UNIVERSIDAD DE QUINTANA ROO

División de Ciencias e Ingeniería

Implementación de un sistema informático de almacenamiento e intercambio de archivos

Trabajo de Tesis

para obtener el grado de

Ingeniero en Redes

PRESENTA

José de Jesús Chan Coh

Directora de Tesis

MTI. Melissa Blanqueto Estrada

Asesores

MTI. Vladimir Veniamín Cabañas Victoria

Lic. Sergio Alberto Solís Sosa

Chetumal, Quintana Roo, México. Febrero de 2013

UNIVERSIDAD DE QUINTANA ROO

División de Ciencias e Ingeniería

Trabajo de Tésis elaborado bajo supervisión del Comité de Asesoría y aprobada como requisito parcial para obtener el grado de:

INGENIERO EN REDES

Comité de Trabajo de Tesis

Directora:

MTI. Melissa Blanqueto Estrada

Asesor:

M.T.I. Vladimir Veniamin Cabañas Victoria

Asesor:

Lic. Sergio Alberto Solís Sosa

Chetumal, Quintana Roo, México, Febrero de 2013.

Dedicatoria

A Dios, por darme vida y salud.

A mis padres, Guadalupe y Felix, que fueron mi inspiración en el transcurso de la carrera, que son mi fe y fortaleza para seguir siempre hacia adelante.

A mis hermanos, Lucero, Cindy y Jimmy, por su cariño y amor en los momentos dificiles.

A mis tios, Hugo y Benita, quienes siempre estuvieron ahí, para brindarme su apoyo.

A mis tios, Rufino y Fanny, que siempre me brindaron su apoyo, sin pregunta alguna.

A mi gran familia, por su compresión, apoyo, respeto y admiración.

A cada uno de ellos, muchas gracias.

Agradecimientos

Quiero agradecer con sincero respeto a todos mis maestros, quienes me inculcaron grandes valores y conocimientos.

Con gran admiración y respeto, agradezco al Lic. Sergio Solís, por permitirme aprender de él, y que además de ser un gran maestro, me dio la oportunidad de ser un gran amigo.

En especial agradezco a mi gran amiga y hermana Rosalina Ek Dzib, que juntos llegamos el final, nunca me dejó caer y me brindó el más valioso de los sentimientos, su amistad.

Este trabajo fue financiado en la convocatoria 2012 “Apoyo a la titulación”, de la División de Ciencias e Ingeniería.

A todos ellos, gracias.

Resumen

Los repositorios son herramientas que se caracterizan por almacenar información importante y responsable dentro cualquier institución o empresa. Un repositorio institucional contiene mecanismos para importar, identificar, almacenar, preservar, recuperar y exportar un conjunto de objetos digitales, normalmente desde un portal web.

El objetivo del proyecto fue desarrollar un prototipo propio, el sistema de almacenamiento e intercambio de archivos “BackPack” gestiona toda información producida por los usuarios activos dentro de la intranet de la Universidad de Quintana Roo, esto con la intención de agilizar la comunicación.

Este sistema optimiza los métodos y características esenciales de la primera versión del sistema, el desarrollo de la interfaz mejoró la experiencia de los usuarios.

Se siguió una metodología que facilitó la construcción de la aplicación y permitió analizar la efectividad de los métodos y tecnologías propuestas.

Como consecuencia de los resultados de este proyecto, se tiene previsto una segunda fase en la que se extiendan las capacidades del sistema desarrollado, y en la que el prototipo se convierta en una plataforma que resuelva los problemas de interoperabilidad con los navegadores web.

Asimismo, es necesario dar continuidad a la declaración de políticas y lineamientos institucionales, que determinen la forma de operación del sistema de información entre la comunidad universitaria.

Contenido

Índice de ilustraciones

Índice de tablas

CAPÍTULO I

INTRODUCCIÓN

1.1. Antecedentes

Internet, desarrollado a partir de los años 70, creció paulatinamente hasta principios de los años 90, al abrir el acceso se provocó un salto exponencial en el número de usuarios, el tipo y la cantidad de información que se intercambiaba. El siguiente gran salto, mucho más importante, ocurrió unos años después cuando se introdujo la World Wide Web. Los primeros usuarios de Internet; investigadores de universidades y de otras instituciones no tardaron en experimentar con el nuevo entorno para agilizar el intercambio de ideas y datos, utilizando para ello las nuevas herramientas como correo electrónico y los protocolos FTP (para la transferencia de archivos) y telnet (para la conexión entre PC's remotos).

Con el desarrollo de la tecnología en el procesamiento de datos surge también la necesidad del almacenamiento de grandes volúmenes de información. Los usuarios demandan cada vez más espacio en almacenamiento para compartir, difundir, resguardar y distribuir su información.

En la Universidad de Quintana Roo existe un sistema de almacenamiento e intercambio de archivos que se implementó en una primera versión, el cual permite que la comunidad universitaria almacene e intercambie información a través de su perfil de correo electrónico, sin embargo, las necesidades de los usuarios han evolucionado y se ha hecho necesaria una reingeniería del sistema para satisfacer los requerimientos del usuario; este trabajo de tesis pretende desarrollar una nueva versión de este sistema, aplicando una metodología para desarrollo de software adecuada..

1.2. Descripción del problema

Entre la comunidad universitaria es frecuente que se manejen grandes volúmenes de información, los cuales son almacenados en diferentes tipos de dispositivos o bien en forma impresa lo cual representa un fuerte gasto económico además ésta última opción no es amigable con el medio ambiente.

En la Universidad de Quintana Roo el correo electrónico es uno de los medios más usados para el intercambio de información, hace posible que sea enviada a través de la Internet de manera confiable, rápida y fácil. Sin embargo el tamaño de la información puede llegar a ser un problema, puesto que este servicio restringe el volumen de información a enviar.

Además de disponibilidad y recuperación de la información digital son necesarias las herramientas que permitan la recuperación de datos, información y recursos de distintas fuentes, combinarlos, integrarlos, analizarlos, así como publicar y difundir nuevos materiales haciendo uso de recursos ya disponibles en Internet.

Con lo anterior podemos plantear la siguiente interrogativa: ¿La implementación de un sistema informático de almacenamiento e intercambio de archivos podrá beneficiar a la comunidad universitaria de la Universidad de Quintana Roo?

1.3. Objetivos

Objetivo general:

Implementar un sistema informático para el almacenamiento y transferencia de archivos que beneficie a la comunidad universitaria.

Objetivos específicos:

- Identificar los requerimientos de los usuarios para almacenar y compartir archivos.
- Diseñar el modelo de información requerido para llevar a cabo el almacenamiento e intercambio de archivos.
- Implementar y probar el sistema para analizar su rendimiento.

1.4. Justificación

En la Universidad de Quintana Roo (UQROO), al igual que en muchas otras universidades del mundo, la información y los contenidos digitales representan un recurso importante que abre nuevas posibilidades para la investigación, la docencia y la administración. Para la comunidad académica, que produce y consulta grandes cantidades de información, los recursos digitales en Internet tienen un papel fundamental, por la facilidad de encontrar información relacionada con las actividades de docencia, investigación y gestión académica.

La comunidad universitaria almacena información en usb flash, discos compactos y otros dispositivos de almacenamiento, los cuales al momento del intercambio de la información se recurre al préstamo del mismo, lo que implica el riesgo de que se pierda o se infecte por algún virus en la PC donde se conecte. Este método se vuelve inadecuado si la persona requiere la información rápidamente y se encuentra en otro lugar geográfico. Es posible usar el sistema de correo electrónico, sin embargo la principal limitación es el tamaño del archivo que se desea transferir.

La propuesta de este proyecto es implementar un sistema informático de almacenamiento y transferencia de archivos que sea operado en la intranet de la UQROO, lo cual pretende agilizar el proceso de recepción y envió de información.

Por otra parte el Centro de Tecnologías de la Información y la Comunicación (CTIC), perteneciente a la UQROO, cuenta con los requerimientos de una intranet de alta velocidad, hardware y software indispensables para que el sistema informático en cuestión funcione adecuadamente para el número de estudiantes y administrativos que necesiten recurrir a este medio. Con esto, podemos determinar que se dispone de la infraestructura tecnológica necesaria en capacidad de almacenaje, velocidad de transferencia y seguridad para el manejo de grandes volúmenes de información.

El usuario estará familiarizado con un sistema hecho a la medida de sus necesidades y que le permitirá establecer un enlace seguro y confiable.

1.5. Alcance y restricciones

Alcance

Desarrollar una versión inicial de software que cubra las necesidades actuales de la comunidad universitaria.

Restricción

Las principales restricciones que afectarían al desarrollo de este proyecto se resumen en:

- Las múltiples conexiones que se incrementen con la popularización del servicio y el crecimiento de usuarios podrían ocasionar que la capacidad del servidor no sea adecuada para cubrir la demanda.
- La cantidad de información ingresada al sistema por los usuarios podría exceder la capacidad del servidor.

CAPÍTULO II

MARCO TEÓRICO

2.1. Los repositorios

Las necesidades en la educación e investigación demandan un acceso más eficiente a los recursos, a través de intercambio de información y sistemas compatibles sean interoperables. Estas necesidades están presentes en diversas organizaciones a nivel mundial y distintos esfuerzos y estándares se están desarrollando para coadyuvar a una mejor gestión de los recursos digitales.

Actualmente los repositorios son una herramienta para administrar y difundir los recursos electrónicos producidos por miembros de diversas comunidades, incrementando y fortaleciendo el acceso a los recursos académicos a nivel institucional y mundial. Sin embargo, no existe un consenso general sobre las características exactas que deben tener estos depósitos de contenidos para ser considerados como un repositorio, ello debido en gran parte a que existen divergencias en torno a los objetivos que se quieren lograr con la creación y mantenimiento de un repositorio en una institución académica.

Lo anterior conlleva a discusiones en torno a quiénes son responsables de construir y mantener los repositorios, el tipo de material que se puede depositar, la asignación de derechos patrimoniales, la cobertura del costo de instalación y mantenimiento, así como los mecanismos para asegurar la calidad, integridad y preservación del material.

Un repositorio no es sólo una colección de objetos digitales y es común confundirlos con bibliotecas digitales, la línea es muy fina entre ambos, por ello es importante dejar claro qué es un repositorio.

De acuerdo a Heery et. al (Heery & Anderson, 2005), existen cuatro principales características de un repositorio: la primera es que debe contar con mecanismos que permitan el depósito de material por parte del creador, el dueño u otra persona (por ejemplo, un bibliotecario); la arquitectura del repositorio debe manejar tanto el contenido como sus metadatos; deben existir servicios básicos como búsqueda y recuperación, administración, controles de acceso y permisos, entre otros.

Por último, el repositorio debe de ser sustentable a largo plazo, administrado y por lo tanto, un repositorio no se refiere únicamente a los contenidos ya que la forma de administración, sus funciones y servicios forman parte de sus características.

Tipos de repositorios

De acuerdo a una investigación realizada por la Universidad Nacional Autónoma de México se identifican diferentes tipos de repositorios (López Guzmán, y otros, 2006), algunos dependen del tipo de validación de sus contenidos, algunos otros por la forma en que se administran. A continuación se describen algunos de éstos.

Los repositorios de eprints y temáticos

Los primeros repositorios se crearon para que los investigadores pudieran colocar sus artículos electrónicos en línea y para que otros investigadores del mismo campo pudieran acceder a ellos a través de las interfaces de búsqueda y recuperación. El motivo principal era poder compartir sus resultados de investigación de una forma más rápida, mejorando así la comunicación entre los investigadores. Adicionalmente, de esta forma se eliminaban las barreras económicas de acceso que existen para consultar los artículos en revistas académicas comerciales. (Hanard, 2001).

Estos repositorios suelen conocerse como temáticos ya que son creados y mantenidos por una institución, pero cualquier persona puede depositar material de un tema en particular. Generalmente, son repositorios que albergan artículos científicos (eprints), ya sea la versión previa a la publicación (preprint), la versión publicada o una versión posterior (post-print).

Un ejemplo es el repositorio de arXiv, mantenido por Los Alamos, el cual contiene eprints de las áreas de física, matemáticas, ciencias de la computación y biología cuantitativa.

El éxito de estos repositorios se debe a que fue uno de los principales motores para el movimiento de acceso abierto u Open Access (OA). El movimiento propone que los resultados de la investigación científica, en forma de artículos arbitrados, estén disponibles digitalmente en línea y de forma gratuita, así como libre de prácticamente toda restricción de copiado (copyright). Los repositorios de este tipo tienden a solamente aceptar artículos arbitrados y de ciertos temas. OA intenta eliminar barreras económicas como pagos de suscripción a revistas, así como barreras de permiso, principalmente de reproducción y distribución, a la literatura científica mundial. (Hagemann, 2001)

Los repositorios de materiales académicos

Este tipo de repositorios son una herramienta útil para ofrecer una amplia gama de materiales académicos y no sólo de artículos arbitrados.; se parte del hecho de que muchos repositorios, pero no necesariamente todos, apoyan el movimiento de Open Access, ofreciendo artículos arbitrados en su colección (Heery & Anderson, 2005) pero también contenidos de valor académico que no pasan por este proceso.

Este modelo aumenta las posibilidades de acceso a los recursos electrónicos, permite generar nuevas formas de publicación y arbitraje, facilita compartir y reutilizar datos crudos de investigación u objetos de aprendizaje, entre otras funciones. Uno de los principales objetivos es apoyar y mejorar la enseñanza, aprendizaje e investigación en la institución.

Los repositorios institucionales

Este tipo de repositorios, que incluyen material académico diverso, tienden a ser organizados por una institución más que por áreas temáticas. El objetivo principal es que el repositorio funciona como un tipo de vitrina para mostrar la producción académica de la institución que lo maneja. Se conocen como repositorios

institucionales (RI) (institutional repositories). Los RI están basados y apoyados en una institución académica y ofrecen material digital producido por sus miembros. Tanto el contenido del repositorio como las políticas de selección y almacenamiento de los materiales está definido institucionalmente (Jhonson, 2002). Aunque las características adicionales pueden variar, tienen por lo menos las siguientes características en común:

- Son accesibles en línea y contienen material académico que está definido y producido por una institución;
- El repositorio es interoperable y abierto al utilizar un software para el intercambio de metadatos con otros repositorios, que colecciona, almacena y difunde el material.

En este sentido los repositorios institucionales forman parte del proceso de comunicación académica y tienen el compromiso de ser acumulativos y perpetuos.

Para el año 2011 se tenían registrados 1636 repositorios institucionales en todo el mundo (ROAR, 2011). Actualmente, hay una tendencia mundial en incorporar repositorios a las instituciones académicas, algunas ya cuentan con él y otras están en vías de hacerlo.

2.2. Open Access

El desarrollo del campo de los repositorios ha puesto en marcha diversas iniciativas que buscan su estandarización tecnológica y operativa. También hay iniciativas o movimientos que están revolucionando los esquemas tradicionales de la publicación y edición de publicaciones científicas.

El fruto del trabajo intelectual de investigadores en todo el mundo ha estado "monopolizado" por las casas editoras de revistas de divulgación científica debido a que son la herramienta para poder dar a conocer los resultados de su trabajo. Como consecuencia la difusión del conocimiento se ha visto limitado tanto por los

altos costos que involucra la publicación tradicional, como por la monopolización antes mencionada.

Open Access (OA) surge como un movimiento que cuestiona el monopolio que las grandes editoriales ejercen sobre la distribución de la información científica y propone dos soluciones para que la literatura erudita sea más accesible:

- Que los autores publiquen en revistas OA ó
- Que los autores además de publicar en revistas científicas, depositen una copia de su artículo en un repositorio temático o institucional para que otras personas puedan consultarlo.

El movimiento OA consiste, de acuerdo con la Declaración de Budapest (Hagemann, 2001), en que cualquier usuario pueda leer, descargar, copiar, distribuir, imprimir, con la posibilidad de buscar o enlazar todos los textos de artículos de literatura erudita, recorrerlos para indexación exhaustiva, usarlos como datos para software, o utilizarlos para cualquier otro propósito legal, sin barreras financieras, legales o técnicas, distintas de la fundamental de tener acceso a la propia Internet. La única limitante a la reproducción y distribución de los artículos publicados, y la única función del copyright en este dominio, no puede ser otra que dar a los autores control sobre la integridad de su trabajo y el derecho a ser apropiadamente acreditados y citados.

La Declaración de Berlín (Invenia, 2002) (Acceso Abierto al Conocimiento en Ciencias y Humanidades) promueve Internet como el instrumento funcional que sirva de base global del conocimiento científico y la reflexión humana, bajo OA.

Siguiendo los pasos de las iniciativas de Budapest y Berlín, está la carta de ECHO (European Cultural Heritage Online: Open Access Infrastructure for a Future Web of Culture and Science, 2011) que tiene como objetivo la definición de criterios para la explotación adecuada de las potencialidades de los nuevos medios para la preservación archivística, la exploración académica y la distribución pública del patrimonio cultural de la humanidad.

OA está abriendo un camino para aquellos científicos que desean difundir sus conocimientos e investigaciones, sin ánimo de lucro y en algunos casos de forma directa, sin la intervención de editoriales. Las instituciones tienen en OA una opción para concentrar y poseer la producción de sus investigaciones, para difundirse y aprovecharse por más comunidades.

2.3. Proyectos basados en repositorios

Numerosas organizaciones a nivel mundial se han dado a la tarea de impulsar los repositorios con la producción de la investigación que generan sus universidades e investigadores, sumándose a diferentes iniciativas de trabajo colaborativo que han surgido alrededor del mundo y que plantean soluciones aplicables a situaciones específicas.

Entre los proyectos más difundidos están:

1. **Dawning of the Dutch Network of DAREN (Digital Academic REpositories)** (Van del Kuil & Feijen, 2004). DARE es una iniciativa común de universidades holandesas cuyo objetivo es hacer que su producción académica sea digitalmente accesible. También colaboran la Biblioteca Nacional de los Países Bajos, la KNAW (Academia de Artes y Ciencias de los Países Bajos) y la NWO (Organización de los Países Bajos para la Investigación Científica).
2. **Red de Bibliotecas Virtuales de Ciencias Sociales de América Latina y el Caribe CLACSO** (Consejo Latinoamericano de Ciencias Sociales, 2010). La red CLACSO, a la que se puede tener acceso por Internet y sin ningún costo, tiene como objetivo principal el promover y facilitar el acceso a los resultados de las investigaciones de los centros que son miembros.

   Entre los servicios que proporciona esta Red de Bibliotecas sobresalen la Sala de Lectura con textos completos de libros, artículos, ponencias y documentos de trabajo publicados por la red CLACSO y otras instituciones;

Bases de datos sobre la producción académica de los centros miembros: registros bibliográficos de las publicaciones, investigaciones con descripción de cada investigación, investigadores con dirección de correo electrónico de contacto; enlaces a bibliotecas y bases de datos de ciencias sociales.

3. **Harvesting Institutional Resources in Scotland Testbed (HaIRST)** (Dunsire, 2005). Es uno de 14 proyectos iniciados por JISC (Joint Information System Committee) dentro del programa Focus on Access to Institutional Resources (FAIR). Este proyecto comenzó en agosto de 2002 por un consorcio de tres universidades escocesas: Strathclyde vía el Centro para la Investigación de la Biblioteca Digital (CDLR), Napier y St.Andrews. HaIRST enfoca su investigación en el diseño, la puesta en práctica y el despliegue de un servicio experimental para un acceso autónomo, a nivel todo Reino Unido, de recursos institucionales creados en Escocia. El principal objetivo del proyecto es investigar y aconsejar sobre algunos de los requisitos o requerimientos técnicos, culturales, y de organización asociados al depósito, al acceso, y al descubrimiento de recursos institucionales en el ambiente de la información.
4. **eScholarshipRepository** (California Digital Library, 2000). Algunas de las principales actividades de la Universidad de California es asegurar la creación, diseminación y preservación de los productos de la investigación y la enseñanza; en base a su experiencia ha llegado a la conclusión de que la comunicación es la base para todas las actividades académicas al interior de una organización educativa y se ha dado cuenta, al igual que otras organizaciones que se han sumado a la iniciativa OA, que la publicación de materiales académicos se ha vuelto costosa y de acceso restringido. Por ello, por medio de la biblioteca Digital de California patrocina la Iniciativa eScholarship. Los servicios que ofrece esta iniciativa son en respuesta a la necesidad de encontrar mecanismos alternativos de publicación proporcionando acceso persistente y que el contenido pueda encontrarse con facilidad.

5. **BioMed Central** (Cental, 2011). Es una casa editorial independiente comprometida a proporcionar acceso abierto inmediato a investigación biomédica. En esta iniciativa, el temor de que la calidad de los artículos publicados mediante el modelo OA desaparece ya que BioMed Central está comprometida a mantener altos estándares por medio de una completa y rigurosa revisión de los documentos que publica, adicionalmente ofrece una amplia variedad de publicaciones y otros servicios siempre conservando la mirada en una misma dirección: que todos los artículos publicados por esta casa editorial sean de libre acceso y puedan ser re-utilizados y re-distribuidos. Principios básicos que rigen el movimiento OA.

2.4. Tecnologías para la creación y operación de RI

Varias son las tecnologías que se interrelacionan para la implementación y la creación de RI, entre las principales destacan tres componentes relacionados; el primero aborda la utilización de los metadatos de acuerdo a su estructuración e intercambio, el segundo se refiere al protocolo OAI-PMH utilizado para su transmisión e interoperabilidad, y el tercero a la revisión del software comúnmente utilizado en los sistemas generales de repositorios. Antes de puntualizar en los tres aspectos, a continuación se mencionan las características que esas tecnologías deben cumplir en un repositorio:

- Que tengan soporte comunitario
- Fácilmente integrables
- Que proporcionen mecanismos de autentificación y autorización
- Que proporcionen mecanismos de verificación y seguridad de los contenidos
- Que tengan un sistema de administración central
- Que establezcan claramente las licencias de contenido y sus restricciones

- Servicios web habilitados
- Que utilice uno o varios esquemas flexibles de metadatos (descriptivos, técnicos, de preservación, derechos) para la captura y encapsulado de la información de los archivos
- Que soporten la federación y escalabilidad de los repositorios
- Mecanismos de captura y egreso
- Mecanismos y políticas de preservación digital, respaldo y recuperación de datos
- Que maneje distintos formatos de archivos (texto, imágenes, conjuntos de datos, video, audio, simulaciones, etc.)

De forma amplia, la plataforma tecnológica de repositorios institucionales consiste de los siguientes elementos:

- Servidores Windows o Unix/Linux
- Un servidor web, como Apache y sus herramientas de aplicación web
- Un manejador de bases de datos como MySQL, DB2, Oracle, Post gres, SQL Server
- Dirección URL permanente y persistente

Esquema de Metadatos Dublin Core

Los metadatos se refieren a la información acerca de un recurso digital y facilitan la descripción y recuperación del mismo en los sistemas de información, su manejo en los repositorios es la clava para abrir el contenido digital de las instituciones y hacerlos accesibles, para estos la descripción de un objeto (Quién lo creó, cómo se llama, de qué trata, cuándo fue publicado, etc.) es una de las

dimensiones más importantes para que los contenidos puedan ser registrados y expuestos como observamos en la Tabla 1.

Tabla 1.- Información de metadatos Dublin Core.

Contenido	Propiedad intelectual	Creación e identidad
Título	Creador	Fecha
Tema	Editor	Tipo
Descripción	Colaborador	Formato
Fuente	Derechos	Identificador
Lengua		
Relación		
Cobertura		

El protocolo OAI-PMH

El protocolo para la transmisión de contenidos en Internet denominado OAIPMH (Open Archives Initiative – Protocol for Metadata Harvesting) [http://www.openarchives.org] tuvo sus primeros trabajos de desarrollo desde 1999. Originalmente pensado para mejorar el acceso a los repositorios de eprints, depósitos de documentos de investigación científica a texto completo, fue paulatinamente mejorado y adaptado hasta convertirse en el estándar utilizado para facilitar la disponibilidad de cualquier tipo de documento digital procedente de distintos repositorios, por medio de la transmisión de metadatos a través de la World Wide Web, utilizando estándares abiertos HTTP (Hypertext Transport Protocol) y XML (eXtensible Markup Language).

La arquitectura de OAI-PMH se basa en clientes y servidores. Los primeros son los proveedores que proporcionan la información (metadatos) para ser cosechados por los segundos, proveedores de servicios (búsqueda y

recuperación) o recolectores; crean y ofrecen herramientas de acceso a los metadatos cosechados, tales como interfases de búsqueda, estadísticas de tipos, accesos, cantidad, etc. Los repositorios institucionales generalmente tienen ambos papeles ofreciendo sus metadatos para cosechar así como servicios de consulta para el usuario final.

OAI-PMH utiliza transacciones HTTP para emitir preguntas y obtener respuestas entre un servidor o archivo y un cliente o servicio recolector de metadatos. El segundo puede pedir al primero que le envíe metadatos según determinados criterios. En respuesta el primero devuelve un conjunto de registros en formato XML, incluyendo identificadores (URL por ejemplo) de los objetos descritos en cada registro.

Las peticiones se emiten utilizando los métodos GET o POST del protocolo HTTP y constan de una lista de opciones con la forma de pares del tipo: clave=valor. Existen seis peticiones que un cliente puede realizar a un servidor: Para los proveedores de datos es importante tomar en consideración las siguientes definiciones y descripciones:

• **Registros (*Records*)-** Un registro son los metadatos de un recurso digital en un formato específico. Los registros están compuestos por tres partes: cabeza con metadatos (obligatorios) y una declaración descriptiva (opcional).

• **Estampas de fecha (*Datestamps*)-** Es la fecha de la última modificación de un registro y es obligatoria. La estampa permite ofrecer información acerca de los metadatos para búsquedas más selectivas utilizando desde (*from*) y hasta (*until*).

• **Esquemas de metadatos (*Metadata schema*)-** El OAI-PMH soporta la diseminación de múltiples formatos de metadatos de un repositorio, aunque todos deben de soportar como mínimo Dublin Core.

• **Conjuntos (*Sets*)-** Los conjuntos permiten la partición lógica de los repositorios y permite realizar búsquedas más selectivas (*set parameter*). No son obligatorias, no existen recomendaciones para su implementación y no son exhaustivas. Es

importante y necesario que cada comunidad tenga acuerdos con respecto a la definición de sus conjuntos.

• **Formato de petición (*Request format*)-** Los repositorios deben soportar los métodos HTTP de GET y POST para peticiones. Existen seis tipos de peticiones: Identify, LIstMetadataFormats, ListSets, ListIdentifiers, ListRecords, GetRecord. El cosechar de datos no necesariamente utiliza todas las peticiones pero un repositorio debe implementarlos todos.

• **Respuesta (*Response*)-** Las respuestas son tipo HTTP y el tipo de contenido debe de ser texto/xml.

Herramientas de software

Para hacer más flexible la creación administración de repositorios se han desarrollado varios tipos de software que tienen la intención de mejorar el ambiente para el usuario y mejorar la disponibilidad al momento de seguir las especificaciones para crear repositorios, entre los más conocidos se encuentran los siguientes:

2. **Invenio** (INVENIO, 2011). Es una suite de software libre que le permite ejecutar su propia biblioteca digital o repositorio de documentos en la web. La tecnología que ofrece el software cubre todos los aspectos de la gestión de la biblioteca digital de la ingestión de documentos a través de la clasificación, indexación, y la preservación de su difusión. Invenio cumple con las normas tales como la Iniciativa de Archivos Abiertos recolección protocolo de metadatos (OAI-PMH) y utiliza MARC 21 (MARC 21 Format for Bibliographic Data, 2001) como su formato bibliográfico subyacente. La flexibilidad y el rendimiento de Invenio convierten en una solución integral para la gestión de repositorios de documentos de moderado a gran tamaño (varios millones de registros).

   Invenio ha sido desarrollado originalmente en el CERN para ejecutar el

servidor de documentos del CERN, que es el laboratorio de física de partículas más grande del mundo y cuya producción intelectual arroja la gestión de más de 1.000.000 de registros bibliográficos en física de altas energías desde 2002, cubriendo artículos, libros, revistas, fotos, videos y más. Invenio está siendo co-desarrollado por una colaboración internacional que incluye institutos como el CERN, DESY, EPFL, FNAL, SLAC y está siendo utilizado por cerca de treinta instituciones científicas de todo el mundo

3. **Greenstone** (Greentone, 2009). Es una aplicación que permite la creación y utilización de una biblioteca digital con sus respectivas colecciones. Fue desarrollado por la Universidad de Waikato de Nueva Zelanda, y se distribuye bajo una licencia GNU. Sus requerimientos son los siguientes:

   • Se ejecuta en plataformas Windows, Linux y Macintosh OS X, así como con un servidor Apache.

   • Utiliza GDBM como manejador de bases de datos, además de Java Runtime Environmet para su interfaz gráfica.

   • Es compatible con varios estándares de metadatos, entre ellos MARC

4. **DSpace** (Dspace, 2011). Es un software desarrollado por el MIT (Massachussets Institute of Technology) con apoyo de Hewlet Packard (HP). Tiene muy en cuenta la visón de "comunidades"; es decir, una institución dedicada a la investigación está compuesta por departamentos, centros de investigación, participación de estudiantes y otras unidades y como cada una de estas unidades tiene requerimientos específicos, diferentes unos de otros, DSpace hace posible un flujo de trabajo y manejo de políticas que permite distribuir los contenidos, y manejar la propiedad intelectual de cada uno de ellos. También está orientado al problema de preservación a largo plazo de los materiales de investigación. Una característica importante de esta opción es el uso de PostgreSQL en lugar de MySQL como sistema manejador de bases de datos que es mucho más robusto y flexible además de que tiene un rendimiento excelente.

5. **Fedora-commons** (FEDORA, 2009). Es un sistema flexible para la gestión de repositorios digitales, capaz de administrar de forma eficiente hasta un millón de objetos. Cuenta con tres interfaces: una para administrar el repositorio, otra para detectar y distribuir los objetos y una última para dar acceso a la información en Web. Fue desarrollado por la Universidad de Cornell en el año 2001 y actualmente es utilizado en proyecto de repositorios de 3R de la UNAM (UNAM, 2009). Sus características son las siguientes:

   • Trabaja sobre cualquier plataforma Unix, así como con cualquier servidor Web que cuente con TomCat, McKoiSQL y Java SE Development Kit.

   • Utiliza XML para decodificar los metadatos, mismos que pueden estar asentados bajo cualquier esquema.

   • Interactúa con cualquier otro sistema que utilice

6. **Eprints** (Epints, 2011)**.** Sus dependencias de software extra son mucho menores en comparación con CDS Invenio. De acuerdo con los propios desarrolladores, es un sistema en extremo fácil de poner en marcha y con un mínimo de experiencia técnica. Está pensado para ser muy flexible y adaptable a necesidades muy particulares. Es un producto en constante desarrollo que comienza a incorporar funciones como búsquedas avanzadas, metadatos extendidos, etc. Una desventaja de su desarrollo continuo es que se está pensando cobrar por las características avanzadas aunque el núcleo del software y sus características funcionales básicas sigan siendo de libre distribución.

2.5. Repositorio Institucional UQROO

Las tecnologías especificas en un RI pueden ser varias e implementarse en distinta forma, este proyecto de tesis propone el estudio de tecnologías y de prácticas para el desarrollo de los repositorios institucionales digitales, sin embargo debido a las necesidades de los usuarios y al planteamiento generado con anterioridad, se propone la creación del prototipo de un repositorio

institucional universitario de recursos digitales, para almacenar y compartir los recursos digitales producidos por la comunidad universitaria, para hacer más eficiente su producción y utilización, así como para facilitar su visibilidad real tanto interna como externamente.

A nivel mundial diversas universidades e iniciativas están dando marcha a proyectos similares, bajo el concepto de repositorios institucionales, que facilitan la gestión de documentos producidos y utilizados por comunidades universitarias de investigación y docencia.

Para contribuir a un uso más eficiente de las tecnologías de información dentro de la UQROO, con la innovación de tecnologías y el desarrollo de sistemas que favorezcan el uso y difusión de los recursos que se producen, este documento presenta la propuesta para llevar a cabo un trabajo de investigación aplicada, con la finalidad de obtener un modelo para la creación de un repositorio de almacenamiento e intercambio de archivos digitales.

CAPÍTULO III

MARCO CONTEXTUAL

3.1. Marco contextual

El 31 de mayo de 1991 en la ciudad de Chetumal Quintana Roo, se decreta la creación de la Universidad de Quintana Roo, como un organismo descentralizado y de interés público y social del estado de Quintana Roo.

La Universidad de Quintana Roo (UQROO) Atiende las demandas de Quintana Roo y del sureste mexicano, con proyección hacia Centroamérica y el Caribe. Incorpora adelantos tecnológicos en áreas sustantivas, como telecomunicaciones basadas en redes de fibra óptica e inalámbricas; edificios inteligentes, generación de energías sustentables, laboratorios y talleres equipados con tecnología de punta, instalaciones deportivas semiolímpicas, entre otras (Universidad de Quinatana Roo, 2011)

En la UQROO se encuentra el Centro de Tecnologías de la Información y la Comunicación (CETIC), albergando al departamento de Sistemas y el departamento de Cómputo y Telemática, que se encargan de proporcionar y brindar sus servicios a la comunidad universitaria. La segunda se organiza en: servicios de comunicación y colaboración, videoconferencias, soporte técnico, coordinación, redes y telecomunicaciones y desarrollo web.

En el área de desarrollo web; el cual es el encargado del desarrollo de sistemas para el departamento de cómputo y telemática propone el rediseño, desarrollo e implementación de su sistema de almacenamiento e intercambio de archivos, y brinda las características en hardware y software necesarias para su funcionamiento.

CAPÍTULO IV

METODOLOGÍA

3.2. Metodología de la investigación

Sampieri, Fernandez-Collado y Lucio (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2008) hacen mención de un párrafo: “Williams, Unrau y Grinnell (2005) establecen una excelente metáfora de lo que representa un planteamiento cualitativo: es como entrar a un laberinto, sabemos dónde comenzamos, pero no donde habremos de terminar. Entramos con convicción, pero sin un “mapa preciso.”

El tipo de estudio que presenta es de campo, ya que el entorno principal de trabajo para recabar información acerca del uso del nuevo sistema será en la UQROO, se indagará y se explorará acerca de la factibilidad de su uso, también se usarán técnicas de investigación como la observación, las entrevistas y la participación directa, además de aplicarse algunos instrumentos de investigación como son: entrevistas semi-estructurada y bitácoras de campo.

4.2. Metodología de software

Al surgir la necesidad de adaptar los sistemas informáticos a las exigencias del mercado, el programador realizaba un relevamiento de las solicitudes de quien necesitaba cierto programa o producto de software, y con aquellos requerimientos bajo el brazo comenzaba la dura tarea de codificar. Esta tarea no estaba administrada, supervisada o gestionada de ningún modo, por lo que se iba corrigiendo a medida que surgían lo errores, tantos los lógicos provenientes de la codificación, como los de requerimientos solicitados por el cliente o usuario final.

La metodología para el desarrollo de software es un modo sistemático de realizar, gestionar y administrar un proyecto para llevarlo a cabo con altas posibilidades de éxito.

Como se observa en la Ilustración 1.-Ciclo de vida de un producto de software. (Dante Cantone, 2006), desde un punto de vista general puede considerarse que el ciclo de

vida de un software tiene tres etapas claramente diferenciales las cuales se detallan a continuación:

- **Planificación:** se da la idea de un planteamiento detallado que guie la gestión del proyecto, temporal y económicamente.
- **Implementación:** se acuerda el conjunto de actividades que componen la realización del producto.
- **Puesta en producción:** el proyecto entra en la etapa de definición, allí donde se le presenta al cliente o usuario final, sabiendo que funciona correctamente y responde a los requerimientos solicitados en su momento.

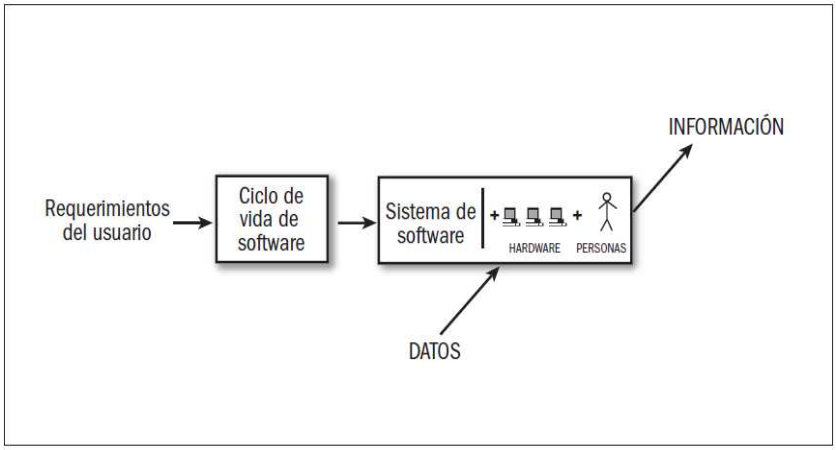

Ilustración 1.-Ciclo de vida de un producto de software. (Dante Cantone, 2006)

A estas tres grandes etapas es conveniente añadir otras dos que, si bien pudieron enunciarse junto a las otras, es conveniente hacer una diferenciación ya que se tiende a menospreciarlas o a no darles la importancia que requieren.

- **Inicio:** este es el nacimiento de la idea. Aquí se definen los objetivos del proyecto y los recursos necesarios para su ejecución.
- **Control de producción:** control del producto, analizando como el proceso difiere o no de los requerimientos originales e iniciando las acciones correctivas si fusen necesarias.

En cada una de las etapas de un modelo de ciclo de vida, se pueden establecer una serie de objetivos, tareas y actividades que lo caracterizan. Los cuales se describen a continuación:

- **Expresión de las necesidades:** esta etapa tiene como objetivo el armado de un documento en el cual se reflejan los requerimientos y funcionalidades que ofrecerá al usuario el sistema a implementar (qué, y no cómo, se va implementar).
- **Especificaciones:** formaliza los requerimientos; el documento obtenido en la etapa anterior se tomara como punto de partida para esta etapa.
- **Análisis:** se determinan los elementos que intervienen en el sistema a desarrollar, su estructura, relaciones, evolución temporal, funcionalidades, se tendrá una descripción clara de qué producto se va construir, qué funcionalidades aportara y qué comportamiento tendrá.
- **Diseño:** ahora que se sabe qué hacer, se tendrá que determinar cómo se deberá hacer (¿Cómo debe ser construido el sistema?; se definen en detalle entidades y relaciones de las bases de datos, se selecciona el lenguaje que se va utilizar, el sistema gestor base de datos, etc.).
- **Implementación:** se empieza por codificar algoritmos y estructuras de datos, definidos en las etapas anteriores, en el correspondiente lenguaje de programación o para un determinado gestor de base de datos.
- **Debugging:** el objetivo de esta etapa es garantizar que el programa no contenga errores de diseño o codificación.
- **Validación:** esta etapa verifica de que el sistema desarrollado cumple con los requerimientos expresados inicialmente por el cliente y que han dado lugar al proyecto.
- **Evolución:** en la mayoría de los proyectos se considera esta etapa como Mantenimiento y evolución, y se le asigna, no solo el agregado de nuevas funcionalidades (evolución); sino la corrección de errores que surgen (mantenimiento)

.

La metodología orientada a objetos es la adecuada para esta investigación ya que esta no comprende los procesos como funciones sino que arma módulos basados en componentes, es decir, cada componente es independiente del otro. Esto permitirá que el código sea reutilizable. Es más fácil de mantener porque los cambios están localizados en cada uno de estos componentes.

El ciclo de vida que se utilizará es el llamado en "**ciclo de vida en cascada puro**". Este modelo fue propuesto por Winston Royce en el año 1970. Es un ciclo de vida que admite iteraciones. Después de cada etapa se realiza una o varias revisiones para comprobar si se puede pasar a la siguiente. Como se observa en la Ilustración 2, es un modelo rígido, poco flexible, y con muchas restricciones.

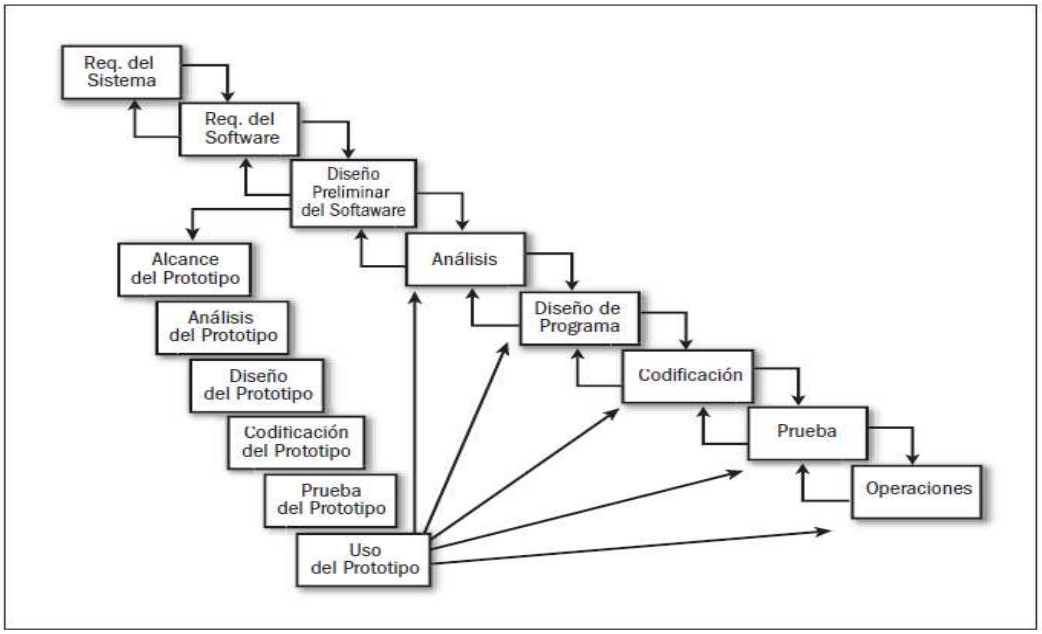

Ilustración 2.- Ciclo de vida en cascada puro. (Dante Cantone, 2006)

Una de sus ventajas, además de su planificación sencilla, es la de proveer un producto con un elevado grado de calidad sin necesidad de un personal altamente calificado.

CAPÍTULO V

DESARROLLO

5.1. Análisis de requerimientos

El Departamento de Desarrollo Web detectó ciertas deficiencias en el sistema de almacenamiento e intercambio de archivos (UQROOFTP), por esa razón se dio la tarea de mejorarlo y solucionar cualquier aspecto que fuera necesario.

La principal necesidad detectada fue facilitar la interacción con el usuario, por lo que se acordó desarrollar una versión mejorada del sistema de almacenamiento e intercambio de archivos para la UQROO. Para esto, se analizaron las características que presentaba la primera versión UQROOFTP:

- Autenticación con validación de usuario y contraseña del correo institucional.
- Interfaz poco amigable con el usuario.
- Solamente se podían compartir archivos individuales.
- Permitía 200MB de almacenamiento.
- Presentaba opciones de creación de carpetas, intercambio de archivos, carga de archivos, copiar, mover, pegar, renombrar y eliminar.

Por lo anterior, era necesario que la nueva versión presentara las siguientes funcionalidades:

Disponibilidad y rendimiento.

Garantizar la disponibilidad del sistema e incrementar la capacidad de almacenamiento, para esto es necesario analizar los requerimientos mínimos de hardware y software.

Seguridad.

Mejorar la seguridad al momento de autenticarse. Por lo cual es indispensable diseñar una estrategia para difundir las políticas de uso apropiado del sistema, así como excluir la navegación indeseable hacia archivos o carpetas que puedan acceder al servidor.

Facilidad de uso.

Una de las nuevas características del sistema es el cambio total de la interfaz, mejorando el entorno para una interacción confiable y amigable con el usuario, mantener y mejorar el acceso a todas las opciones del archivo.

Además de la opción de intercambio de un solo archivo, se agregaría la opción de intercambio de carpetas.

También se requiere mayor capacidad de almacenamiento.

5.2. Arquitectura de la aplicación

La arquitectura cliente servidor consiste básicamente en un cliente que realiza peticiones a otro programa (el servidor) que da la respuesta, tal como se observa en la Ilustración 3. En esta arquitectura la capacidad está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizacional debidas a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

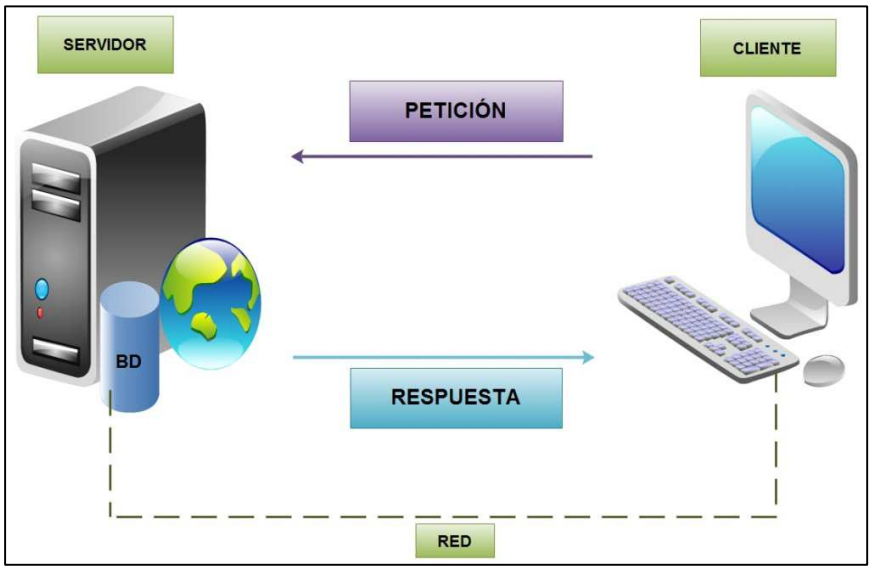

Ilustración 3.- Arquitectura cliente-servidor.

Las características que posee la arquitectura del servidor se muestran en la Ilustración 4.

Ilustración 4.- Características del servidor.

- Cool Stack

Esta es una colección de algunas de las aplicaciones de código abierto y se configuró sobre la plataforma Sun Solaris. Entre las aplicaciones que incluye se encuentran: Apache, PHP y MySQL. Con esta colección fue más fácil configurar los servidores web y base de datos con soporte para PHP.

- Servidor web Apache

Apache presenta características altamente configurables, bases de datos de autenticación y negociación de contenido. Además de ser el servidor http más usado es el componente en la plataforma de aplicaciones MySQL y PHP.

- Servidor de Base de Datos MySQL

Puesto que MySQL es un sistema de gestión de base de datos muy utilizada en aplicaciones web y está muy ligada a PHP, se configuró este servidor de base de datos.

- Aplicaciones de Administración.
  - Webmin

Esta herramienta configura aspectos internos del sistema operativo, también modifica y controla aplicaciones, en este caso el servidor web apache y el servidor de base de datos MySQL.

  - PhpMyAdmin

Esta herramienta administra la base de datos MySQL. Además de que se ejecuta bajo el servidor web Apache y tiene soporte para PHP.

Las características de la arquitectura del cliente se muestran en la Ilustración 5.

Ilustración 5.- Características del cliente.

- Aplicaciones Adobe Flash©

Una característica necesaria en el cliente es que el navegador que esté en uso tenga instalado soporte para aplicaciones Adobe Flash©.

- Navegador

El cliente deberá usar cualquier navegador para interactuar con el sistema, las versiones varían según las necesidades del usuario, por tal motivo se

recomiendan navegadores como Firefox 4, Internet Explorer 7, Google Chrome 15, Opera 9 y safari 5, o versiones superiores.

5.3. Diseño

Los puntos clave a considerar para desarrollar un sitio adecuado con contenidos útiles para los usuarios son:

- Detectar y entender las necesidades de los usuarios.
- Entender y ajustar la información al medio en el que se va a desarrollar (Internet).
- Planear antes de diseñar.

El sistema a desarrollar según la etapa de especificaciones presentará todas las características que mejorarán el aspecto del sistema y su funcionalidad.

Los siguientes son elementos que intervendrán en el desarrollo del sistema, basado en la estructura y funcionalidades:

- Mapa de navegación.
- Estructura o layout
- Módulo de autenticación
- Módulo de inicio
- Módulo de descarga

Mapa de navegación

La creación del mapa de navegación es útil en el desarrollo de los sitios ya que permite a través de una gráfica, mostrar las secciones que conformarán el sitio y los niveles de navegación para cada uno de los canales, tal como observamos en la Ilustración 6.

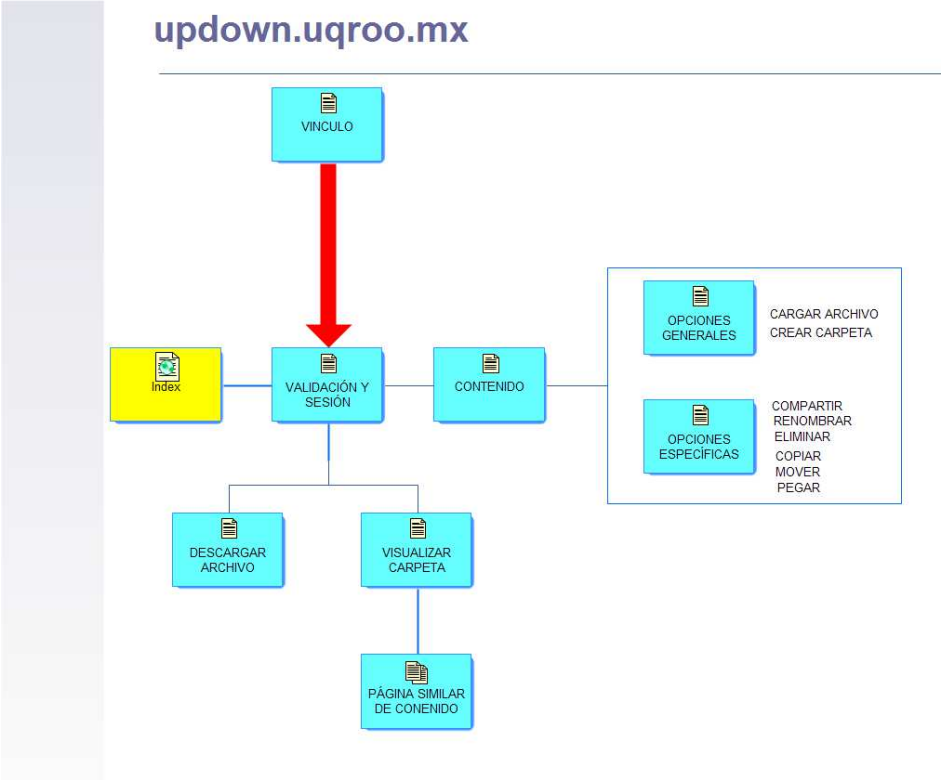

Ilustración 6.- Mapa de Navegación.

Los sitios web se diseñarán para una resolución de 1024 x 768 pixeles. Esta resolución debe considerar un área de seguridad, ya que los navegadores de internet utilizan espacio vertical para componentes como la barra de herramientas, y espacio horizontal para elementos como las barra de desplazamiento.

De tal forma, el área de seguridad de la resolución 1024 x 768 es aproximadamente de 982 x 600 como se observa en la Ilustración 7. Se recomienda ubicar cualquier elemento relevante del sitio dentro de las dimensiones del área de seguridad descrita.

Elementos adicionales del sitio fuera de los 600 pixeles de altura requerirán del desplazamiento de la barra para poder visualizarse

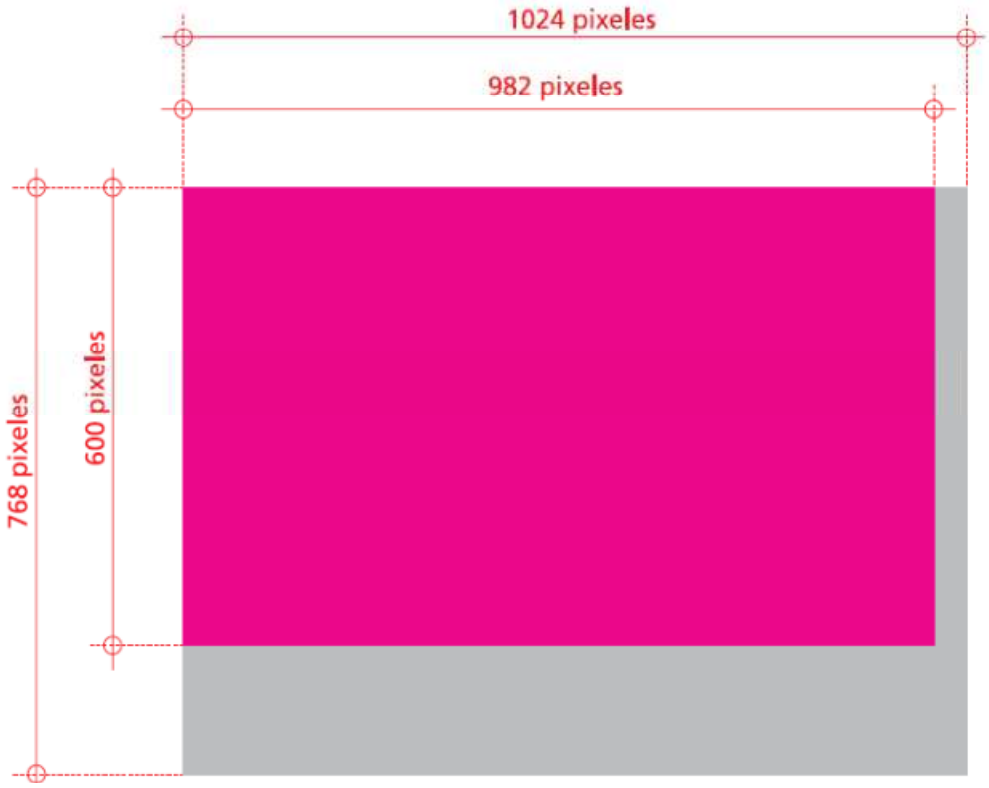

Ilustración 7.- Resolución de pantalla.

Estructura o layout

La estructura o layout del sitio web define la ubicación de los espacios de información y navegación, considerando los fundamentos de usabilidad (consistencia, jerarquización, fácil reconocimiento y aprendizaje de los espacios de información, etc.).

En términos generales, la estructura está diseñada para integrar los diferentes espacios de información a una resolución de 1024 x 768 pixeles, procurando páginas no saturadas. Cada pantalla se encuentra organizada en 3 columnas.

Los elementos que componen la estructura se detallan en las siguientes láminas, y corresponden a las pantallas de página principal, página índice y página interior o de contenido.

A continuación se detallan los elementos estructurales de cada página, donde se puede apreciar un principio importante de usabilidad: la consistencia en la ubicación de los elementos (ver Ilustración 8).

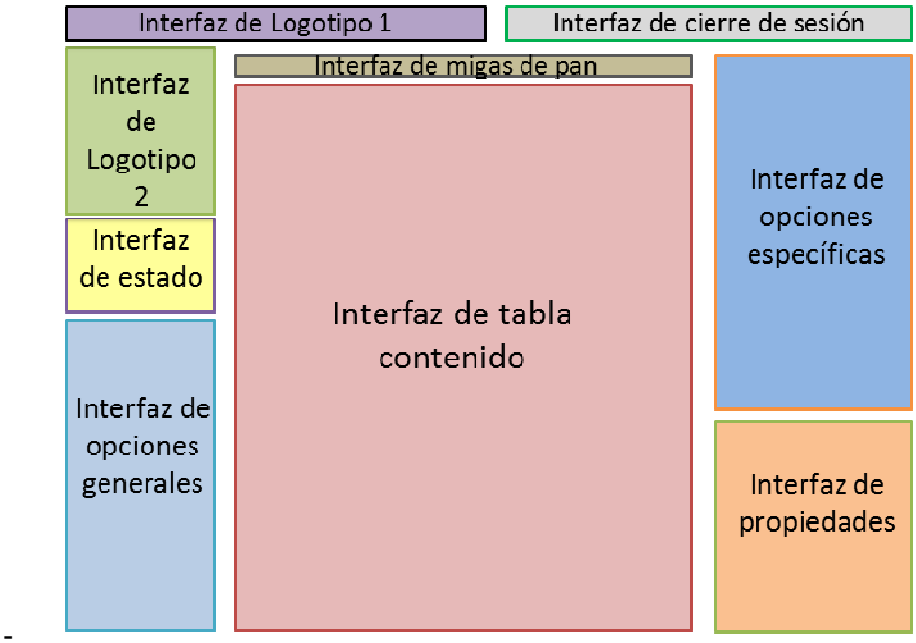

Ilustración 8.- Diseño de estructura files.php.

Módulo de autenticación

La ventana principal de inicio presentará un recuadro para la autenticación, donde el usuario registrará sus datos de correo electrónico y contraseña institucional, como se observa en la Ilustración 9.

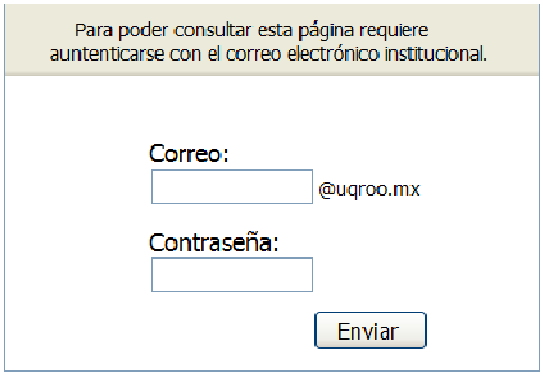

Ilustración 9.- Diseño de autenticación.

Módulo de inicio

Esta será la ventana que visualizará todos los archivos que podrán ser almacenados en el servidor, además de presentar todas las opciones que caracterizarán al sistema. Las opciones de esta ventana son:

- *Cargar archivo*, con esta opción el usuario podrá seleccionar el archivo que almacenará en el sistema.
- *Nueva carpeta*, con esta opción se crea una carpeta.
- *Tabla de archivo*, esta es una tabla que visualizará el nombre del archivo que ha sido alojado en el servidor, del lado izquierdo se visualizará un icono para identificar el tipo del archivo, del lado derecho tendrá una opción de tipo radio para seleccionar el archivo y poder ejecutar todas las opciones con las que cuenta el sistema.
- *Cerrar sesión*, con esta opción se cerrara la sesión iniciada.
- *Compartir*, esta opción permitirá generar un link de descarga para realizar el intercambio del archivo o carpeta. Abrirá una ventana con las siguientes opciones, ver Ilustración 10:
- *No compartir*, con esta opción se dejara de compartir cualquier archivo o carpeta.
- *Compartir sin contraseña*, al seleccionar esta opción en la parte inferior de la ventara se visualizará un cuadro de texto que contendrá el link generado para el intercambio.
- *Compartir con contraseña*, con esta opción se podrá introducir una contraseña para proteger al archivo. También se visualizará el cuadro de texto que contendrá el link para el intercambio.
- *Guardar cambios*, esta opción guarda todos los cambios generados.
- *Cerrar*, con esta opción cerramos la ventana.

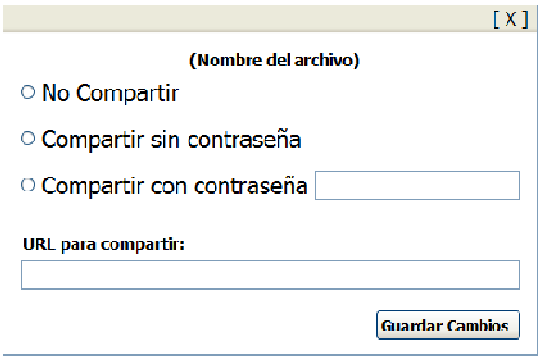

Ilustración 10.- Diseño de opciones para compartir.

- *Renombrar*, esta opción cambia el nombre del archivo o carpeta.
- *Mover*, con esta opción podremos cambiar de lugar el archivo o carpeta.
- *Copiar*, al presionar esta opción hacemos un duplicado de archivo o carpeta.
- *Pegar*, después de seleccionar las opciones de mover o copiar, con esta opción podremos aplicar los cambios para visualizar el archivo o carpeta en la nueva dirección.
- *Eliminar*, con esta opción eliminamos un archivo o carpeta.
- *Propiedades*, este será un cuadro que contendrá las descripciones generales del archivo o carpeta, como lo son; nombre, tamaño, fecha de modificación y tipo de archivo.

Otra de las características que presentará esta interfaz es la visualización de una barra de estado; el tamaño de almacenamiento otorgado, el tamaño disponible y el usado.

Módulo de descarga

En este módulo se agregará el link generado para ejecutar el intercambio de información, donde podrá descargase algún archivo o navegar por una carpeta compartida.

5.4. Implementación

En esta etapa se comienza la codificación del sistema, para este proceso se optó por codificar en lenguaje PHP.

PHP, acrónimo de *"PHP: Hypertext Preprocessor",* es un lenguaje de código abierto (*Open Source*) interpretado de alto nivel, orientado para desarrollo web, ya que puede ser embebido en páginas HTML. Su sintaxis es similar a C, Java y Perl, se considera fácil de aprender. PHP puede ser utilizado en cualquiera de los principales sistemas operativos del mercado, incluyendo Linux, muchas variantes Unix (incluido HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS y probablemente alguno más. PHP soporta la mayoría de servidores web de hoy en día, incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape y iPlanet, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd y muchos otros. PHP tiene módulos disponibles para la mayoría de los servidores, PHP también tiene soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros.

Para la codificación fue necesario tener instalado la herramienta de programación Macromedia Dreamweaver 8. Lo primero que se requiere para conectarnos al servidor de prueba; donde se colocará el sistema, es administrar un sitio a través de FTP (File Transfer Protocol).

Abrimos Dreamweaver y en primera instancia podremos observar una ventana como se observa en la Ilustración 11.

Ilustración 11.- Ventana principal Macromedia Dreamweaver 8.

En la barra de menú, seleccionamos la opción *sitio* y después *nuevo sitio*. Como en la Ilustración 12.

Ilustración 12.- Creación de un nuevo sitio.

Se abrirá una ventana y la primera opción que se seleccionara es *Datos locales* como se muestra en la Ilustración 13, después seleccionamos la opción *Datos remotos* como en la Ilustración 14, los campos se rellenaron con los datos correspondientes, por motivos de seguridad no se visualizarán datos importantes que puedan ocasionar conflictos con el administrador del sitio.

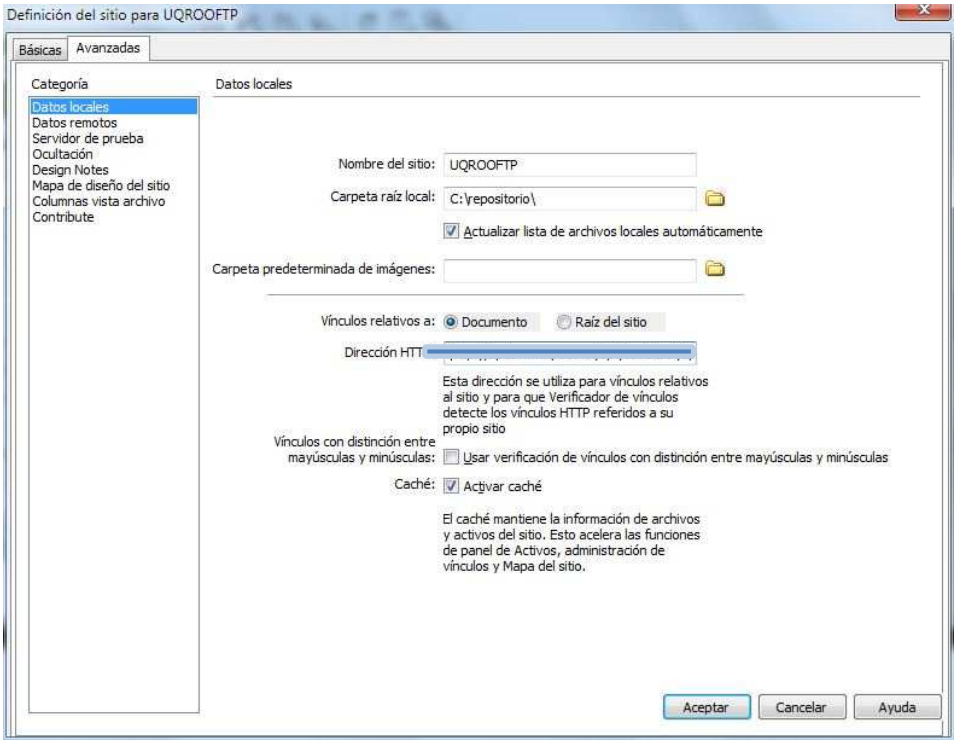

Ilustración 13.- Ventana de Datos locales.

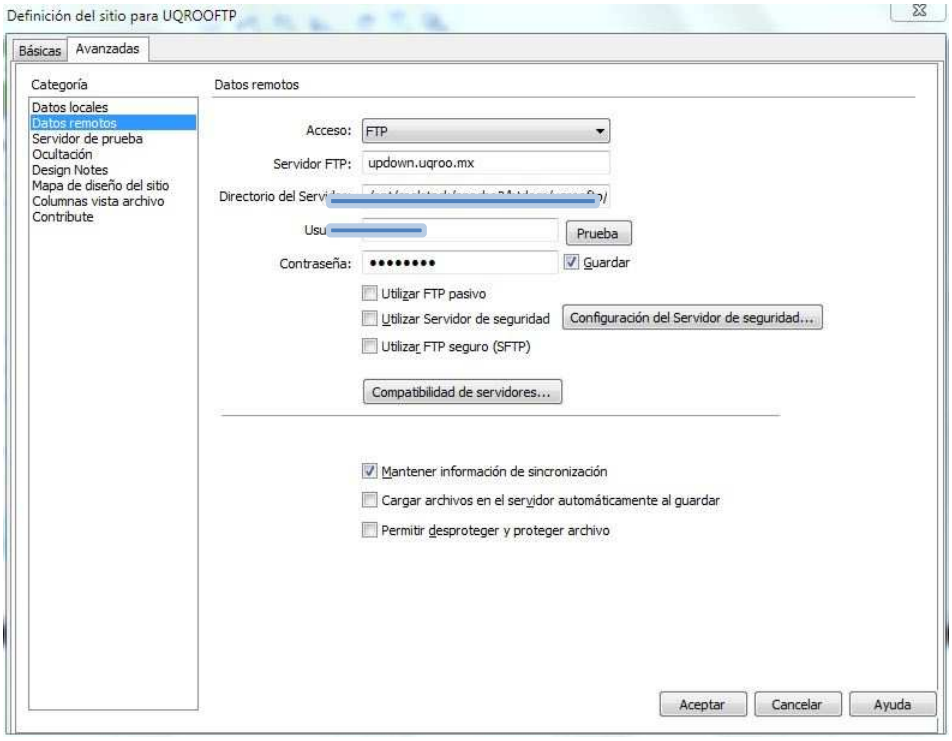

Ilustración 14.- Ventana de Datos remotos.

Después de haber creado el sitio, en la ventana principal se seleccionó la opción *crear nuevo,* en este caso todo el sistema se desarrollará mediante el lenguaje PHP, entonces se seleccionó la opción *PHP.*

Para interpretar un archivo, php simplemente interpreta el texto del archivo hasta que encuentra uno de los caracteres especiales que delimitan el inicio de código PHP. El intérprete ejecuta entonces todo el código que encuentra, hasta que encuentra una etiqueta de fin de código, que le dice al intérprete que siga ignorando el código siguiente. Este mecanismo permite embeber código PHP dentro de HTML: todo lo que está fuera de las etiquetas PHP se deja tal como está, mientras que el resto se interpreta como código.

Codificación de módulos

Módulo de inicio

EL primer archivo que se creó fue *files.php*, este archivo será la ventana principal del sistema, para su desarrollo se optó por el uso de tablas.

La interfaz del sistema se desarrolló para pantallas de 1024 pixeles, por lo que se tomó un ancho de la tabla de 982 pixeles y se mantuvo un margen de 22 pixeles por cada lado, como se observa a continuación:

```
<table    width="982"    height="600"    border="0"    align="center"
cellpadding="0" cellspacing="0" >

</table>
```

Desarrollo de Interfaces

Se codifico una lista para visualizar los archivos almacenados en nuestro servidor, en primera instancia el usuario observará una leyenda como la siguiente:

"Esta carpeta está vacía, añade archivos utilizando el botón "Cargar Archivo" que se encuentra a tu izquierda".

En el código para reconocer que se ha iniciado una nueva sesión se utiliza session_start() la cual identifica que se ha establecido una conexión con el servidor, esta conexión se describe más adelante en el módulo de autenticación. Se cargan todas las librerías necesarias para ejecutar el sistema, el método más importante es opendir que ejecuta la conexión con la ruta establecida para administrar las cuentas (carpeta creada con el nombre del usuario) de cada usuario. $_SESSION["DIR_USR"] almacena la dirección del usuario en el sistema. $_SESSION["DIR_ACTIVO"], esta no contiene nada al principio, va cambiando cuando el usuario navegue entre carpetas.

```
<?php
session_start();
include "sesion.php";
include 'class.DiskUsage.php';
include "class.uptodown.php";
include 'lib.php';
include 'replaceAcento.php';
include 'migas.php';
$obj = new DiskUsage;
$obj2 = new updown;
$directorio = opendir($_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]);
?>
```

Interfaz de almacenamiento

Hay dos formas de almacenar contenido en el servidor; mediante la carga de un archivo o la creación de una carpeta.

 ***Cargar Archivo,*** esta opción ejecuta una ventana modal, que contiene una SCRIPT la cual envía información a un archivo flash "formulario.swf" para realizar la carga de archivos al sistema, véase Ilustración 15.

$_SESSION["UPLOADS"] y $_SESSION["DISPONIBLE"], la primera almacena el tamaño del archivo que el usuario tiene permitido cargar al sistema, por defecto son 100 MB, la segunda almacena la cuota de la que dispone el usuario para subir archivos.

$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]."/" es la dirección donde se almacenará el archivo.

```
<script type="text/javascript">
            var so = new SWFObject("formulario.swf", "swf", "375",
"60", "8", "#F6F6F6");
            so.addVariable("direc", "<?php echo
$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]."/"; ?>");
            so.addVariable("maxupload", "<?php echo
$_SESSION["UPLOADS"]; ?>");
            so.addVariable("cuotadispone", "<?php echo
$_SESSION["DISPONIBLE"] ; ?>");
            so.write("formulario");
</script>
```

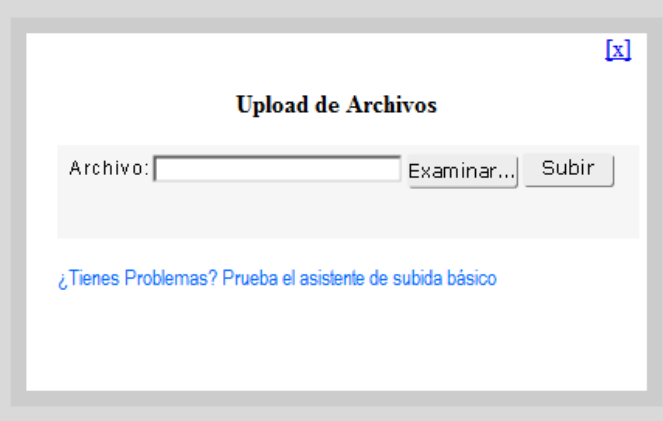

Ilustración 15.- Cargar archivos al sistema.

El mensaje que hace referencia al “asistente de subida básico”, genera otra manera de cargar archivos, si existiera algún problema de visualización del método de carga principal o el navegador tuviera problemas para ejecutar aplicaciones flash, se podrá optar por la segunda opción que ejecutará un método sin el uso de funciones flash, como se observa en la Ilustración 16.

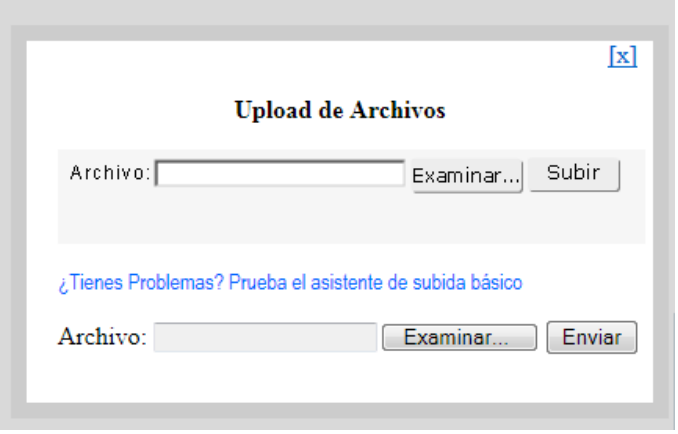

Ilustración 16.- Cargar archivos, segunda opción.

 ***Nueva Carpeta,*** la creación de carpetas es ejecutada mediante una función SCRIPT que genera una ventana como en la Ilustración 17, donde se debe ingresar el nombre que tendrá la carpeta para ser visualizado por el usuario.

El método mkdir es la que crea la carpeta en el sistema, y la guarda en la dirección actual en el sistema. $nombre contiene el nombre de la carpeta y se aplican los permisos correspondientes para crear la carpeta.

```
mkdir($_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]."/".$nombre,0777);
```

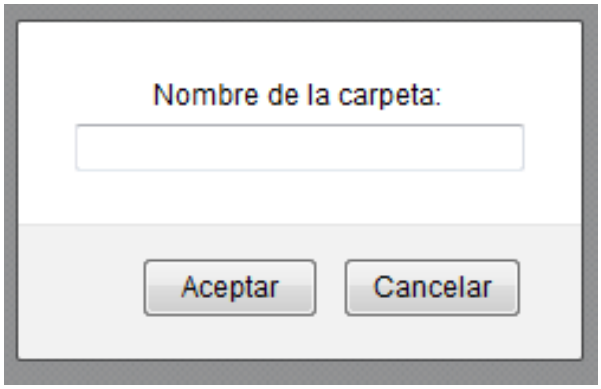

Ilustración 17.- Nueva carpeta.

Interfaz de tabla de contenido

Después de haber cargado el archivo o creado una carpeta, observaremos el contenido agregado al sistema y sus características, como se describe en la Ilustración 18.- Tabla de contenido.

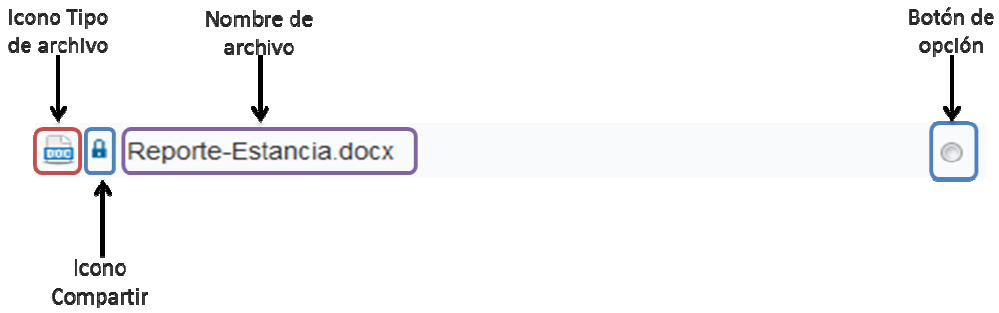

Ilustración 18.- Tabla de contenido.

El primer icono (de izquierda a derecha) mostrará el tipo de archivo que se ha cargado (se tiene una colección de íconos de los archivos más conocidos), si se seleccionó la opción “Nueva carpeta” el ícono que se mostrara será una carpeta.

Lo que realmente hace el código es un reconocimiento entre un archivo o una carpeta con los métodos is_file o is_dir respectivamente, antes recorre toda la dirección del archivo con el método explode(".", $nombre) hasta encontrar un punto, si existe un punto toma la extensión final, y la envía a $filetypes para su

reconocimiento, si la extensión es conocida mostrara en pantalla el icono correspondiente, de lo contrario aparecerá un icono de archivo desconocido. Por lo general si no encuentra ningún punto, el icono será una carpeta.

```
<?php
// codigo usado para encontrar el tipo de archivo
            $nombre = $archivo_completo;
            $partes = explode(".", $nombre);
            $extencion = end($partes);
            $exten = strtolower($extencion);
            if(is_file($archivo_completo)!=$filetypes[$exten]) {
                  echo '<img src="imgn/unknown.gif">';
               $tipo = 1;
               $typefiles = $exten;
                  }else
                  if(is_file($archivo_completo)==true) {
                  echo '<img src="imgn/'.$filetypes[$exten].'"/> ';

                  $tipo = 1;
                  $typefiles = $exten;
                  }else
            if(is_dir ($archivo_completo)==true){
               echo '<img src="imgn/folder.png">';
               $tipo = 2;
               $typefiles = "Carpeta";
               }
                  ?>
```

El segundo icono cambiará según la opción seleccionada al compartir un archivo, son tres tipos:

Icono	**Característica**
	Archivo No compartido.
	Archivo Compartido sin contraseña establecida.
	Archivo Compartido con contraseña segura.

```
<?php
// tippo de candado para visualizar si un archivo esta compartido
$imgid =$obj2->getllaves($nombreArch);
      if ($imgid==1){
            echo '<img src="img/candadocerrado.gif" width="16"
height="16">';
      }elseif ($imgid==2){
            echo '<img src="img/candadobierto.gif" width="16"
height="16">';
      }elseif ($imgid==3){
            echo '<img src="img/candadocerradoabierto.gif" width="16"
height="16">';
                  }
?>
```

El código anterior hace referencia a los iconos que podrán visualizarse. Para hacer este reconocimiento se creó una clase llamada class.uptodown.php, en esta clase se establece una conexión a la base de datos, en el código se manda llamar una función llamada a través de $obj2->getllaves($nombreArch) que se encarga de consultar en la base de datos y regresa un valor acerca del estado del archivo, es decir si se ha compartido.

La siguiente información que se observará es el nombre del archivo, en el código $nombreArch contiene el nombre del archivo, como se observa en el código hay un ciclo encargado de ir creando cada fila al momento de agregar más archivos al sistema, como se mencionó anteriormente al no existir ningún archivo mostrara un mensaje que puede observarse al final del código.

Otra característica es poder hacer clic sobre el nombre del archivo y ejecutar la descargar del mismo o navegar cuando se refiere a carpeta. Por último se encuentra el botón de opción, que al estar seleccionado el usuario podrá realizar cualquier acción en la interfaz de *"opciones"* y visualizar las propiedades del archivo en la interfaz "*propiedades*".

```
  <?php

// codigo para la tabla del contenido

  $band = 0;
  $c=0;
  $contar=0;

  while ($archivo = readdir($directorio))
   {
   $c++;
   if($archivo!="." and $archivo!=".." ){
   $band++;
   $archivo_completo =
$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]."/".$archivo;

  if(is_file($archivo_completo) or is_dir
($archivo_completo)==true){
      if(is_file($archivo_completo)) $tipo = "archivo";
      if(is_dir($archivo_completo)) $tipo = "directorio";
  $nombreArc = $archivo;
  $nombreArch = replaceAcento($nombreArc);
  ?>
  <tr <?php if($band==2) {echo 'class="gris"';} ?>>
    <td <?php if($band==2) {echo 'class="gris"';} ?> height="34"
align="center" valign="middle" bgcolor="#FFFFFF" class="etiqueta">
        <div align="center">
          <?php
<?php echo $nombreArch; ?>
<?php

       }$contar++;
       }
       // if . and ..
}//while

  ?>
                        </table>

      <?php if($c==2){ ?>
      <div style="min-height:100; margin-right:30px; background-
color:#FFFFFF">
               <div align="center" class="earchivos">Esta carpeta
está vacía, añade archivos utilizando el botón "Cargar Archivo" que
se encuentra a tu izquierda.</div>
        </div>

      <?php }?>
```

Interfaz opciones de archivo

Son varias las opciones que podremos ejecutar al seleccionar el archivo, estas aplicarán cambios que podrán visualizarse en la tabla de contenido, véase Ilustración 19.

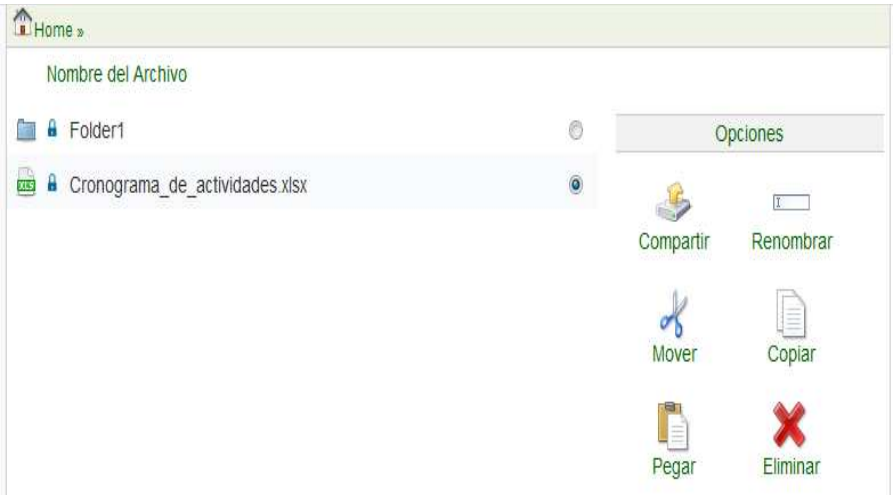

Ilustración 19.- Opciones de archivo.


Compartir, esta opción recurre nuevamente al uso de una ventana modal que genera diferentes métodos de seguridad para el intercambio, como se muestra en la Ilustración 20.

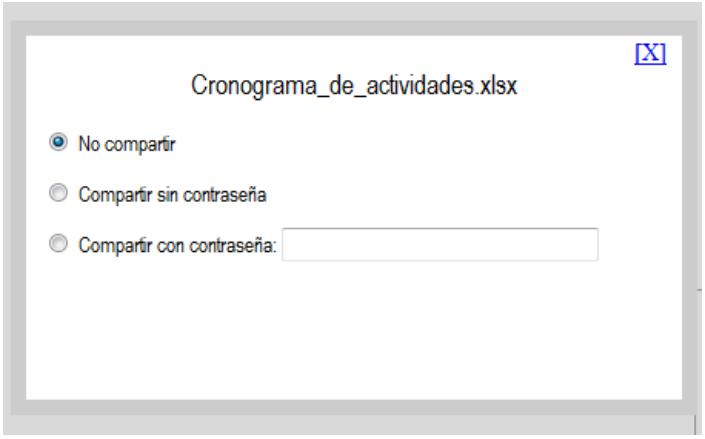

Ilustración 20.- Opciones de seguridad para intercambio.

Para generar la ventana modal se llama un script que la despliega, en el siguiente código observamos que contiene la ventana modal, en el código se inserta un formulario, se asigna un valor al radio que va de 1 a 3, por ejemplo value="1", mediante la función GET se realiza el envío de la información para insertar en la base de datos el archivo a compartir, getinsertar2 () es una función AJAX, que recibe la información, AJAX genera el proceso y manda por método GET esta información a un archivo llamado insertararchivo.php, este archivo recibe la información y realiza el proceso para general la URL para compartir el archivo.

```
<form id="formc" name="formc" method="post"  action=""
style="margin-left:10px; margin-top:0px">
  <label> <br />
  <input name="opcompartir" type="radio" value="1" />
  </label>
 No compartir
 <p>
    <label>
    <input onClick="getInsertar2()" name="opcompartir" type="radio"
value="2" />
    </label>
  Compartir sin contrase&ntilde;a </p>
 <p>
    <label>
    <input onClick="getInsertar2()" name="opcompartir" type="radio"
value="3" />
    </label>
    Compartir con contrase&ntilde;a:
    <label></label>
    <input name="pwd" type="password"  id="pwd" style="
width:200px;" maxlength="8" />
 </p>
  <div id="urlcopy" style="display:none;">
  <div>
  <strong>URL para compartir:</strong></div>

    <div align="right">
      <input readonly="readonly" name="urlcomp" type="text"
id="urlcomp" style=" width:400px;" value="" />
      <a href = "javascript:despliegaModal2('hidden');">
      <input onClick="getInsertar()" type="button" name="Submit2"
value="Guardar Cambios"></a>

  </div>
    </div>
</form>
```

Podremos observar en la Ilustración 21 el enlace generado para realizar el intercambio de archivos, en el código anterior se observa que antes de aparecer el enlace el cuadro de texto y el botón “Guardar cambios” permanecen ocultos.

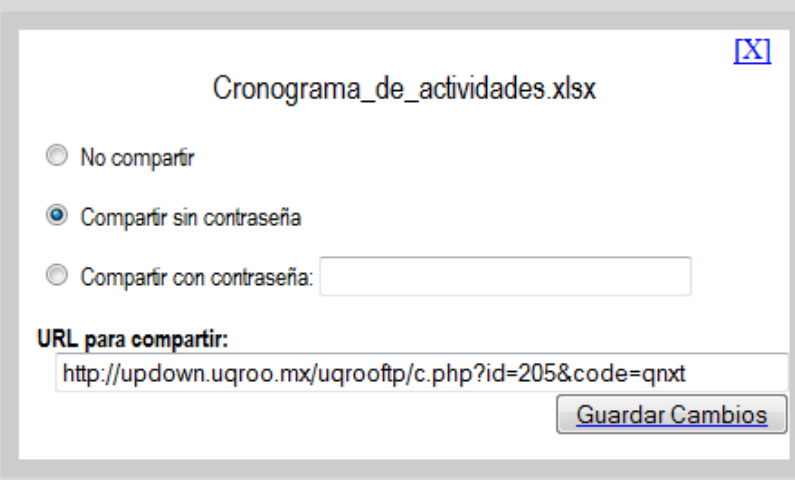

Ilustración 21.- Generar enlace para el intercambio.

Para establecer o realizar alguna modificación en el archivo compartido será necesaria seleccionar de nuevo el archivo y elegir la opción "*compartir*" nuevamente, los cambios serán aplicados al momento de dar clic en "Guardar Cambios", por lo general si el clic se realiza en el ícono de salida, los cambios realizados no se guardaran.

El nombre del archivo, el tipo de archivo, la opción escogida para el intercambio serán enviados por el método GET y se recibirán para almacenarlos en la base de datos con $obj->getInserta, si la opción escogida fue la primera en la base de datos se eliminara el archivo con $obj->eliminarBd.

Para generar el código propio para compartir el archivo se utiliza la función randomText().

```php
<?php
include "class.uptodown.php";
$obj = new updown;
$archivo = $_GET["archivo"];
$answer = $_GET["opcion"];
$tipo = $_GET["tipo"];

$l1=randomText();
$l2=randomText();
$l3=randomText();
$l4=randomText();
$code=$l1.$l2.$l3.$l4;

if($_SESSION["DIR_ACTIVO"]==""){
      $direcBd = $_SESSION["DIR_ACTIVO"].$archivo;
}else{
      $direcBd = $_SESSION["DIR_ACTIVO"]."/".$archivo;
}
if($answer == "1"){
$respo = $obj->eliminarBd($direcBd);

}elseif ($answer == "2") {

      $pwd="";
      $url = $obj->getInserta($archivo, $pwd, $code, $tipo);
echo $url;
    }
    elseif ($answer == "3") {

        $pwd = $_GET["pwd"];
        $url = $obj->getInserta($archivo, $pwd, $code, $tipo);
echo $url;
    }

function randomText() {
      $base = "123456789abcdefghijklmnpqrstuvwxyz";
            return($base{mt_rand(0,33)});
}

?>
```

Renombrar, como se observa en la Ilustración 22, se abre un cuadro de dialogo, en el código esta acción es aplicada por un SCRIPT, que solicita el nuevo nombre que tendrá el archivo o carpeta seleccionada y lo envía mediante el método GET al archivo *renombrar.php*, quien recibe la información y la contiene en $antiguo = $_GET["nom_antiguo"] y $nuevo =$_GET["nom_nuevo"], se utiliza el método rename para ejecutar el cambio de nombre del nuevo archivo, $_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"] contiene la ruta en el sistema en la que guardara el archivo con el nuevo nombre.

Cuando el archivo o capeta se encuentra compartido se ejecuta la función $obj2->renombrarBd($newdir,$oldir) que establece el cambio de la dirección del archivo con el nuevo nombre en la base de datos.

```
<?php
session_start();
include "class.uptodown.php";
$obj2 = new updown;

$antiguo = $_GET["nom_antiguo"];
$nuevo =$_GET["nom_nuevo"];
if($_SESSION["DIR_ACTIVO"]==""){
      rename($_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"].$antiguo,$
_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"].$nuevo);
      $oldir = $_SESSION["DIR_ACTIVO"].$antiguo;
      $newdir = $_SESSION["DIR_ACTIVO"].$nuevo;
}else{
      rename($_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]."/".$antig
uo,$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]."/".$nuevo);
    $oldir = $_SESSION["DIR_ACTIVO"]."/".$antiguo;
      $newdir = $_SESSION["DIR_ACTIVO"]."/".$nuevo;

      }
 $obj2->renombrarBd($newdir,$oldir);
?>
```

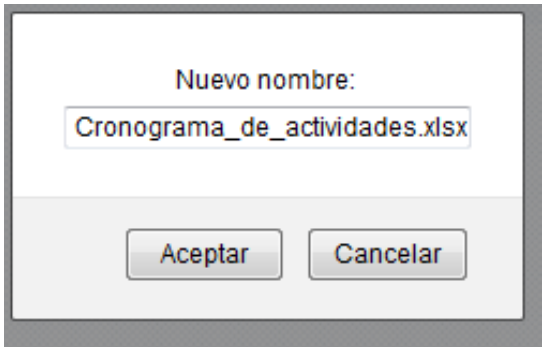

Ilustración 22.- Opcion renombrar.

Mover, esta opción es generada por una función llamada "*Moviendo*" que permite una copia del archivo, este $_SESSION["DIRECCION"]=$direccom almacena temporalmente la dirección del archivo mientras se navega entre carpetas, $_SESSION["DIREC_Bd"]=$direcBd guarda la nueva ruta del archivo cuando este se encuentra compartido. $_SESSION["RESPONSE"]="2" contendrá el número 2 como una opción, para que la opción pegar ejecute la acción de mover.

```
<?php
session_start();
$movefile=$_GET["archivo"];
$_SESSION["MOVER"]=$movefile;

if($_SESSION["DIR_ACTIVO"]==""){
      $direccom =
$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"].$_SESSION["MOVE
R"];
      $direcBd = $_SESSION["DIR_ACTIVO"].$_SESSION["MOVER"];
}else{
      $direccom =
$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]."/".$_SESSION["
MOVER"];
      $direcBd = $_SESSION["DIR_ACTIVO"]."/".$_SESSION["MOVER"];
}
$_SESSION["DIRECCION"]=$direccom;
$_SESSION["DIREC_Bd"]=$direcBd;
$_SESSION["RESPONSE"]="2";
?>
```

 **Copiar**, esta opción se codifico igual a la opción mover, con la única diferencia que $_SESSION["RESPONSE"]="1" ejecutara la opción para copiar.

```
<?php
session_start();
$movefile=$_GET["archivo"];
$_SESSION["MOVER"]=$movefile;

if($_SESSION["DIR_ACTIVO"]==""){
      $direccom =
$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"].$_SESSION["MOVER"]
;
      $direcBd = $_SESSION["DIR_ACTIVO"].$_SESSION["MOVER"];
}else{
      $direccom =
$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]."/".$_SESSION["MOV
ER"];
      $direcBd = $_SESSION["DIR_ACTIVO"]."/".$_SESSION["MOVER"];
}
$_SESSION["DIRECCION"]=$direccom;
$_SESSION["DIREC_Bd"]=$direcBd;
$_SESSION["RESPONSE"]="1";
?>
```

 Pegar, $respuesta contiene la opción seleccionada, 1 si es copiar, 2 si es mover, si la respuesta es la primera se ejecuta smartCopy($oldfile,$newfile) una función que realiza una copia inteligente del archivo a duplicar, el uso de esta función identifica si se desea copiar entre un archivo o una carpeta. Cuando la respuesta es la segunda se utiliza de nuevo la misma función con la diferencia de utilizar deleteDirectory($oldfile) para eliminar carpetas en la dirección anterior, en dado caso de ser un archivo compartido se ejecuta renombrarBd($newdir,$oldir) que reemplaza la dirección anterior con la nueva y elimina el archivo en la dirección anterior con el método unlink($oldfile).

```
$respuesta = $_SESSION["RESPONSE"];
if ($respuesta == '1'){
if ($sizes <= $_SESSION["DISPONIBLE"]){

      smartCopy($oldfile,$newfile);
      }
}

if ($respuesta == '2'){
      smartCopy($oldfile,$newfile);
      if(is_dir($oldfile)){
            deleteDirectory($oldfile);
            }else{
            $obj2->renombrarBd($newdir,$oldir);
            unlink($oldfile);
            }

      }
```

 **Eliminar**, al aplicar esta opción es necesario asegurarse del hecho que el archivo o carpeta seleccionado se eliminara por definitivo, para este proceso es necesario aplicar los permisos necesarios para eliminar el archivo chmod($direccom, 0777), se utiliza el método unlink($direccom) que elimina el archivo en la ruta actual, si el archivo esta compartido utilizamos $obj2->eliminarBd($direcBd), si en dado caso se elimina una carpeta se ejecuta deleteDirectory($direccom) una función que realiza una limpieza por toda la carpeta y subcarpetas hasta borrarla completamente, de la misma manera si esta compartida la carpeta se ejecuta $obj2->deleteCompartir($direcBd), que elimina la información de la base de datos.

Al eliminar estos archivos del sistema es necesario generar nuevamente la capacidad de almacenamiento actual del usuario, esto mediante $size = $obj->_directorySize($_SESSION["DIR_USR"]). $direccom hace referencia a la dirección actual del archivo.

```php
<?php
session_start();
include "class.uptodown.php";
include 'class.DiskUsage.php';
$obj2 = new updown;
$obj = new DiskUsage;
$file = $_GET["archivo"];

if($_SESSION["DIR_ACTIVO"]==""){
      $direccom =
$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"].$file;
      $direcBd = $_SESSION["DIR_ACTIVO"].$file;
}else{
      $direccom =
$_SESSION["DIR_USR"]."/".$_SESSION["DIR_ACTIVO"]."/".$file;
      $direcBd = $_SESSION["DIR_ACTIVO"]."/".$file;
}

if(is_file($direccom))
       {
            chmod($direccom, 0777);
            unlink($direccom);
                 $obj2->eliminarBd($direcBd);
        }

if(is_dir($direccom))
       {
             deleteDirectory($direccom);
                  $obj2->deleteCompartir($direcBd);
        }
           $size = $obj->_directorySize($_SESSION["DIR_USR"]);
           $_SESSION["free"] = $obj->_sizeFormat($size['size']);
?>


<?php
//Delete folder function
function deleteDirectory($direccom) {
          if (!file_exists($direccom)) return true;
    if (!is_dir($direccom) || is_link($direccom)) return
unlink($direccom);
        foreach (scandir($direccom) as $item) {
            if ($item == '.' || $item == '..') continue;
            if (!deleteDirectory($direccom . "/" . $item)) {
                chmod($dir . "/" . $item, 0777);
                if (!deleteDirectory($direccom . "/" . $item))
return false;
            };
        }
        return rmdir($direccom);
    }
?>
```

Interfaz de propiedades

El objetivo fundamental de esta interfaz es mostrar las características que posee el archivo o carpeta al estar seleccionado, como se observa en la Ilustración 23; el nombre, la fecha de modificación, el tamaño y el tipo de archivo. Entre las funciones correspondientes se pueden identificar diferentes métodos como “sizefile”, “datefile” y “typefile”.

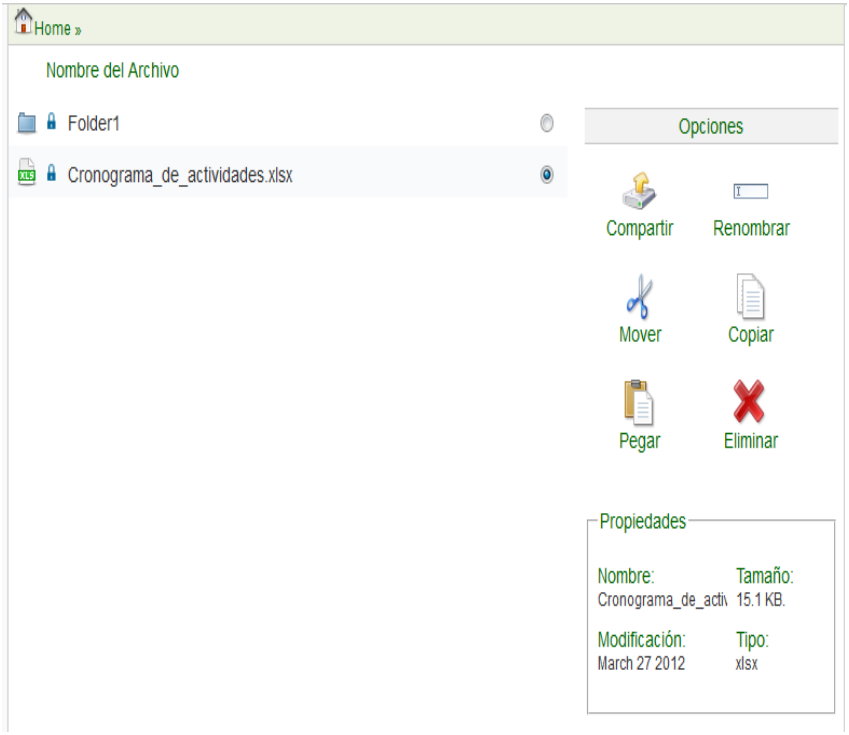

Ilustración 23.- Interfaz de propiedades.

Interfaz de estado

Esta interfaz tiene como objetivo proporcionar información sobre la cuenta del usuario como se observa en la Ilustración 24, esta información es obtenida en el proceso de autenticación y la contiene la sesión ($_SESSION["CUOTA"]) que se conecta a la base de datos, donde verifica la cuota asignada al usuario por el administrador.

Después establecida la sesión es llamada la función en el objeto $obj->_directorySize($_SESSION["DIR_USR"]), se encarga de verificar el peso total de los archivos cargados por el usuario para hacer la comparación de la capacidad restante se ejecuta $obj->_sizeFormat($size['size']) que da el formato adecuado a la cuota utilizada, es decir en KB, MG, o GB. y se almacena en $_SESSION["free"].

```
<?php
      // codigo usado para calcular espacio disponible y ocupado...
            $size = $obj->_directorySize($_SESSION["DIR_USR"]);
            $_SESSION["free"] =  $obj->_sizeFormat($size['size']);
            $sesionCuota = ($_SESSION["CUOTA"]*1024*1024);
            $_SESSION["DISPONIBLE"] = $sesionCuota - $size["size"];
            $bytesDisponibles = $obj-
>_sizeFormat($_SESSION["DISPONIBLE"]);
      ?>
```

La barra en gris muestra el porcentaje de la cuota utilizada, esta barra irá cambiando de color según sea el porcentaje, verde si es menor al 50 %, naranja si es mayor al 50 % y rojo si es mayor al 90 %. Los datos correspondientes al almacenamiento utilizado se visualizarán en las leyendas "Ocupado" y "Disponible".

Ilustración 24.- Interfaz de estado.

```
<div class="cuotas" style="margin-left:30px; width:154px; height:90px">
        <div style="line-height:20px"> Cuota:<?php echo
$_SESSION["CUOTA"]."MB";?></div>
<!-- Codigo para mostrar la barra de espacio -->
        <div class="cuotas2" style="width:145px; height:19px; float:left;
background-color:#CCCCCC; border-bottom:1px solid #999999; border-
left:1px solid #999999; border-right:1px solid #999999; border-top:1px
solid #999999;" >
            <?php $porcen = (($_SESSION["free"])*
100)/($_SESSION["CUOTA"]); ?>
                <?php
                            if($porcen >='1' and $porcen<'50'){
                             $colorf="#00CC33";
                            }else{
                            if($porcen >='50' and $porcen<'90'){
                             $colorf="#FFA500";
                            }else{
                            if($porcen >='90' and $porcen<='100'){
                             $colorf="#FF0000";
                            }}}

                            $trunc = (int)$porcen;
                ?>
            <span style="width:<?php echo $porcen;?>%; height:19px;
float:left; background-color:<?php echo $colorf;?>"><?php  echo
$trunc."%";?></span>
        </div>
        <div style="line-height:25px;">Ocupado:<?php echo
$_SESSION["free"]; ?> </div>
        Disponible:<?php echo $bytesDisponibles; ?>
      </div>
```

Interfaz de migas de pan

Esta interfaz muestra la navegación por las carpetas que el usuario recorre, como se observa en la Ilustración 25.

Ilustración 25.- Navegación por las migas de pan.

Mientras se recorre entre las carpetas se ira creando un enlace donde el usuario podrá acceder con más facilidad a cualquier carpeta anterior, para realizar esta acción se llama a la función crearpan($usr_activo), que recibe la dirección actual del archivo y mediante el método explode("/", $link) explorará la dirección hasta encontrar la última diagonal la cual devolverá el nombre de la carpeta actual y se ira almacenado como la carpeta actual. Es necesario crear una matriz que contendrá el nombre de la carpeta y su dirección, para que al hacer clic pueda dirigirnos hacia la opción seleccionada.

El valor que devuelve return ($liga) es la manera en que se visualizaran las migas de pan en la interfaz como se observó en la Ilustración 25.

```
<?php
function crearpan($usr_activo){
$link =$usr_activo;

// crear partes de  la direccion
$parte = explode("/", $link);
$liga2 = "";
// recorrer el array creado
for ($i=0;$i<count($parte);$i++){
$url .= $liga2.$parte[$i];
$liga2 = "/";
$liga .= '&raquo; <span onclick="cambiadir('."'".$url."'".')"
style="cursor:pointer" >'.$parte[$i].'</span> ';

}
return ($liga);
}
```

Durante la navegación entre las carpetas, el usuario tiene la opción de volver hacia atrás, con la opción “Subir Nivel”, la cual aparecerá debajo de las migas de pan como se observa en la Ilustración 26 .

Ilustración 26.- Navegación entre carpetas.

Cuando el usuario hace clic en la opción "Subir Nivel",Anteriormente se mencionó que $_SESSION["DIR_ACTIVO"] almacenará la dirección actual en la que se encuentra el usuario, en el código si esta acción difiere, el método strrpos recorrerá la dirección hasta encontrar la última diagonal, devolverá el nombre del archivo encontrado, y con el método substr regresaremos una posición antes en la dirección actual.

```
<?php
session_start();
      if($_SESSION["DIR_ACTIVO"] !=""){
            $pos  = strrpos($_SESSION["DIR_ACTIVO"], '/');
       $_SESSION["DIR_ACTIVO"]= substr($_SESSION["DIR_ACTIVO"],0,$pos);
            }
?>
```

Interfaz cierre de sesión

$_SESSION["band_acc_updown"]= false indica que la sesión ha finalizado, y establecemos el final del lenguaje php exit(), además de indicarle que nos envíe a la página de autenticación.

```
<?php
session_start();
$_SESSION["band_acc_updown"]= false;
?>
<META HTTP-EQUIV="Refresh" CONTENT="0; URL=index.php">
<?php exit();?>
```

En el archivo files php, indicamos que la sesión ha finalizado y cerramos el acceso al directorio actual del usuario.

```
<?php closedir($directorio);  ?>
```

Módulo de autenticación

En la Ilustración 27 se observa el uso del correo institucional como la opción más confiable para la conexión al sistema, y es el requisito principal para obtener una cuenta en el sistema, se solicita al usuario ingresar sus datos al inicio, si algún dato es incorrecto la página se refrescara para solicitarlos nuevamente.

Ilustración 27.- Módulo de inicio.

El proceso a realizarse es una consulta al servidor LDAP de la universidad, el archivo *index.php* contiene un formulario que solicita el correo y contraseña, cuando los datos son correctos serán enviados mediante el método *post* al archivo *valida.php*.

```
<form id="formuqroo" name="formuqroo" method="post" action="valida.php">
```

El uso de sesiones; session_start(), es una parte esencial del sistema, puesto que crea el acceso a un usuario nuevo. La clase class.AuthLdap.php se encarga de identificar si el usuario y password introducidos pertenecen al correo institucional son correctos.

```
<?php
session_start();
   include "class.uptodown.php";
   $obj2 = new updown;
   $dir_server =  "172.16.2.106 389"; // Primary LDAP server
   $base_dn = "ou=people,o=uqroo.mx,o=isp"; // Base DN of our
organisation
   include "ldap/class.AuthLdap.php";
      $LDAP_Username = $_POST["usuario"];
      $LDAP_Password  = $_POST["pass"];
      if(isset($_POST["enviarr"]) and $LDAP_Username!="" and
$LDAP_Password!=""){

      $ldap = new AuthLdap();
      $server[0] = $dir_server;
      $ldap->server = $server;
      $ldap->dn = $base_dn;
```

Ya autenticado el usuario se establece la conexión y se inicia una nueva sesión, donde se identifica al usuario en la siguiente dirección $_SESSION["DIR_USR"]="/opt/sitios/updowndata/".md5($LDAP_Username).

Una acción utilizada para prevenir el uso inadecuado del sistema, es notificar al usuario las políticas de uso del sistema, por lo que para acceder completamente se llama a una función SCRIPT, que genera una ventana con las políticas que el usuario deberá aceptar. Cuando el usuario acepta todos los términos y condiciones de uso quedará registrado en la base de datos del servidor, mediante el método mkdir crea una carpeta en el servidor con su nombre y se le asigna una cuota de almacenamiento, por defecto son 200 MB.

```
if($ldap->connect()) {
       if($ldap->checkPass($LDAP_Username,$LDAP_Password)&&
!empty($LDAP_Username)){
             //(cn= todo el nombre, givenname= solo nombre, sn= apellidos)
             $name_user= $ldap->getAttribute($LDAP_Username,"cn");

            $_SESSION["nombreu"]=$name_user[0];
            $_SESSION["nickname"]=$LDAP_Username;
            $_SESSION["band_acc_updown"]=true;
      $_SESSION["DIR_USR"]="/opt/sitios/updowndata/".md5($LDAP_Username);
            $_SESSION["DIR_ACTIVO"]="";
            $_SESSION["CUOTA"]=$obj2->getCuota($LDAP_Username);
            $_SESSION["ACCESO"]=$obj2->getAcceso($LDAP_Username);
            $_SESSION["UPLOADS"]=$obj2->getUploads($LDAP_Username);
            if($_SESSION["CUOTA"]==""){
            $_SESSION["CUOTA"]=200;
            }
            if($_SESSION["UPLOADS"]==""){
            $_SESSION["UPLOADS"]=200;
            }
            if($_SESSION["ACCESO"]==""){
            ?><script language="javascript">
            window.open('politicas.php','_self'); </script>
            <?php
            if(!file_exists($_SESSION["DIR_USR"])){
                  mkdir($_SESSION["DIR_USR"]);
                  $_SESSION["free"] = '1Kb';
            }
}
            ?>
            <META HTTP-EQUIV="Refresh" CONTENT="0; URL=files.php">
            <?
        }else{
        ?>
            <META HTTP-EQUIV="Refresh" CONTENT="0;
URL=index.php?mensaje=Correo y/o contraseña incorrecta.">
      <?
      }
        }
        $ldap->close();
   }else{
    ?>
            <META HTTP-EQUIV="Refresh" CONTENT="0; URL=index.php">
      <?
   }
    ?>
```

De lo contrario, si el usuario existe en la base de datos, con la clase 'clas.uptodown.php' se obtiene información acerca de su cuota para la actualización del estado de almacenamiento.

Lo importante al realizarse el proceso de conexión sucede al crear la carpeta, puesto que el nombre del usuario se ocultará mediante un método de cifrado en el servidor. Con los términos de servicio y el cifrado se aumenta la seguridad de la conexión.

Módulo de descarga

Este módulo se nombró *c.php* y solamente funcionará si el link de descarga proporcionado es correcto, como se observa en la Ilustración 28 el link generado para el intercambio contiene un *id* y un *código* que será consultado en la base de datos para iniciar la descarga del archivo o la visualización de la carpeta.

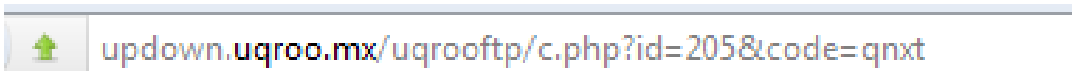

Ilustración 28.- Enlace de descarga.

Nuevamente se inicia una sesión que permitirá la conexión al archivo compartido, se recurre a la clase 'class.uptodown.php' que en base al id y al código realiza una consulta que retorna la dirección del archivo, la contraseña asignada (si es que posee), el nombre del usuario y el tipo de descarga (si es un archivo, o una carpeta), el uso de la sesión $_SESSION["DIR_USRO; permite vincular la dirección del usuario que ha compartido el archivo. Y el uso de la sesión "]$_SESSION["DIR_ACTIVOS"]=""; guardara la dirección actual para navegar en el caso del intercambio de carpetas.

```php
<?php
session_start();
include "class.uptodown.php";
$obj = new updown;

$id = $_GET['id'];
$code = $_GET['code'];
 $archivo = $obj->compEnlace($id,$code);
 $password = $obj->password($id,$code);
 $nickname = $obj->nickname($id,$code);
 $tipos = $obj->tipos($id,$code);

$mensaje = $_GET['mensaje'];
$nombre = $archivo;
$partes = explode("/", $nombre);
$extencion = end($partes);

$_SESSION["DIR_ACTIVOS"]="";
$nom_arch = "/opt/sitios/updowndata/".md5($nickname);

$_SESSION["DIR_USRO"] = $nom_arch."/".$archivo;

?>
```

Una característica de esta interfaz; como se observa en la Ilustración 29, es la de mostrar el nombre del archivo consultado, otra es identificar el uso de contraseña. Si existe tal contraseña se visualizará un cuadro de texto donde se proporciona dicha información, si esta es incorrecta aparecerá una leyenda que solicita nuevamente dicha información.

Ilustración 29.- Interfaz de descarga.

La característica más importante es que identifica si se ha compartido un archivo o una carpeta.

```
if ($tipos == 1){
 echo "descargar el archivo";
 $imgen = "mimorral/descargar.jpg";
  }else{
 echo "visualizar la carpeta";
 $_SESSION["entrar"] = true;

 $imgen = "mimorral/descargar2.png";
}
        ?>
```

Como se observó en la Ilustración 29, si este es un archivo aparece un icono que dice “Descargar ahora”, de lo contrario como se observa en la Ilustración 30, la figura dirá “Visualizar ahora”.

Ilustración 30.- Interfaz de visualización de carpeta.

Cuando el enlace compartido hace referencia a una carpeta, el usuario podrá recorrer dicha carpeta y tendrá los permisos para iniciar cualquier descarga al hacer clic sobre el archivo, como se observa en la Ilustración 31.

Ilustración 31.- Interfaz de navegaciòn de carpeta.

Todo el código para recorrer la carpeta y para descargar el archivo, es igual a lo antes referido en el módulo de contenido.

Implementación AJAX

Ajax viene de la palabra Asynchronous JavaScript And XML ajax no es una tecnología, un módulo de apache, un script, una extensión, si no es más bien una mezcla de tecnologías, que permiten crear aplicaciones web, increíbles y dinámicas.

Se trata de una mezcla de tecnologías:

- Uso de estándares XHTML y CSS
- Uso del Document Object Model(DOM)
- Interacción de datos usando XML, XSLT, HTML, hasta JS y Paginas Dinámicas (ASP, PHP, CGI, PYTHON, NET, Etc.).
- Y lo indispensable JAVASCRIPT, para juntar y organizar todo.

Las funciones AJAX mejoran el tiempo de ejecución de cada operación ejecutada por el usuario.

Para poder hacer peticiones remotas, es necesario del uso de un objeto, el XMLHttpRequest, que a pesar de ser un estándar no está bien implementado en algunos navegadores como, Internet Explorer, por ello es necesario hacer una función que genere correctamente este objeto.

Y después solo lo instanciamos una variable a partir de la función var xmlhttp=false.

```
function nuevoAjax()
{
	var xmlhttp=false;
	try
	{
		xmlhttp=new ActiveXObject("Msxml2.XMLHTTP");
	}
	catch(e)
	{
		try
		{
			xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
		}
		catch(E) { xmlhttp=false; }
	}
	if (!xmlhttp && typeof XMLHttpRequest!='undefined') { xmlhttp=new
XMLHttpRequest(); }

	return xmlhttp;
}
```

Ahora creamos las funciones que se encargaran de hacer la petición a la página web. Entre las principales funciones encontramos algunas opciones de las interfaces principales.

Obtenemos el valor del campo, el cual está referenciado por un id, esta es una de las formas más rápidas para acceder a un elemento en específico.

ajax.open("GET","subir_archivo.php",true) el objeto ajax tiene una serie de métodos, que harán posible que nuestra aplicación funcione, el primer método se llama open y abre una conexión entre nuestra página web y otra página(en este caso, la pagina se llama subir_archivo.php), este método usa 3 parámetros:

1. Método de envió de datos, GET o POST
2. El segundo parámetro es la página a la cual deseamos conectarnos, esta puede o no llevar datos, por ejemplo si usamos el método get, es necesario enviar variables, por este medio, si es por post, no es necesario.
3. Y por último un valor booleano para saber si al conexión será Asíncrona o síncrona, y se define con las palabras reservadas o valores, true o false.

Al realizar una petición ajax es necesario revisar el estado en el que está en objeto, esto se refiere a, preguntar si la página que pedimos se cargó, si ajax aún está enviando datos, esperado respuesta, o si la página devuelta existe, entre otros.

En este caso debemos de referirnos al método onreadystatechange. Con este método nos referimos a una función la cual revisará el estado del objeto ajax, el cual consta de 5 estados.

(1). No inicializado
(2). Conexión establecida
(3). Recibiendo respuesta
(4). Procesando respuesta
(5). Finalizado

Con el método onreadystatechange generamos una función que comprueba si el estado es igual a 1 (si no se ha inicializado), y 4, mediante el metodo ajax.readyState, (si se ha finalizado la petición).

Cuando la petición es correcta se refrescara la página con location.reload(true).

```
function subir_archivo(){
      var ajax=nuevoAjax();
      ajax.open("GET","subir_archivo.php",true);
      ajax.setRequestHeader("Content-Type", "application/x-www-
form-urlencoded");
      ajax.onreadystatechange=function(){
            if (ajax.readyState==1){
                              }
            if (ajax.readyState==4){
                        location.reload(true);

            }
      }
      ajax.send(null);

}
```

El último método usado es. ajax.send(null); Este método permite enviar variables, cuando utilizamos el método post para enviar datos, si en cambio se usa el método get, simplemente se concatenan la variables al nombre de la página.

Con las siguientes funciones Ajax se ejecuta de igual manera, por lo que no será necesario mencionar su desarrollo, lo importante es conocer la estructura principal de esta mezcla de tecnologías llamada Ajax.

Implementación de Base de Datos

MySQL, el sistema de gestión de bases de datos SQL Open Source, muy rápido, fiable y fácil de usar, trabaja en entornos cliente/servidor o incrustados, algunas de las características más importantes del software de base de datos MySQL son:

- Escrito en C y en C++
- APIs disponibles para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, y Tcl.

- El servidor está disponible como un programa separado para usar en un entorno de red cliente/servidor.
- Un sistema de privilegios y contraseñas que es muy flexible y seguro, y que permite verificación basada en el host.

Tres de las principales funciones que realizamos en la base de datos, suceden al autenticarse, al compartir información y al descargarla.

Creamos un archivo llamado class.uptodown.php donde se definió la clase *updown*. *Upload* es el nombre asignado a nuestra base de datos, y para conectarse utilizamos mysql_connect, establecemos los parámetros de conexión con el usuario y contraseña. Con esto podremos ingresar a la base de datos y realizar los movimientos necesarios.

```
class updown{
var $link;
function updown(){
      $this->link = mysql_connect('localhost','backpack','JC_Back11');
      mysql_select_db('upload',$this->link);
  }
```

La función getCuota () es la encargada de hacer una consulta a la tabla llamada "*cuotas"* mediante el uso del nickname del usuario, que es el correo electrónico, donde $row=mysql_fetch_assoc($result) busca si el usuario se encuentra en la base de datos y $cuota=$row["cuota"]; almacena la cuota que le fue asignada al usuario, esta función es necesaria en la interfaz de barra de estado.

```
function getCuota($nickname){
  $result=mysql_query("SELECT * FROM cuotas WHERE
nickname='$nickname'",$this->link);
  $row=mysql_fetch_assoc($result);
  $cuota=$row["cuota"];
  return ($cuota);
  }
```

La función AcInsert() se ejecutará cuando el usuario acepte las políticas de uso del sistema, cuando el usuario selecciona la opción SI esta será insertada en la base de datos en la tabla accesos, de otra manera arrojara un error de conexión.

```
function AcInsert(){
  $usr = $_SESSION["nickname"];
  $valor = 'si';
  mysql_query("INSERT INTO accesos
VALUES('$usr','$valor')",$this->link) or die(mysql_error());
   }
```

La función getInserta($archivo, $pwd, $code, $tipo) ejecuta diferentes funciones donde recibe el nombre del archivo, la contraseña si fue asignada, el código generado y el tipo de archivo (1 si es archivo o 2 si es carpeta), la primera acción que se realiza es reconocer la dirección en la que el usuario se encuentra para guardarla, esto mediante el uso de ($_SESSION["DIR_ACTIVO"], lo siguiente es consultar en la tabla compartidos si el usuario ha compartido el mismo archivo, por lo general si esto es así, con el método UPDATE actualizará la información en la tabla con los nuevos datos y regresará el mismo enlace generado con anterioridad.

De lo contrario SELECT MAX(id) generará un nuevo id del archivo y utilizara el método INSERT para guardar los datos del archivo compartido y regresará el enlace generado con la información necesaria para su futura descarga.

```
function getInserta($archivo, $pwd, $code, $tipo){
      if($_SESSION["DIR_ACTIVO"]=="")
            $archivos = $_SESSION["DIR_ACTIVO"].$archivo;
      else
            $archivos = $_SESSION["DIR_ACTIVO"]."/".$archivo;

      $usr = $_SESSION["nickname"];

      $resultExist=mysql_query("SELECT * FROM compartidos WHERE
nickname='$usr ' and  archivos ='$archivos'",$this->link) or
die(mysql_error());

      if(mysql_num_rows($resultExist)>0){
            $row=mysql_fetch_assoc($resultExist);
            $id = $row["id"];
            $code = $row["code"];
            mysql_query("UPDATE compartidos SET password='$pwd'
WHERE id = $id ",$this->link) or die(mysql_error());

            return
("http://updown.uqroo.mx/uqrooftp/c.php?id=$id&code=$code");
      }else{
            $result=mysql_query("SELECT MAX(id) AS max_id FROM
compartidos",$this->link) or die(mysql_error());
            $row=mysql_fetch_assoc($result);
            $id=$row["max_id"]+1;
            mysql_query("INSERT INTO compartidos
VALUES($id,'$usr','$archivos','$pwd','$code',$tipo)",$this->link)
or die(mysql_error());
            return
("http://updown.uqroo.mx/uqrooftp/c.php?id=$id&code=$code");
      }

      }
```

La función getllaves($archivo) se encarga de mostrar en la interfaz de contenido los iconos que aparecerán si el archivo se encuentra compartido. Como se mencionó anteriormente son tres tipos de iconos, la función al recibir la dirección del archivo lleva a cabo la consulta en la tabla *compartidos*, regresará un valor 1 si el archivo no se encuentra, 2 si el archivo se encuentra sin contraseña y 3 si el archivo posee una contraseña.

```
function getllaves($archivo){

if($_SESSION["DIR_ACTIVO"]=="")
      $archivos = $_SESSION["DIR_ACTIVO"].$archivo;
else
      $archivos = $_SESSION["DIR_ACTIVO"]."/".$archivo;

$usr = $_SESSION["nickname"];
$result=mysql_query("SELECT * FROM compartidos WHERE nickname='$usr'
and archivos = '$archivos'",$this->link) or die (mysql_error());
$row=mysql_fetch_assoc($result);

$id= $row['id'];
$pwd = $row['password'];

if($id == "" and $pwd == "" ){
return 1;
}elseif($id != "" and $pwd == ""){
return 2;
}elseif($id != "" and $pwd != ""){
return 3;
}


}
```

Anteriormente se mencionó que cuando el archivo quiere eliminarse y se encuentra compartido es necesario eliminarlo de la base de datos, la función eliminarBd($archivo) recibe la dirección del archivo y mediante el método DELETE elimina el archivo que ha sido compartido por $_SESSION["nickname"] que es el usuario activo.

```
function eliminarBd($archivo){
   $usr = $_SESSION["nickname"];
   mysql_query("DELETE FROM compartidos WHERE nickname='$usr' and
               archivos='$archivo'",$this->link)or
               die(mysql_error());
}
```

De igual manera si el usuario cambia el nombre del archivo que ha compartido, será necesario cambiarlo en la base de datos, con la función renombrarBd($nuevo, $antiguo) recibimos la dirección anterior y la nueva dirección del archivo y

mediante el uso del método UPDATE indicamos donde se encuentra el archivo que requerimos cambiar. Para esto también se utiliza $_SESSION["nickname"] para reconocer al usuario activo.

```
function renombrarBd($nuevo, $antiguo){
 $usr = $_SESSION["nickname"];
 mysql_query("UPDATE compartidos SET archivos='$nuevo' WHERE
nickname='$usr' and archivos='$antiguo' ",$this->link)or
die(mysql_error());

}
```

Para generar la descarga es necesario consultar si el enlace compartido es correcto, la función compEnlace($id, $code) se encarga de consultar en la base de datos en la tabla compartidos si el id y el code existen. Si esto es así devolverá la dirección del archivo o carpeta que se ha compartido y la vinculara como se indica en el módulo de descarga, de lo contrario mostrara la leyenda “URL INCORRECTA”.

```
function compEnlace($id, $code){
$resultExist=mysql_query("SELECT * FROM compartidos WHERE
id='$id' and code='$code'",$this->link) or die(mysql_error());

if(mysql_num_rows($resultExist)>0){
  $row=mysql_fetch_assoc($resultExist);
  $archivos=$row["archivos"];
  return ($archivos);
  }
else{
  echo "Url incorrecta";
  exit();
  //return ();
  }
}
```

Si el archivo se ha compartido mediante una contraseña, también será necesario consultar si la contraseña que introducirá el usuario es correcta, para esto se utiliza la función password($id, $code) que mediante el id y el code consulta en la tabla compartidos si el archivo tiene contraseña, si esto es así en el módulo de descarga podrá observarse que la contraseña es correcta y podrá acceder al archivo.

```
function password($id, $code){

$resultExist=mysql_query("SELECT * FROM compartidos WHERE
id='$id' and code='$code'",$this->link) or die(mysql_error());

if(mysql_num_rows($resultExist)>0){

  $row=mysql_fetch_assoc($resultExist);
  $password=$row["password"];
  return ($password);
  exit();
  //return ();

  }
  }
```

5.5. Debugging

Para realizar las pruebas en primera instancia se alojó el sistema en un servidor de pruebas proporcionado en el Departamento de Desarrollo Web, se creó una carpeta con el nombre uptodown que almacenaba todos los archivos generados para el nuevo sistema. Para acceder se usó el dominio www5.uqroo.mx/uptodown/files.php que vinculaba al sistema para realizar las pruebas de funcionamiento.

Esta es la etapa en la cual hacemos pruebas del sistema para la búsqueda y corrección de errores, por lo general en este tipo de sistemas web, es indispensable que el sistema funcione en diversas versiones de navegadores. Una de las pruebas aplicadas a nuestro sistema web fue utilizar Browsershots (http://browsershots.org) que es una aplicación o servicio web de código abierto que permite realizar capturas de pantallas de un desarrollo web en distintos navegadores y distintos sistemas operativos. Solo basta con agregar la dirección de la página que deseamos visualizar y seleccionar los navegadores.

Como observamos en la Ilustración 32, se presenta una lista extensa de navegadores que podemos seleccionar, en este caso se seleccionaron los navegadores con las versiones más actuales.

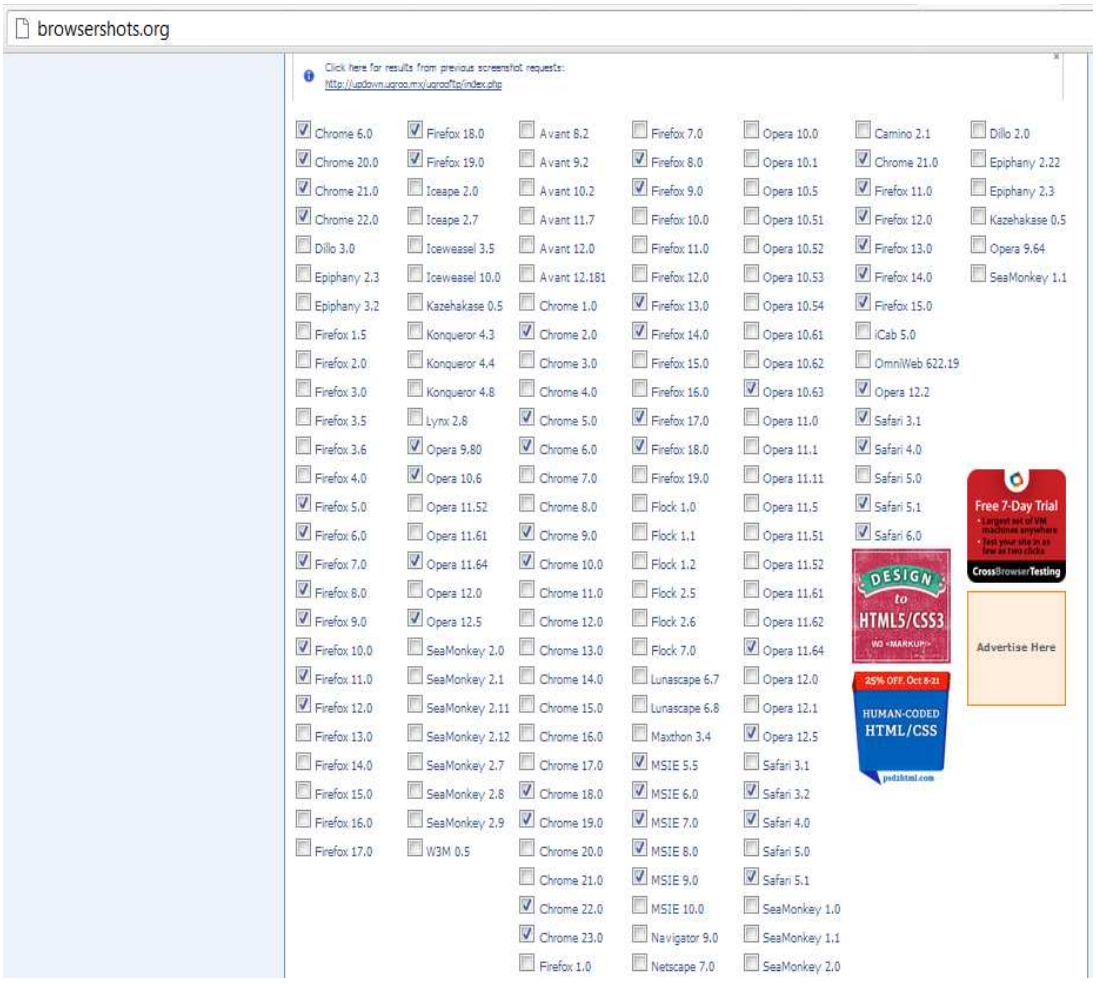

Ilustración 32.- Selección de navegadores.

Browsershots nos direcciona a las capturas de pantallas que aplico, como observamos en la Ilustración 33 en esta caso a 138 navegadores, entre los cuales, observamos que se cargaron 88 correctamente, uno que no pudo cargarse, y 49 en los que el tiempo de carga expiró.

En la misma ilustración observamos cómo se dividieron las 88 capturas por plataforma, 42 capturas fueron de Windows, 34 de Linux, 8 MAC OS y 4 de BSD.

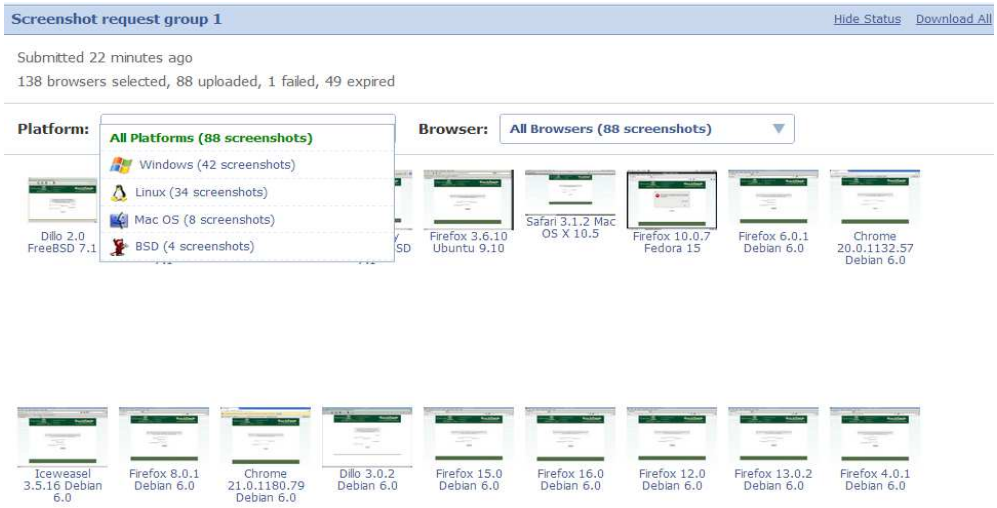

Ilustración 33.- Capturas de pantallas por plataforma.

En la Ilustración 34 observamos el número de capturas que se realizaron en cada navegador, para Firefox 37, Chrome 10, SeaMonkey 9, Opera 9 e Internet Explorer 5 capturas de pantalla.

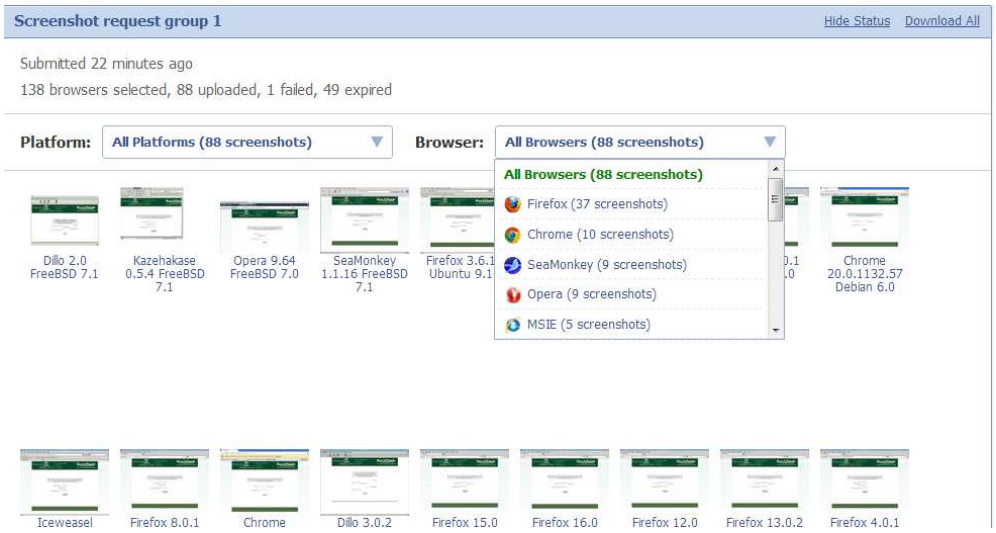

Ilustración 34.- Captura de pantallas por Navegadores.

Con estas capturas definimos algunos aspectos de diseño de nuestro sistema para mejorar la interacción con el usuario considerando a: Internet Explorer, Mozilla Firefox, Opera, Google Chrome y Safari como los principales navegadores para realizar las pruebas.

Algunas de las características y criterios de nuestro sistema que analizamos en los navegadores son:

a) Anticipación

b) Autonomía

c) Los colores y legibilidad

d) Eficiencia

e) Navegabilidad

f) Aprendizaje

g) Interfaz visible

h) sobrecarga de información

i) Diseño

Uno de los beneficios encontrados en los navegadores Internet Explorer, Mozilla Firefox y Google Chrome es el uso de herramientas de desarrollo para páginas web.

Internet Explorer

Las Herramientas de desarrollo se incluyen en todas las instalaciones de Internet Explorer 8. Esta herramienta permite a los desarrolladores de sitios web depurar rápidamente Microsoft JScript; investigar comportamientos específicos de Internet Explorer; o agilizar la realización de iteraciones a fin de crear prototipos de un nuevo diseño o probar soluciones para un problema sobre la marcha. Las herramientas proporcionan visibilidad del explorador, de modo que pueda inspeccionar las hojas de estilos en cascada (CSS) y el HTML del sitio tal

y como están presentes en Internet Explorer, no solo en el código fuente original. Esto resulta especialmente útil para los sitios dinámicos, los sitios complejos y los sitios que usan marcos de trabajo, como las páginas Active Server (ASP) o PHP.

Como se muestra en la Ilustración 35 se consideró las herramientas de desarrollo de Internet Explorer.

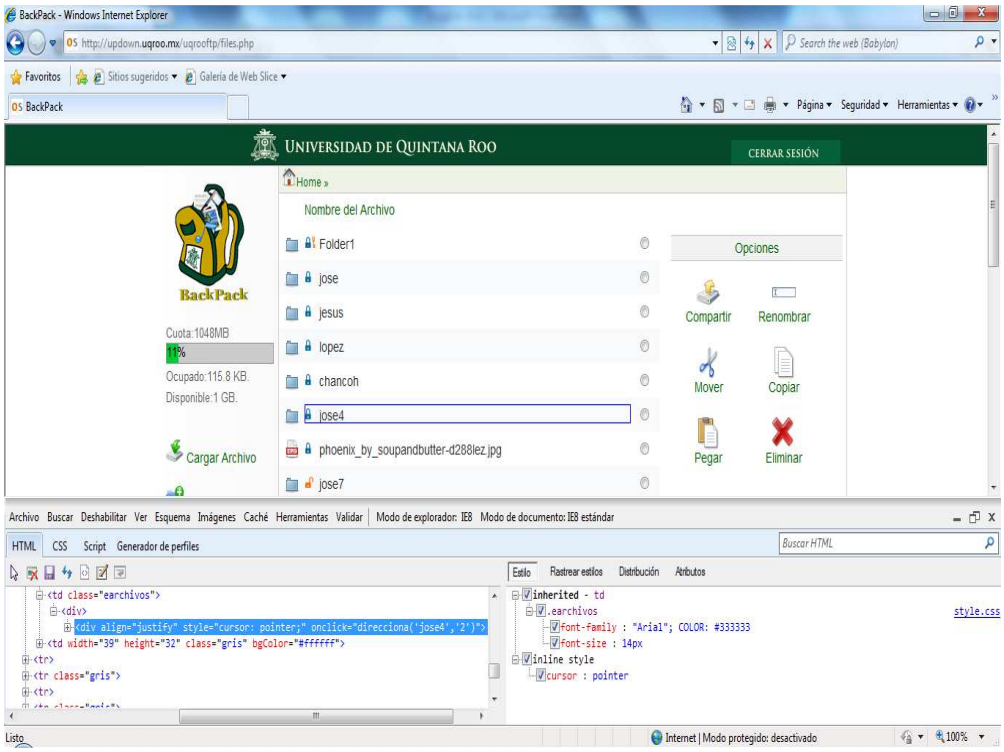

Ilustración 35.- Herramienta de desarrollo, Internet Explorer.

Google Chrome

Entre las funciones que ofrece Google Chrome se encuentra las *Herramientas del desarrollador* que permiten depurar javascript, inspeccionar el Modelo de Objetos del Documento (DOM) o analizar el tiempo de ejecución de cada función para optimizar el rendimiento.

Como se observa en la Ilustración 36, para inspeccionar el contenido de la página web la aplicación está divida en otros apartados: *Elements* que muestra los elementos HTML en forma organizada; *Timeline* que muestra el tiempo de operaciones que realiza el navegador y la cantidad de memoria que consumen estas operaciones; *Profiles* que permite hacer un seguimiento de las funciones de JavaScript y *Console* que es una consola JavaScript.

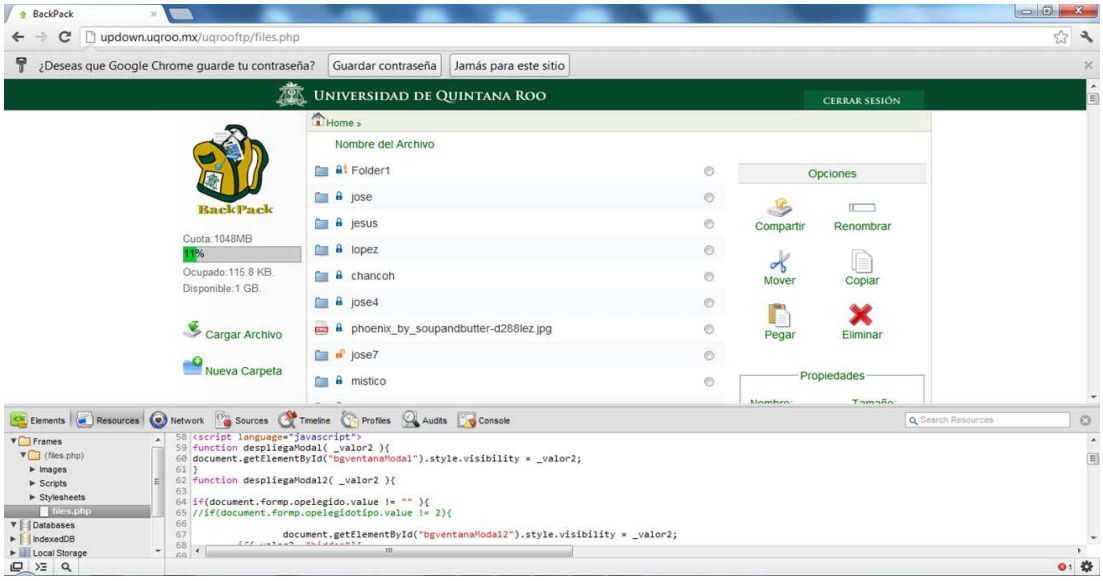

Ilustración 36.- Herramientas del desarollador, Google Chrome

Mozilla Firefox

Firebug es una conocida extensión para Firefox que permite editar, depurar y monitorear el código HTML, CSS y Javascript de páginas webs en tiempo real.

En la Ilustración 37 observamos que tanto el diseño, los colores, los tamaños, los márgenes, los elementos HTML, los tipos de letra, fondos, etc. son editables usando esta extensión.

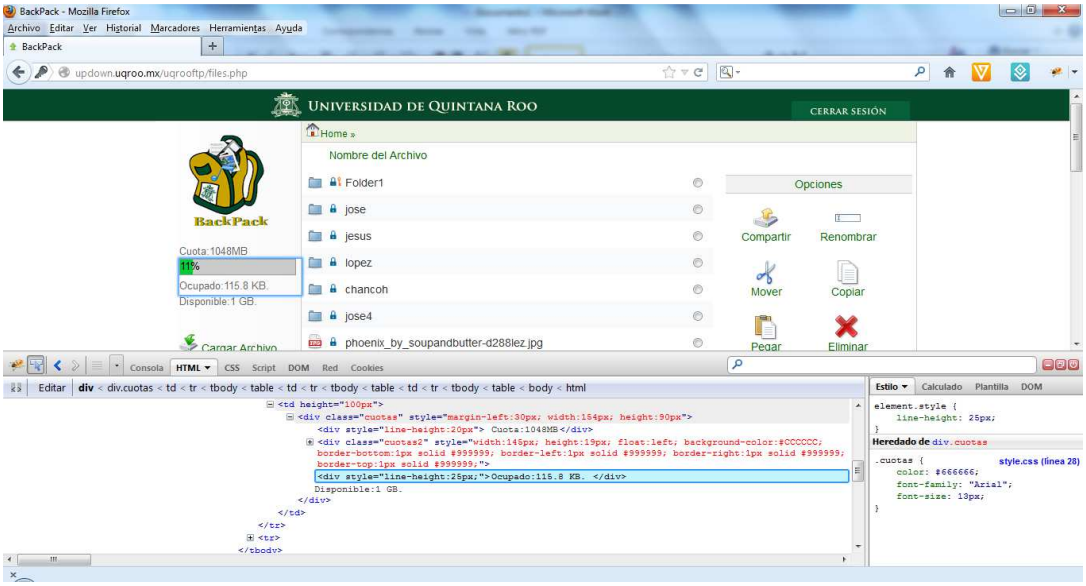

Ilustración 37.- Firebug, Mozilla Firefox.

En la Tabla 2 hacemos una comparación sobre la funcionalidad entre los navegadores que seleccionamos.

Tabla 2.- Comparación entre navegadores.

	Internet Explorer	Mozilla Firefox	Opera	Google Chrome	Safari
Autenticación	X	X	X	X	X
Conexión a la base de datos	X	X	X	X	X
Carga de archivos	X	X	X	X	X
Creación de carpetas	X	X	X	X	X
Visualización de contenido	X	X	X	X	X
Navegación entre carpetas	X	X	X	X	X
Eliminación de archivos y carpetas	X	X	X	X	X

Modificación de nombres	X	X	X	X	X
Copiar, mover y pegar	X	X	X	X	X
Visualización de barra de estado	X	X	X	X	X
Visualización de propiedades	X	X	X	X	X

Por ser un sistema desarrollado iterativamente en cascada, después de encontrar cada error se buscaba la corrección al instante. Las pruebas se hacían una vez terminadas las correcciones.

En la tabla anterior observamos que cada interfaz desarrollada funcionó correctamente, los errores más comunes encontrados fueron de sintaxis o lógicos, los cuales se corrigieron durante la programación.

CAPITULO VI

CONCLUSIONES

En la primera fase del proyecto, cuando se establecieron los alcances del mismo, se obtuvo un primer acercamiento sobre la relevancia y complejidad que conlleva el diseño de la interfaz de usuario en un sistema de recuperación de información, en particular cuando se habla de grandes volúmenes de información.

Cada una de las etapas de este proyecto de tesis se enfocó al desarrollo de un prototipo de un sistema informático de almacenamiento e intercambio de archivos, por lo que el resultado presentado permitirá la actualización del mismo. Es muy probable que surjan nuevas modificaciones que podrían añadirse a la estructura del sistema, como la sincronización con el correo electrónico.

Es fundamental que cada aspecto de este sistema sea mejorado para el beneficio de nuestra institución y a través del uso constante se fomente el intercambio y uso responsable de la información.

Durante el desarrollo del sistema fue necesario aprender muchos aspectos importantes para mejorar la versión anterior, comenzar con un proceso desde cero que mejore la experiencia de cualquier usuario y la posibilidad de abarcar un nuevo conocimiento en el área del desarrollo web. Es importante mencionar que cada fase de este proyecto fue supervisado con base a la experiencia y el conocimiento adquirido durante el desarrollo.

En función a las características que debían contemplarse durante el desarrollo se modificó más del 90 % el sistema anterior, se cumplieron cada uno de los requisitos de la planeación, sin embargo durante el proceso llegaron nuevas ideas, las cuales fueron agregadas para mejorar la experiencia del usuario.

Muchos de los compromisos asumidos para el desarrollo del sistema fueron transformados en nuevas ideas que permitieron adquirir nuevos conocimientos.

Es útil que a través de este trabajo de tesis se difundan las fases que se llevaron a cabo para el desarrollo del sistema web. También es importante

caracterizar este sistema para apoyar la producción de información institucional.

Quizás uno de los aspectos que pueda mejorar la experiencia del usuario es aumentar la capacidad de almacenamiento, de la cual se desprende la idea de abarcar mayor cantidad de usuarios. Aunque habría que mejorar la velocidad de respuesta del sistema.

El beneficio más importante es acceder dentro la intranet de la universidad de manera más confiable y rápida a la información. Permitir la accesibilidad a través de los navegadores actuales y aprovechar la cuenta de correo institucional para usar el sistema de almacenamiento e intercambio de archivos.

BIBLIOGRAFÍA

California Digital Library. (19 de 07 de 2000). Recuperado el 03 de 07 de 2011, de http://www.cdlib.org/cdlinfo/2000/07/19/july-2000-release-of-the-cdl/

MARC 21 Format for Bibliographic Data. (10 de 2001). Recuperado el 17 de 07 de 2011, de http://www.loc.gov/marc/bibliographic/

Consejo Latinoamericano de Ciencias Sociales. (2010). Recuperado el 02 de 07 de 2011, de http://www.clacso.org.ar/inicio/inicio.php

European Cultural Heritage Online: Open Access Infrastructure for a Future Web of Culture and Science. (2011). Recuperado el 19 de 06 de 2011, de http://echo.mpiwg-berlin.mpg.de/home

Cental, B. (2011). *BioMed Central: The Open Access Publisher.* Recuperado el 03 de 07 de 2011, de http://www.biomedcentral.com

Dante Cantone. (2006). *Implementacion y Debbugging.* Argentina: ZIGZAG.

Dspace. (2011). *DSpace open source software is a turnkey institutional repository application.* Recuperado el 19 de 07 de 2011, de http://www.dspace.org/

Dunsire, G. (2005). *Harvesting Institutional Resources in Scontland Testbed.* Recuperado el 03 de 07 de 2011, de http://hairst.cdlr.strath.ac.uk/

Epints. (2011). *Eprints.* Recuperado el 20 de 07 de 2011, de http://www.eprints.org/

FEDORA. (2009). *FedoraCommons.* Recuperado el 19 de 07 de 2011, de http://www.fedora-commons.org/about

Greentone. (2009). *Greenstone Digital Library Software.* Recuperado el 18 de 07 de 2011, de http://www.greenstone.org

Hagemann, M. (1 de 12 de 2001). *Budapest Open Acces Initiative.* Recuperado el 10 de 06 de 2011, de http://www.soros.org/openaccess/

Hanard, S. (21 de 04 de 2001). *The self-archiving initiative.* Recuperado el 26 de 05 de 2011, de http://users.ecs.soton.ac.uk/harnad/Tp/nature4.htm

Heery, R., & Anderson, S. (19 de 02 de 2005). *Digital Repositories Review UKOLN, University of Bath Arts and Humanities Data Service.* Recuperado el 21 de 05 de 2011, de http://www.jisc.ac.uk/uploaded_documents/digital-repositories-review-2005.pdf

Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, P. (2008). *Metodología de la Investigación.* Mexico: MCGRAW-HILL.

Invenia. (2002). *Declaración de Berlín sobre acceso abierto (open access).* Recuperado el 19 de 06 de 2011, de http://www.invenia.es/kb:berlin

INVENIO. (02 de 2011). *INVENIO.* Recuperado el 17 de 07 de 2011, de http://invenio-software.org/

Jhonson, R. (11 de 2002). *Institutional Repositories: Partnering with Faculty to Enhance Scholarly Communication.* Recuperado el 19 de 06 de 2011, de D-Lib Magazine: http://www.dlib.org/dlib/november02/johnson/11johnson.html

López Guzmán, Arriaga Aredondo, C., Castro Thompson, A., Galina Russell, A., Gamboa Rodríguez, I., Giménez Heau, F., y otros. (07 de 2006). *E-prints in Library and Information Sciense.* Recuperado el 21 de 05 de 2011, de http://eprints.rclis.org/handle/10760/12757#.Tt57hbIUqso

ROAR. (07 de 12 de 2011). *Registry of Open Access Repositories*. Recuperado el 11 de 12 de 2011, de http://roar.eprints.org/view/type/institutional.html

UNAM. (2009). *Universidad Nacional Autónoma de México.* Recuperado el 20 de 07 de 2011, de Acervos Digitales de la UNAM: http://www.rad.unam.mx/proyecto/index.php/antecedentes

Universidad de Quinatana Roo. (11 de 11 de 2011). *Universida de Quintana Roo*. Recuperado el 9 de 10 de 2011, de http://www.uqroo.mx

Van del Kuil, A., & Feijen, M. (10 de 2004). *The Dawning of the Dutch Network of Digital Academic REpositories (DARE): A Shared Experience.* Recuperado el 02 de 07 de 2011, de http://www.ariadne.ac.uk/issue41/vanderkuil/