**

- Utilizar navegador (Mozilla Firefox 36.0.1 o Google Chrome 37.0.2062.94).
- Sistema operativo multiplataforma.

- **Para el servidor:**

- Sistema operativo Windows o Linux en cualquiera de sus distribuciones.
- Servidor de base de datos MySQL.
- Servidor con módulo PHP disponible (WAMP, XAMPP, u otros).

Hardware:

- **Para el servidor:**

- Capacidad de disco duro superior a 80 Giga bytes. Se requiere un mínimo de 2 GB de RAM y 2.1 GHz de velocidad de procesamiento.

- **Para el cliente:**

- Utilizar Capacidad de disco duro superior a 50 Giga bytes. Se requiere un mínimo de 256 MB de RAM y 1.6 GHz de velocidad de procesamiento.

Seguridad:

- Los usuarios deben autenticarse antes de interactuar con el sistema.
- El sistema deberá garantizar el acceso controlado a la información. Este debe influir en la forma de presentar las interfaces para cada usuario dependiendo del nivel de acceso a la información.

- La autenticación de un usuario en el sistema debe realizarse utilizando el certificado SSL.
- El sistema debe desloguearse al estar más de 10 minutos inactivo.

Interfaz:

- La interfaz gráfica uniforme incluyendo pantallas, menús y opciones. La consistencia de la interacción entre usuario y componente estará determinada por el diseño de las interfaces de usuario que mantendrán los elementos como menús y zonas de trabajo, en posiciones fijas, además de la mayor uniformidad posible entre cuadros de texto y botones. Tanto los títulos de los componentes de la interfaz, como los mensajes para interactuar con los usuarios, así como los mensajes de error, deberán ser en idioma español y tener una apariencia uniforme.

2.2.2 Plan de Iteraciones

Luego de conocer las HU y determinar la estimación del esfuerzo por cada una de ellas, se comienza a realizar el plan de iteraciones. En el mismo estarán contenidas las HU, teniendo en cuenta su prioridad en el desarrollo de la herramienta se decide en que iteración será implementada. Las HU con mayor importancia por ser funcionalidades indispensables para la aplicación deben ser implementadas en las primeras iteraciones.

A continuación se muestra el plan dividido en 4 iteraciones con la HU en el orden a realizar por cada iteración según su orden de relevancia, así como la descripción y el total de semanas que durarán cada una de estas.

Tabla 7: *Plan de iteraciones.*

Iteración	Descripción de la iteración	Orden de la HU a implementar	Duración total
1	Las HU que tienen prioridad de negocio muy alta se implementan en la primera iteración, ya que estas funcionalidades son fundamentales para cubrir las necesidades del cliente. Además son imprescindibles para lograr una buena base, funcionamiento y desarrollo de la aplicación.	1 al 13	21 días
2	Las HU de prioridad de negocio alta se implementan en la segunda iteración, las cuales son tan importantes como las	14 al 29	24 días

	citadas anteriormente pero debido a la cantidad de HU se decidió realizarlas en otra iteración.		
3	Las HU de prioridad de negocio media se implementan en la tercera iteración, ya que complementan a las HU de la iteración anterior aunque son de menor importancia.	30 al 39	21 Días
4	Las HU de prioridad de negocio baja se implementan en la cuarta iteración, ya que las mismas son las de menor interés para el cliente.	40 al 55	21 Días

2.2.3 Plan de Entrega

Una vez definido el plan de iteraciones se procede a realizar el plan de entrega, el cual tiene como objetivo dar una propuesta de las fechas aproximadas de culminación de las iteraciones y sus correspondientes cantidades de HU. A continuación se presenta el Plan de entregas del sistema:

Tabla 8: *Plan de Entrega.*

Iteración	Iteración # 1	Iteración # 2	Iteración # 3	Iteración # 4
Cantidad de HU	13	16	10	16
Fecha de entrega	25/03/2015	20/04/2015	10/05/2015	1/06/2015

2.3 Iteraciones

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. Al final de la última iteración el sistema estará listo para entrar en producción (22).

Diseño del sistema

Para el diseño de aplicaciones, la metodología XP no requiere la representación del sistema mediante diagramas de clases utilizando UML, aunque para una mejor comprensión del problema se podría incluir dentro de los artefactos generados en el ciclo de vida del proyecto. En su lugar se usan otras técnicas: las tarjetas CRC (clase, responsabilidad y colaboración) como una extensión informal a UML las cuales se especifican más adelante.

2.3.1 Tarjetas CRC

Para el desarrollo de la aplicación una de las herramientas de diseño empleada fueron las tarjetas CRC (clase, responsabilidad y colaboración). Normalmente son usadas cuando se determina primero las clases que se necesitan, su interacción y después se implementa la solución. Las mismas se dividen en tres secciones: el nombre de la clase, sus responsabilidades y sus colaboradores. Una clase describe cualquier objeto o evento, mediante los atributos y los métodos, las responsabilidades son las tareas que realizan o los métodos correspondientes a la clase y los colaboradores son las demás clases con las que requiere relación para cumplir con sus responsabilidades. La estructura de una tarjeta CRC se describe brevemente a continuación: a modo de título se escribe en la tarjeta el nombre de la clase, se coloca a la izquierda todas las responsabilidades, y a la derecha las clases implicadas en cada responsabilidad, en la misma línea que su requerimiento correspondiente.

Una sesión CRC es una técnica donde todo el equipo se reúne, dichas tarjetas CRC en blanco son puestas en una mesa de trabajo y mediante la retroalimentación de ideas el equipo es el encargado de escribir los datos pertenecientes a los nombres de las clases, sus responsabilidades y colaboraciones. Durante la sesión las tarjetas son movidas de lugar y las responsabilidades y colaboraciones pueden variar también. Esta técnica permite la total interacción de los miembros del equipo contribuyendo a la solución de un problema de diseño. El objetivo que se persigue es lograr que todos los participantes aporten sus mejores ideas para el desarrollo del sistema. Además con la finalidad de obtener un diseño simple y evitar la implementación de características que no son necesarias.

Tabla 9: *Tarjeta CRC SolicitudController.*

Tarjeta CRC	
Clase: SolicitudController	
Responsabilidades	**Colaboraciones**
-Crear solicitud. -Eliminar solicitud. -Listar solicitudes. -Mostrar en una vista detallada los datos de una solicitud. -Responder a una solicitud.	Solicitud. ResponderSolicitud.

Tabla 10: *Tarjeta CRC UsuarioController.*

Tarjeta CRC	
Clase: UsuarioController	
Responsabilidades	**Colaboraciones**
-Crear usuario. -Modificar rol de usuario -Eliminar usuario. -Listar usuario. -Mostrar en una vista detallada los datos de un usuario. -Generar documentos en formato pdf de los usuarios registrados en el sistema.	Usuario.

Tabla 11: *Tarjeta CRC AlmacenModeloController.*

Tarjeta CRC	
Clase: AlmacenModeloController	
Responsabilidades	**Colaboraciones**
-Mostrar en una vista detallada los datos de un modelo, y los comentarios realizados a cada uno de ellos. -Votar si te gusta un modelo. -Votar si no te gusta un modelo. -Descargar un modelo.	AlmacenModelo. CompartirModelo. ComentarioModelo. VisitaModelo. MegustaModelo. DescargarModelo.

Tabla 12: *Tarjeta CRC ComentarioModeloController.*

Tarjeta CRC	
Clase: ComentarioModeloController	
Responsabilidades	**Colaboraciones**
-Crear comentario. -Eliminar comentario. -Listar comentarios. -Aprobar comentario. -Votar si te gusta el cometario. -Votar si no te gusta el comentario.	ComentarioModelo. VotoComentarioModelo.

Tabla 13: *Tarjeta CRC SiteController.*

Tarjeta CRC	
Clase: SiteController	
Responsabilidades	**Colaboraciones**
-Verificar que los usuarios que se autentiquen estén en el dominio (uci). -Cerrar la sesión del usuario autenticado.	LoginForm. Usuario.

2.4 Arquitectura de software

Siguiendo la definición que ofrece el Instituto de Ingenieros Eléctricos y Electrónicos, más conocido como la IEEE (*Institute of Electrical and Electronics Engineers*): "*La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución"* (43).

Durante el desarrollo de un software frecuentemente se encuentran distintos puntos de vista entre el equipo de trabajo responsable de la construcción del sistema, la arquitectura es el instrumento encargado de unificar estas diferencias. La misma tiene como propósito principal brindar elementos que ayuden a la toma de decisiones y proporcionar un lenguaje común. La arquitectura del software para lograrlo se enfoca en seleccionar y combinar estilos y patrones arquitectónicos.

Como propuesta de solución para el desarrollo de la aplicación se determinó el patrón Modelo-Vista-Controlador (MVC), el cual permite un mejor soporte para futuras actualizaciones del producto, dividiendo la aplicación en tres componentes distintos: el modelo, la vista y el controlador de forma tal que los cambios ocurridos en una de estas capas, no afecte las demás.

2.4.1 Patrón Modelo Vista Controlador (MVC)

El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio. Este no tiene conocimiento específico de los Controladores o de las Vistas, ni siquiera contiene referencias a ellos. Además responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador). Las principales responsabilidades de los modelos son, acceder a la base de datos y llevar un registro de las vistas del sistema, para que sea capaz de notificar a estas sobre los cambios que pueda producir un agente externo

a los datos. El *framework* Yii implementa dos tipos de modelos: modelo de formulario y active record (registro activo). A continuación una muestra de la arquitectura en la capa modelo.

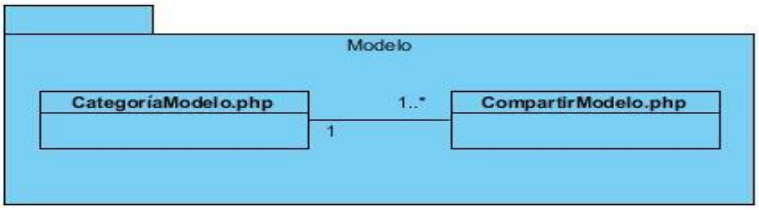

Figura 1: *Aplicación de la arquitectura en la capa Modelo.*

La vista es el elemento que maneja la presentación visual de los datos representados por el Modelo. Esta transforma el modelo en una interfaz que posibilita que el usuario pueda interactuar con ella. Una vista está asociada a un modelo, pero se pueden encontrar varias vistas relacionadas con un mismo modelo. Interactúa preferentemente con el Controlador y es responsable de tener un registro de su controlador asociado. Es la encargada de recibir datos procesados por el controlador o del modelo y visualizarle la información al usuario. A continuación una muestra de la arquitectura en la capa vista.

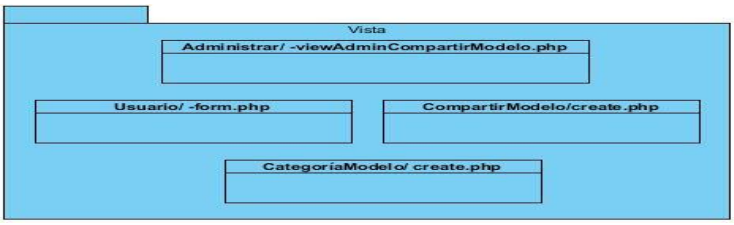

Figura 2: *Aplicación de la arquitectura en la capa Vista.*

El controlador es el encargado de procesar las acciones del usuario y responder a los mismos realizando los cambios adecuados en el modelo o en la vista. El mismo es una especie de capa intermedia que conecta la vista con el modelo. Es responsable de implementar la funcionalidad de la aplicación, es decir la lógica del negocio y permitir la validación de los datos. El Controlador tiene la tarea de recibir los eventos de entrada como datos introducidos por una persona u opciones del menú seleccionadas por ella. Interpreta los movimientos del ratón y el teclado e informa al modelo y/o a la vista para que cambien según resulte apropiado. A continuación una muestra de la arquitectura en la capa controlador.

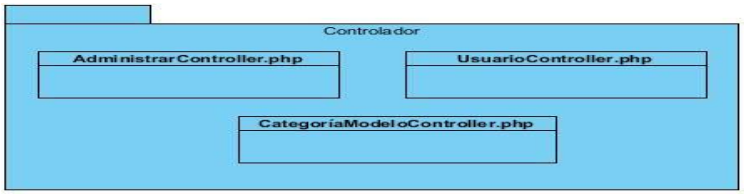

Figura 3: *Aplicación de la arquitectura en la capa Controlador.*

2.5 Patrones de diseño

Un patrón de diseño es una solución repetible, generalmente ya probada en problemas recurrentes que se dan durante el proceso de diseño de un software. Esta solución puede ser utilizada en diferentes situaciones, no es un diseño terminado que puede traducirse directamente a código, sino más bien una descripción detallada sobre cómo resolver el problema. Cada patrón de diseño se centra en un problema concreto, describiendo cuándo aplicarlo y si tiene sentido hacerlo teniendo en cuenta otras restricciones de diseño, así como las consecuencias, ventajas e inconvenientes de su uso. Con la utilización de los patrones los diseños serán mucho más flexibles, modulares y reutilizables. Emplean un conjunto de buenas prácticas simplificando así el trabajo, y facilitan la comprensión y aprendizaje de personas ajenas al sistema al formalizar un vocabulario común. Es importante conocer y estudiar los diferentes patrones que existen para poder determinar cuál de ellos usar ya que presentan diferentes soluciones (44).

2.5.1 Patrones GRASP

Los Patrones de Software para la Asignación General de Responsabilidad GRASP (*General Responsabilty Assignment Software Patterns*) son patrones de diseño. Estos describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones. A continuación se muestran los que fueron utilizados en la implementación de la aplicación:

Creador: Ayuda a identificar quién debe ser el responsable de la creación o instanciación de nuevos objetos o clases. Es el encargado de almacenar o manejar varias instancias de la clase. Se ocupa de guiar la asignación de responsabilidades relacionadas con la creación de los objetos, encontrando un creador para cada objeto. La siguiente figura muestra un ejemplo del patrón creador en el sistema propuesto (45).

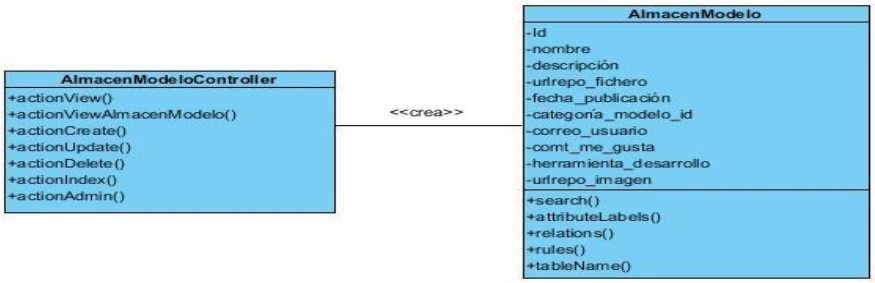

Figura 4: *Diagrama de clases que representa el Patrón Creador.*

Controlador: Es el encargado de asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. También es conocido como un objeto de interfaz no destinado al usuario que se encarga de manejar un evento del sistema. En el diseño orientado a objeto cuando el sistema recibe eventos de entrada externa, hay que elegir los controladores que manejen estos eventos. El patrón controlador aumenta el potencial de reutilización y garantiza que los procesos sean manejados por la capa controladora y no por la capa de presentación. La siguiente figura muestra un ejemplo del patrón controlador en el sistema propuesto (45).

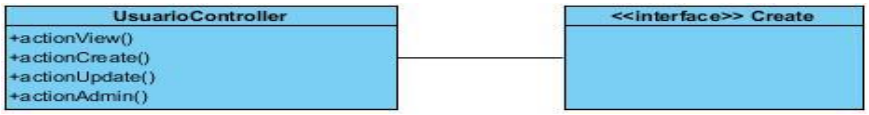

Figura 5: *Diagrama de clases que representa el Patrón Controlador.*

Experto: es un patrón que sugiere que siempre se debe asignar una responsabilidad al experto en información, o sea, la clase que cuenta con la información necesaria para cumplir una tarea debe ser la responsable de ejecutar la misma. Es un principio básico que suele utilizarse en el diseño orientado a objetos (45).

En el desarrollo de la aplicación se tiene en cuenta el cumplimiento de este patrón, en las vistas se define todo lo mostrado al usuario, en los modelos se garantizan las validaciones y en los controladores se desarrollan las funcionalidades. La aplicación del patrón Experto permitió la correcta distribución del comportamiento de las clases encargadas de manejar la información del sistema. Este patrón se puede evidenciar en las clases controladoras que funcionan como experta en información, por ejemplo la clase CategoriaModeloController funciona como experta en información para llevar a cabo el trabajo con las categorías de los modelos, es la responsable de crear, modificar, eliminar y listar una categoría, esta clase

obtiene toda la información necesaria para trabajar con los datos relacionados con las categorías de los modelos. La siguiente figura muestra un ejemplo del patrón experto en el sistema propuesto.

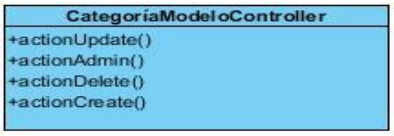

Figura 6: *Diagrama de clases que representa el Patrón Experto.*

Bajo Acoplamiento: el patrón bajo acoplamiento estimula la asignación de responsabilidades de manera que su localización no incremente el acoplamiento hasta un nivel que produzca resultados negativos propios de un alto acoplamiento. Soporta el diseño de clases más independientes, que reducen el impacto de los cambios. El mismo no se puede considerar de manera aislada a otros patrones como el Experto o el de Alta Cohesión, sino que necesita incluirse como uno de los diferentes principios de diseño que influyen en la desición de asignar responsabilidades. Es necesario tener un bajo acoplamiento cuando las clases son muy genéricas y con grandes probabilidades de reutilización. Si el bajo acomplamiento se lleva a los extremos dará origen a un diseño deficiente por producir objetos incoherentes y complejos, que hacen todo el trabajo. Es necesario un grado moderado de acoplamiento entre las clases si se quiere crear un sistema orientado a objeto, donde las tareas se realicen por colaboración entre objetos conectados (45).

Para alcanzar un bajo acoplamiento las clases controladoras heredan únicamente de la clase *Controller.*

Alta Cohesión: es un principio evaluativo que aplica un diseñador mientras evalúa todas las decisiones de diseño y es la meta principal que debe buscarse en todo momento. Este patrón expresa que se debe asignar a las clases responsabilidades que trabajen sobre una misma área de la aplicación y que no tengan mucha complejidad. Una clase con alta cohesión ante una tarea de gran magnitud colabora con otros objetos y se comparten el esfuerzo. Es útil tener clases con alta cohesión ya que es más fácil darle mantenimiento, entenderla y reutilizarla. Optimizan la claridad y la facilidad con que se entiende el diseño. Además se simplifican las mejoras en funcionalidad (45).

Para la implementación de la aplicación se hizo uso de las formas de organización que brinda Yii. El mismo permite la organización del trabajo en cuanto a la estructura del proyecto y la asignación de responsabilidades con una alta cohesión. Un ejemplo de ello es la clase UserIdentity, la cual necesita instanciar un objeto de la clase Ldap, de esta forma delega responsabilidad a dicho objeto para que este acceda a sus métodos y contribuya así a la ejecución de la acción autenticar usuario, en la misma se controla el acceso de los usuarios al sistema.

Consideraciones finales

- Los requisitos del sistema y la arquitectura propuesta permitieron obtener un diseño robusto de la aplicación.
- Las definiciones de historias de usuario permitieron tener la base para la implementación de la aplicación.
- Las definiciones de las tarjetas CRC aportaron en la identificación de las responsabilidades de las clases y las colaboraciones de las mismas.
- Se aplicaron los patrones necesarios para lograr un diseño flexible y eficiente.

3. CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBA

En el presente capítulo se especifican las fases de implementación y prueba, la implementación del sistema perteneciente a la fase Iteraciones y las pruebas realizadas pertenecientes a la fase de Producción. Con el objetivo de validar que el sistema cumpla con lo esperado se llevan a cabo las pruebas de aceptación realizadas a cada una de las HU y se identifican y describen las tareas de ingeniería. Además se muestra el diagrama de despliegue del sistema a implementar.

3.1. Tareas de Ingeniería

Las Tareas de Ingeniería (TI) definen cada una de las actividades que dan cumplimiento a cada HU. Se describen de forma tal que se entienda lo que el sistema tiene que hacer y así facilitar su construcción. Estas tareas pueden escribirse utilizando un lenguaje técnico y no necesariamente deben ser entendibles para el cliente. A continuación se exponen 12 de estas tareas.

Tabla 14: *Tarea de ingeniería 1 de la HU1.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 1
Nombre de la Tarea: Implementar autenticar usuario.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2 días
Fecha de inicio: 2/3/2015	**Fecha Fin:** 4/3/2015
Programador Responsable: Naidiley Vargas Pérez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario pueda autenticarse.	

Tabla 15: *Tarea de ingeniería 1 de la HU8.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 8
Nombre de la Tarea: Implementar adicionar categoría de un activo de media.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2 días
Fecha de inicio: 4/3/2015	**Fecha Fin:** 6/3/2015
Programador Responsable: Naidiley Vargas Pérez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario autorizado pueda adicionar una categoría de un activo de media.	

Tabla 16: *Tarea de ingeniería 1 de la HU11.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 11
Nombre de la Tarea: Implementar listar las categorías de un activo de tipo media.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2 días
Fecha de inicio: 20/3/2015	**Fecha Fin:** 22/3/2015
Programador Responsable: Naidiley Vargas Pérez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario autorizado pueda listar las categorías de un activo.	

Tabla 17: *Tarea de ingeniería 1 de la HU14.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 14
Nombre de la Tarea: Implementar subir modelo.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2 días
Fecha de inicio: 26/3/2015	**Fecha Fin:** 28/3/2015
Programador Responsable: Naidiley Vargas Pérez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario autorizado pueda subir un modelo.	

Tabla 18: *Tarea de ingeniería 1 de la HU15.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 15
Nombre de la Tarea: Implementar descargar modelo.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1 día
Fecha de inicio: 28/3/2015	**Fecha Fin:** 29/3/2015
Programador Responsable: Leonarkis Acosta Suárez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario autorizado pueda descargar un modelo.	

Tabla 19: *Tarea de ingeniería 1 de la HU17.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 17

Nombre de la Tarea: Implementar mostrar datos detallados del modelo.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1 día
Fecha de inicio: 29/3/2015	**Fecha Fin:** 30/3/2015
Programador Responsable: Leonarkis Acosta Suárez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario autorizado pueda ver los datos detallados del modelo.	

Tabla 20: *Tarea de ingeniería 1 de la HU30.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 30
Nombre de la Tarea: Implementar buscar activo de media.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2 días
Fecha de inicio: 20/4/2015	**Fecha Fin:** 22/4/2015
Programador Responsable: Leonarkis Acosta Suárez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario pueda buscar un activo de media.	

Tabla 21: *Tarea de ingeniería 1 de la HU32.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 32
Nombre de la Tarea: Implementar aceptar activo de media.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 2 días
Fecha de inicio: 25/4/2015	**Fecha Fin:** 27/4/2015
Programador Responsable: Leonarkis Acosta Suárez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario autorizado pueda aceptar un activo de media.	

Tabla 22: *Tarea de ingeniería 1 de la HU35.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 35
Nombre de la Tarea: Implementar añadir comentario de un activo de media.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3 días
Fecha de inicio: 1/5/2015	**Fecha Fin:** 4/5/2015
Programador Responsable: Leonarkis Acosta Suárez.	

Descripción: Se implementan las funcionalidades necesarias para que el usuario pueda añadir un comentario de un activo de media.

Tabla 23: *Tarea de ingeniería 1 de la HU39.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 39
Nombre de la Tarea: Implementar evaluar activo de media según el gusto.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 3 días
Fecha de inicio: 7/5/2015	**Fecha Fin:** 10/5/2015
Programador Responsable: Leonarkis Acosta Suárez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario pueda evaluar un activo de media según el gusto.	

Tabla 24: *Tarea de ingeniería 1 de la HU40.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 40
Nombre de la Tarea: Implementar generar reporte de activo más descargado.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1 día
Fecha de inicio: 12/5/2015	**Fecha Fin:** 13/5/2015
Programador Responsable: Naidiley Vargas Pérez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario pueda consultar el activo más descargado.	

Tabla 25: *Tarea de ingeniería 1 de la HU48.*

Tarea de Ingeniería	
Número de Tarea: 1	**Número Historia de Usuario:** 48
Nombre de la Tarea: Implementar generar reporte de usuarios que más publican activos.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 1 día
Fecha de inicio: 25/5/2015	**Fecha Fin:** 26/5/2015
Programador Responsable: Naidiley Vargas Pérez.	
Descripción: Se implementan las funcionalidades necesarias para que el usuario pueda consultar los usuarios que más publican activos.	

3.2. Diseños de casos de prueba

Una de las alternativas que propone XP para validar las soluciones son las pruebas de aceptación, también conocidas como funcionales o de caja negra. Estas son supervisadas por el cliente basándose en los requerimientos planteados en las HU. Además son destinadas a evaluar si se obtuvo la aplicación requerida y su correcto funcionamiento.

Las pruebas de aceptación son creadas a partir de las HU. Se debe especificar uno o varios escenarios de prueba de aceptación para comprobar que una HU ha sido correctamente implementada. El objetivo final es asegurar el cumplimiento de todas las HU y que el sistema sea aprobado por el cliente. La aplicación se considera finalizada cuando todas las HU hayan pasado correctamente sus pruebas de aceptación.

A continuación se presentan algunas pruebas de aceptación realizadas a la solución propuesta y con los resultados obtenidos:

Tabla 26: *Prueba de aceptación para la HU Autenticar usuario.*

Caso de Prueba de Aceptación	
Código:	**Historia de Usuario:** 1
Nombre: Autenticar usuario.	
Descripción: El sistema debe permitir la autenticación de usuario para acceder al entorno de trabajo del mismo haciendo uso de usuario y contraseña.	
Condiciones de ejecución: La página de autenticación de la herramienta debe estar abierta.	
Entrada/Pasos de ejecución: Introducir los datos en los campos (usuario y contraseña). Clic en el botón **Iniciar sesión.**	
Resultado esperado: El usuario debe ser autenticado en el sistema.	
Resultado obtenido: El usuario autenticado satisfactoriamente en el sistema.	
Evaluación de la prueba: Satisfactoria.	

Tabla 27: *Prueba de aceptación para la HU Listar las categorías de un activo de media.*

Caso de Prueba de Aceptación	
Código:	**Historia de Usuario:** 11
Nombre: Listar las categorías de un activo de media.	

Descripción: El sistema debe permitir listar las categorías de un activo de media.
Condiciones de ejecución: El usuario debe estar autenticado en la aplicación.
Entrada/Pasos de ejecución: Seleccionar la opción **Modelo** del menú ubicado en la parte superior. Clic en la opción **Listar Categorías.**
Resultado esperado: Los activos de media deben ser listados en la aplicación.
Resultado obtenido: Todos los activos de media fueron listados.
Evaluación de la prueba: Satisfactoria.

Tabla 28: *Prueba de aceptación para la HU Subir modelo.*

Caso de Prueba de Aceptación	
Código:	**Historia de Usuario:** 14
Nombre: Subir modelo.	
Descripción: El sistema debe permitir subir modelos solo a los usuarios con el rol autorizado.	
Condiciones de ejecución: El usuario debe estar autenticado en la aplicación.	
Entrada/Pasos de ejecución: Seleccionar la opción **Modelo** del menú ubicado en la parte superior. Clic en el botón **Subir Modelo.**	
Resultado esperado: El sistema debe almacenar en la base de datos los valores introducidos del modelo, y mostrar un mensaje de satisfacción al usuario.	
Resultado obtenido: Los datos del modelo se almacenaron en la base de datos y se mostró un mensaje de satisfacción al usuario.	
Evaluación de la prueba: Satisfactoria.	

Tabla 29: *Prueba de aceptación para la HU Buscar activo de media.*

Caso de Prueba de Aceptación	
Código:	**Historia de Usuario:** 30
Nombre: Buscar activo de media.	
Descripción: La búsqueda de los activos de media se realiza introduciendo el nombre del artefacto o nombre de un usuario. Además se pueden listar por criterios de ordenamiento definidos como: los más comentados, los que más votos tienen, los más descargados, los más visitados, y por la fecha de publicación.	
Entrada/Pasos de ejecución: Introducir el nombre de un activo de media o el nombre de un usuario. Clic en el botón **Buscar.**	
Condiciones de ejecución: El usuario debe estar autenticado en la aplicación.	

Resultado esperado: El sistema debe listar el o los activos de media que corresponden con el criterio buscado.
Resultado obtenido: Se obtiene un listado con uno o más activos de media que pertenecen al criterio de búsqueda dado.
Evaluación de la prueba: Satisfactoria.

Tabla 30: *Prueba de aceptación para la HU Aceptar activo de media.*

Caso de Prueba de Aceptación	
Código:	**Historia de Usuario:** 32
Nombre: Aceptar activo de media.	
Descripción: El usuario con el rol de publicador es el encargado de aceptar los activos de media.	
Condiciones de ejecución: El usuario con el rol de publicador debe estar autenticado en la aplicación.	
Entrada/Pasos de ejecución: Seleccionar el modelo que está en estado de revisión y se desea aceptar. Clic en el botón **Aceptar.**	
Resultado esperado: El sistema debe permitir que se le actualice al modelo el estado que estaba en revisión ha aceptado. Además debe mandar una notificación al usuario informándole que el modelo ha sido aceptado.	
Resultado obtenido: Se actualizó el estado del modelo y de esta forma los usuarios pueden acceder al mismo y descargarlo. Por otra parte la notificación fue enviada por correo al usuario que subió el activo.	
Evaluación de la prueba: Satisfactoria.	

Tabla 31: *Prueba de aceptación para la HU Evaluar activo de media según el gusto.*

Caso de Prueba de Aceptación	
Código:	**Historia de Usuario:** 39
Nombre: Evaluar activo de media según el gusto.	
Descripción: El usuario puede ejercer su voto sobre un activo de media según la aceptación que tenga dicho artefacto en el individuo. Para realizar esta acción se da clic en la manito que aparece en cada objeto de diseño cuando se listan estos, si la mano está hacia arriba y se da clic en ella indica que te gusta el artefacto, si está hacia abajo y se da clic indica que no te gusta.	
Condiciones de ejecución: El usuario debe estar autenticado en la aplicación.	
Entrada/Pasos de ejecución: Clic en la manito que aparece para indicar si te gusta o no el activo de media.	
Resultado esperado: El sistema debe mostrar la cantidad de votos que tiene el activo de media por el cual el usuario votó.	

Resultado obtenido: Se obtiene la cantidad de votos que tiene el activo de media por el cual el usuario votó.
Evaluación de la prueba: Satisfactoria.

3.3 Resultados esperados

El sistema propuesto le da solución al problema planteado inicialmente, ya que ofrecerá gran ayuda para resolver las deficiencias existentes en el centro en cuanto a centralización y seguridad de los activos de tipo media. Además al poder reutilizar los activos se ganará en tiempo y los trabajos serán entregados en un menor período de duración.

Al sistema se le aplicaron pruebas de aceptación y fueron desarrolladas en tres iteraciones. En la primera iteración se obtuvieron 38 pruebas satisfactorias y 17 no conformidades, en una segunda iteración se le dio solución a las no conformidades encontradas anteriormente y se hallaron 9 no conformidades, y en una tercera iteración los 55 casos de prueba de aceptación realizados resultaron satisfactorios. A continuación se muestran estos resultados en la gráfica.

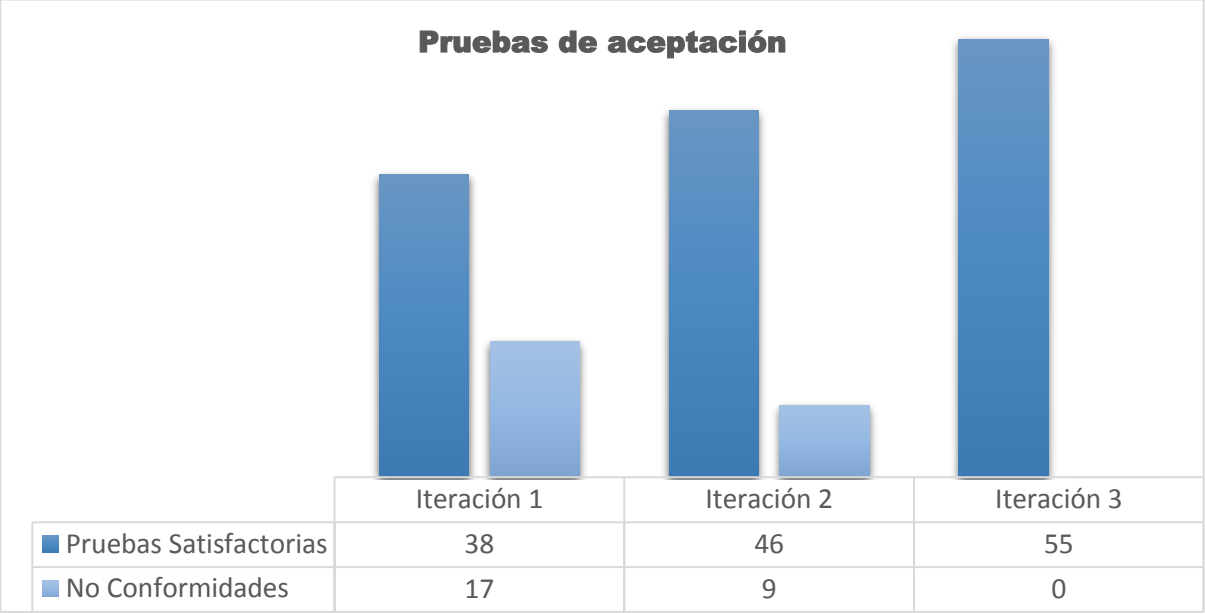

Figura 7: *Resultados de las pruebas realizadas.*

La propuesta de solución fue presentada a la dirección del centro VERTEX con el objetivo de ser validada para su utilización en el proceso de gestión de activos de tipo media. Una vez sometida al juicio de diferentes expertos (ver Anexo 3), el Consejo de Dirección del entorno productivo asume, atendiendo a su pertinencia, aporte y novedad, la *Plataforma colaborativa para la gestión de activos de tipo Media* como la herramienta a ser desplegada para la utilización de sus especialistas en el proceso de desarrollo de software (ver Anexo 4).

3.4 Diagramas de despliegue

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema. Es utilizado para describir la vista de despliegue estática de una aplicación. Describe también la topología del sistema, es decir la estructura de los elementos de hardware y software que ejecuta cada uno de ellos. El diagrama de despliegue muestra la configuración de los elementos de hardware (nodos) y cómo los elementos y artefactos del software se trazan en esos nodos. Se representan con un conjunto de nodos y sus relaciones. Un nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional, como son los procesadores o dispositivos de hardware. Este se relaciona con el diagrama de componente, ya que un nodo normalmente incluye uno o más componentes (46).

A continuación se muestra el diagrama de despliegue del sistema a implementar.

Figura 8: *Diagrama de despliegue.*

- **Estaciones de trabajo con la aplicación cliente:** Está definida por las estaciones de trabajo que el usuario utilizará para acceder a la aplicación web.
- **Servidor de Base de Datos:** Se hace referencia al gestor de bases de datos donde se encuentran los datos necesarios para el trabajo con el sistema. El servidor de base de datos elegido es MySQL.

- **Servidor de aplicación:** El servidor de aplicación es utilizado para la publicación de la aplicación. Es la herramienta principal para ejecutar la lógica de negocio en el lado del servidor.
- **Protocolos de comunicación:** Un protocolo de comunicación es un conjunto de reglas establecidas entre dos dispositivos para permitir la comunicación entre ambos.
  - **Conexión HTTPS:** Protocolo de Trasferencia de Hipertexto Seguro conocido por sus siglas en inglés (HTTPS), es el protocolo utilizado entre los navegadores de los clientes y el servidor web. Este elemento de la arquitectura representa un tipo de comunicación segura y confiable entre clientes y servidor ya que garantiza que cualquier dato introducido será cifrado.

Consideraciones finales

- Con la finalización de la última fase de la metodología se puede contar con una aplicación capacitada para brindar las funcionalidades que fueron descritas en las historias de usuario.
- Se especificaron las tareas de ingeniería, con ellas se agilizó el proceso y se evitó una sobrecarga de trabajo.
- Con las pruebas de aceptación se validó el correcto funcionamiento del sistema, y se comprobó la acertada implementación de las historias de usuario definidas según las necesidades del cliente.

CONCLUSIONES GENERALES

Al completar el desarrollo de la aplicación web se le dio cumplimiento al objetivo planteado inicialmente, alcanzándose los siguientes resultados:

- Esta aplicación contribuye a mejorar la centralidad y seguridad de la información en el proceso de gestión de activos de tipo media en el centro VERTEX.
- El módulo de reportes obtenido permite a la dirección del centro tomar decisiones a partir de datos estadísticos y gestionar otros tipos de activos de diseño como los audios.
- Con esta plataforma se tienen los medios para evitar la pérdida de materiales derivada de los movimientos de personal del centro y las contingencias tecnológicas, las cuales han sido las razones principales de las pérdidas mencionadas.
- Se espera que la puesta en funcionamiento de esta plataforma redunde en un beneficio económico y profesional para el centro por concepto de reutilización, de todos los activos gestionados.

RECOMENDACIONES

Al concluir la presente investigación y teniendo en cuenta los resultados obtenidos se recomienda para futuras versiones:

- Desarrollar una funcionalidad que permita visualizar los modelos 3D en la web.
- Implementar un foro para el debate de los usuarios.
- Agregar nuevos activos de media al sistema.
- Continuar realizando mejoras a la arquitectura de la información, así como al diseño.
- Implementar una funcionalidad que permita filtrar los activos por extensión.

BIBLIOGRAFÍA

1. Villarroel, Miguel. Una Plataforma Colaborativa de Recuperación de Información. 2000.

2. Delgado, Kenneth. Las plataformas en la educación a distancia. 2005. Vol. 37, 1, pág. 8.

3. Leyva Velázquez, Deivis. Estrategia de portabilidad para desplegar aplicaciones Web desarrolladas con la tecnologia .NET de Microsoft en plataforma Linux, utilizando MONO. 2008.

4. Diccionario de la Lengua Española. [En línea] [Citado el: 11 9, 2014.] http://www.wordreference.com/definicion/colaboracion.

5. TheFreeDictionary. [En línea] [Citado el: 11 18, 2014.] http://es.thefreedictionary.com/colaboracion.

6. Fernández-Pampillón Cesteros, Ana. Las plataformas e-learning para la enseñanza y el aprendizaje universitario en Internet. 2009.

7. García Sánchez, Maikel y Borges Martínez, Yenisel. Sistema automatizado para la gestion de las evaluaciones de desempeno de los profesores de la facultad 10 de la Universidad de las Ciencias Informaticas. 2010.

8. Definición .DE. [En línea] [Citado el: 11 27, 2014.] http://definicion.de/gestion.

9. Díaz Berenguer, Abel y Román Vieito, Alberto Ramón. Sistema de Administracion y Configuracion de la solucion de Captura y Catalogacion de Medias. 2009.

10. Durán Pérez, Yadeilis y Torres Sales, Yanet. Desarrollo de un sistema de administracion web para la linea de desarrollo seguridad del centro de desarrollo de informatica industrial. 2010.

11. SciELO. [En línea] [Citado el: 12 8, 2014.] http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352006000500009.

12. Diccionario de la Lengua Española. [En línea] [Citado el: 12 14, 2014.] http://www.wordreference.com/definicion/catalogación.

13. Repositorio Documental de la Universidad de Salamanca. [En línea] [Citado el: 02 15, 2015.] http://gredos.usal.es/jspui/handle/10366/119930.

14. Svenonius , Elaine. *The intellectual foundation of information organization.* 2000.

15. Gómez , Rosa y Sanabria, Daniel Jorge. *Manual de Catalogación.* 2005.

16. El Rincón del Vago. [En línea] [Citado el: 03 05, 2015.] http://html.rincondelvago.com/tendencias-estadisticas.html.

Bibliografía

17. Lagos, Hugo . Gestión Unidad Técnica Pedagógica, Colegio Salesiano Valparaíso. [En línea] [Citado el: 12 18, 2014.] http://hugolagospedagogia.blogspot.com/2008/07/proyecto-curricular.html.

18. Jacobson, Ivar, Booch, Grady y Rumbauch, James . *El proceso unificado de desarrollo de software.* 2000. ISBN 84-7829-036-2.

19. Kruchten, Philippe. *The rational unified process: an introduction.* 2004.

20. Martínez, Alejandro y Martínez, Raúl. *Guía a Rational Unified Process.* 2002.

21. Jeffries, Ron, Anderson, Ann y Hendrickson, Chet. *Extreme programming installed.* 2001.

22. Letelier, Patricio. *Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP).* 2006.

23. El Lenguaje de Modelado Unificado. [En línea] [Citado el: 01 23, 2015.] http://www.docirs.cl/uml.htm.

24. Guía Breve de CSS. [En línea] [Citado el: 01 29, 2015.] http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo.

25. Gauchat, Juan Diego. *El gran libro de HTML5, CSS3 y JavaScript. Barcelona.* 1. s.l. : Ediciones Técnica Marcombo, 2012. ISBN: 978-84-267-1770-2.

26. Ramírez Batista, Leandro Raúl y Talancón Díaz, Reinier. Sistema para la visualización de imágenes y modelos 3D médicos en la WEB con fines educativos. 2010.

27. Valbuena, Aponte y María, Angela. *Guía comparativa de Frameworks para los lenguajes HTML 5, CSS y JavaScript para el desarrollo de aplicaciones Web.* 2014.

28. Flanagan, David. *JavaScript. La Guía Definitiva.* s.l. : Anaya Multimedia, 2007. ISBN: 978-84-415-2202-2 84-415-2202-2.

29. Pressman, Roger S. *Ingeniería de Software.* 6. s.l. : Mc Graw Hill, 2005. ISBN: 0-07-285318-2.

30. Bauta González, Marcel. *Buscador de Activos Reutilizables de Software.* 2012.

31. Serie Científica de la Universidad de las Ciencias Informáticas. [En línea] 2012. [Citado el: 03 07, 2015.] https://publicaciones.uci.cu/index.php/SC/article/view/1032/581.

32. Visual Paradigm. *Visual Paradigm for UML.* [En línea] [Citado el: 03 20, 2015.] http://www.visual-paradigm.com/product/vpuml/.

33. Clavero Vázquez, Elisa y Tur Sivila, Dianelis. Diseño de un catálogo de servicios Web basado en las especificaciones y las normas de UDDI para la plataforma de informatización en la UCI. 2007.

34. Página Oficial Rational. [En línea] [Citado el: 03 25, 2015.] http://www.rational.com.ar/herramientas/roseenterprise.html.

35. Gilfillan, Ian. *MySQL.* [ed.] Anaya Multimedia. 2003. ISBN: 978-84-415-1558-1 84-415-1558-1.

36. Sitio oficial de PostgresSQL. [En línea] [Citado el: 03 27, 2015.] http://www.postgresql.org.es/sobre_postgresql.

37. Ventajas de PostgreSQL. [En línea] [Citado el: 03 30, 2015.] http://soporte.tiendalinux.com/portal/Portfolio/postgresql_ventajas_html.

38. Glosario Digital. [En línea] [Citado el: 01 16, 2015.] http://www.glosariodigital.com/termino/framework.

39. Potencier, Fabien . Symfony 1.2, la guía definitiva. [En línea] [Citado el: 04 02, 2015.] http://www.librosweb.es/symfony_1_2.

40. Sánchez Rosas, Juan Eladio. Comparativa de frameworks en PHP: CakePHP, Symfony y Zend Framework. [En línea] 09 02, 2007. [Citado el: 04 06, 2015.] http://tuxpuc.pucp.edu.pe/articulo/comparativa-de-frameworks-en-php-cakephp-symfony-y-zend-framework.

41. NetBeans IDE. [En línea] [Citado el: 04 07, 2015.] https://netbeans.org/features..

42. Álvarez, Miguel Angel. Zend Studio. [En línea] [Citado el: 04 10, 2015.] http://www.desarrolloweb.com/articulos/1178.php.

43. González, Aleksander. *Método de evaluación de arquitecturas de software basadas en componentes (MECABIC).* 2005.

44. Campo, Gustavo Damián. *Patrones de Diseño, Refactorización y Antipatrones. Ventajas y Desventajas de su Utilización en el Software Orientado a Objetos.* 2009.

45. Larman, Craig. *UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado.* [ed.] Prentice Hall. 2. 2006. pág. 590 . ISBN 8420534382, 9788420534381.

46. Diagrama de Despliegue. [En línea] [Citado el: 05 07, 2015.] http://www.diadspg.blogspot.com.

47. Villalón, Antonio. *Seguridad en Unix y Redes.* 2002.

48. Manual de PHP. [En línea] [Citado el: 04 23, 2015.] http://www.php.net.

49. Marcos González, Rubén. Recursos para la programación en php. [En línea] [Citado el: 05 02, 2015.] http://tgp0607.awardspace.com/Recursos_PHP.pdf.