Universidad de las Ciencias informáticas

Facultad 10

Título: Módulo de Interconexión de Archivos Históricos

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor:

Lester Peña López

Lemay Blanco González

Tutores:

Ing. Eliurkis Díaz Terrero

Ing. Reynier Pernia Rodríguez

Ciudad de La Habana, Junio 2008

...Lo fundamental es que seamos capaces de hacer cada día algo que perfeccione lo que hicimos el día anterior.

Che.

Declaración de autoría

Por este medio declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmamos la presente a los __ días del mes de __ del 2008.

Autores

Lester Peña López	Lemay Blanco González

Tutores

Ing. Eliurkis Díaz Terrero	Ing. Reynier Pernia Rodríguez

Agradecimientos

Ante todo deseamos expresar nuestro más sincero agradecimiento a la Revolución y en especial a nuestro Comandante en Jefe, que nos ha dado la posibilidad de realizar nuestros sueños.

A nuestros tutores que nos guiaron en la realización de este trabajo, a Jorge Ariel Salomón y Walfrido Serrano por brindarnos su ayuda y colaboración.

A los amigos del aula que siempre nos apoyaron en todo en estos años de estudio, Serguey, Darien, Leslier, Teijón, Ledian, el Luisi, Daniel, Radel, Yuniesky y el Leo.

A nuestros amigos que crecieron junto a nosotros y estuvieron siempre al tanto.

A nuestros familiares que han sido siempre nuestra guía y ejemplo.

A todos gracias.

Dedicatoria

Especial para mi mamá Irelis, mi papá Peña y mi hermano Leansy, que siempre me dan su cariño, alegría y confianza.

A mi compañera Leticia por ser mi consejera y sustento en estos años universitarios.

A todos mis compañeros de estudio, que siempre estuvimos en las buenas y en las malas, ya sea en el parque Lenin disfrutando o estudiando Física a la 1:00 am y sin corriente.

Lester.

A mi mamá Alina que me ha dedicado toda su vida, para ella es esta tesis, a mis abuelos Eva y Manolo, a mi tío Alexis, por darme su apoyo y cariño.

A mi novia Sahily por estar siempre a mi lado.

A Nana por haberme querido tanto, sé que estaría muy orgullosa de mí; a Ita, Nane y Jorgito, por ser mi familia.

A todos mis hermanos del aula por estos cinco años inolvidables de altos y bajos, a mis hermanos del PRE y del barrio, que siempre estuvieron al tanto y confiaron en mí.

Lemay.

Resumen

El presente trabajo tiene como objetivo principal, desarrollar un módulo Web que contenga un Servicio Web, que permita la interconexión de instituciones archivísticas para la búsqueda y recuperación de las descripciones de los archivos históricos, y que a su vez el sistema sea configurable para cualquier sistema gestor de base de datos que este utilizando el cliente. Esto ha sido necesario debido a la carencia de herramientas en nuestro país y a nivel internacional para el intercambio de información entre Archivos Históricos basados en la norma archivística ISAD (G).

Primeramente se realizó un estudio detallado de la norma ISAD (G) para la descripción de archivos históricos y su uso en la actualidad, así como del estándar de intercambio de información archivística EAD compatible con ISAD (G).

Se tuvo en cuenta la utilización de una metodología de desarrollo de software como guía para la realización del trabajo. Para la implementación del sistema se emplearon tecnologías muy usadas en la actualidad como PHP, Servicios Web, ADO DB, XML, JavaScript y Ajax.

Índice

Introducción

Con el surgimiento de Internet alrededor del año 1969 el mundo ha cambiado considerablemente, convirtiéndose en una sociedad en la que el acceso a la información es la primicia. Millones de archivos circulan por esta red y se rigen, al igual que los archivos físicos, por estándares internacionales desarrollados desde la década del 80 en varios países para normalizar su descripción.

La descripción archivística es el método que se utiliza para mostrar todas las especificaciones relacionadas con un documento de archivo. Desde finales del pasado siglo varios países desarrollaron su norma para la descripción de archivos, cada una de ellas diferente de acuerdo con sus necesidades particulares. Por ello, en el año 1993 y tras varios encuentros de los expertos se logra la elaboración de las directrices generales para la descripción de archivos; surge así la primera edición de la norma Internacional General de Descripción Archivística, en lo adelante ISAD (G), la cual determina cuáles son los datos descriptivos que puede contener un archivo y define su estructura jerárquicamente.

Esta norma constituye un significativo avance para la cooperación archivística internacional, debido a que todas las personas, al describir sus archivos de la misma forma, les resulta fácil comprender cada descripción. Pero todavía se necesitaría de un estándar para definir los formatos de edición y el modo de presentación de los elementos de manera uniforme, permitiendo su búsqueda e intercambio a través de internet.

Se desarrolla así en el año 1995, como parte de un proyecto de la Biblioteca de la Universidad de Berkeley en California, el Estándar de Descripción Archivística Codificada (EAD).

EAD permite tratar electrónicamente la información archivística, contribuye a la preservación de documentos históricos de posibles deterioros debido a la manipulación directa, y además posibilita que todas las personas en el mundo tengan acceso digital a estos.(M 2005)

Existen diversos mecanismos que posibilitan el intercambio de documentos de archivo a través de Internet. En la actualidad un método muy utilizado es el Servicio Web.

Un Servicio Web proporciona mecanismos de comunicación estándar entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Existen varios protocolos y entre los más utilizados se encuentran SOAP (Simple Object Access Protocol), RMI (Remote Method Invocation), DCOM (Distributed Component Object Model) y CORBA (Common Object Request Broker Architecture).

En la actualidad existen muchas aplicaciones relacionadas con el mundo de la archivística pero son

escasas las que permitan interconectar instituciones archivísticas para compartir las descripciones de los archivos históricos basadas en los estándares internacionales, y que además se adapten a cualquier SGBD (sistemas gestores de bases de datos), sean fáciles de configurar y de uso libre.

Dada esta situación se presenta el siguiente **problema científico:** ¿Cómo contribuir a mejorar la búsqueda y recuperación de descripciones de archivos históricos basados en los estándares internacionales?

Este trabajo tiene como **objeto de estudio** la gestión de archivos históricos y como **campo de acción** la búsqueda e intercambio de descripciones de archivos históricos.

De acuerdo con el problema planteado anteriormente se propone como **objetivo general**: Diseñar e implementar un módulo Web bajo herramientas libres y multiplataforma, que permita la búsqueda y recuperación de las descripciones de los archivos históricos basadas en estándares internacionales mediante la interconexión de instituciones archivísticas.

Para la realización de este trabajo se proponen las siguientes **tareas:**

1. Estudiar los estándares internacionales para la descripción de archivos históricos.
2. Investigar las técnicas y métodos para el intercambio de información entre sistemas remotos.
3. Realizar un estudio sobre los proyectos para la gestión de archivos históricos a nivel nacional e internacional (estado del arte).
4. Analizar y diseñar una herramienta que permita la búsqueda y recuperación de las descripciones de los archivos históricos basadas en estándares internacionales.
5. Implementar un módulo Web para sistemas de archivos históricos que permita la búsqueda y recuperación de las descripciones de los documentos de archivo basadas en estándares internacionales.

Para dar cumplimiento al objetivo general se plantea la siguiente **idea a defender**: El desarrollo de un módulo Web bajo herramientas libres y multiplataforma, brindará la posibilidad de conocer el material archivístico de varios archivos históricos interconectados, a través de la búsqueda y recuperación de las descripciones de los archivos.

Para la realización de estas tareas se combinarán diferentes **métodos de investigación,** teóricos y empíricos.

Entre los métodos teóricos se encuentran:

Histórico y lógico:

Se utiliza para estudiar cómo ha sido la evolución completa del fenómeno en su trayectoria y cómo ha ido aumentando su importancia para las diferentes áreas de trabajo a nivel nacional e internacional.

Hipotético deductivo:

Se utiliza para deducir a partir de la hipótesis y siguiendo reglas lógicas para obtener nuevos conocimientos.

Dentro de los métodos empíricos se encuentran:

Observación: Posibilita obtener conocimiento acerca del funcionamiento de los sistemas existentes en la actualidad para la gestión de archivos históricos; además permite conocer la realidad mediante la percepción directa de los objetos y fenómenos.

El presente trabajo consta de introducción, cuatro capítulos, conclusiones, recomendaciones, bibliografía, referencias bibliográficas.

En el Capítulo 1, **Capítulo 1. Fundamentación Teórica**, se abordan de forma general los aspectos teóricos más importantes que fundamentan la investigación. Además se justifica el uso de tecnologías para dar cumplimiento a los objetivos trazados en esta investigación, por último, se aborda la metodología de ingeniería del software seleccionada para el desarrollo del sistema.

En el Capítulo 2, **Característica del sistema**, se describen las funcionalidades del sistema. Se desarrolla el modelo de dominio, capturando los conceptos más importantes. Se determinan los requisitos y los casos de usos del sistema.

En el Capítulo 3, **Análisis y Diseño,** se describe el sistema a través de las fases del análisis y diseño que propone la metodología RUP para su desarrollo. Se muestran los diagramas de clases del análisis y de colaboración. Luego en la sección correspondiente al diseño se exponen los detalles relacionados con el diseño del sistema propuesto, utilizándose para su modelado los diagramas de clases de los casos de uso del sistema. Además se presenta el diagrama de clases persistentes y el modelo de datos.

En el Capítulo 4, **Implementación y Prueba,** se modelan los artefactos correspondientes al los flujos de trabajo implementación y prueba. Se muestran los diagramas de componentes y despliegue, así como un caso de prueba del sistema

Capítulo 1. Fundamentación Teórica

1.1 Introducción

En este capítulo se introducen varios conceptos imprescindibles para el desarrollo del módulo de interconexión de archivos; se precisa el concepto de archivística y documento de archivo, y se especifican los archivos históricos como tipo de documento a tratar.

Se describe la norma ISAD (G), así como el estándar EAD (Encoded Archival Description), fundamentales en el progreso del trabajo de diploma; se introducen los conceptos de servicios Web, XML (Extensible Markup Language) y SOAP (Simple Object Access Protocol) como tecnologías de intercambio de información remota, y se realiza una descripción de herramientas similares como RMI, CORBA (Common Object Request Broker Architecture) y DCOM (Distributed Component Object Model); además se argumenta la selección de el lenguaje de programación PHP (Hypertext Pre-processor) para la implementación del módulo Web, y se justifica el uso de las librerías genéricas ADO DB para la conexión a los SGDB.

1.2 Conceptos fundamentales y sus características

1.2.1 ¿Qué es archivística y documento archivístico?

La archivística es el estudio teórico y práctico de los principios, procedimientos y problemas concernientes al almacenamiento de documentos, buscando que dicha documentación se mantenga en el tiempo, y pueda ser consultada y clasificada.(Fuster)

Documento archivístico es toda expresión testimonial, en cualquier lenguaje, forma o soporte (forma oral o escrita, textual o gráfica, manuscrita o impresa, en lenguaje natural o codificado, en cualquier soporte documental, así como en cualquier otra expresión gráfica, sonora, en imagen o electrónica), generalmente en ejemplar único (aunque puede ser multicopiado o difundido en imprenta). (Fuster)

1.2.1.1 Archivo

Es uno o más conjuntos de documentos, independientemente de su fecha, su forma y soporte material, acumulados en un proceso natural por una persona o institución pública o privada en el transcurso de su gestión, y conservados para servir como testimonio e información a la persona o institución que los produce, o a los ciudadanos, o para ser utilizados como fuentes de historia.

Etapas del archivo:

- **Archivo de gestión**: bajo el control de la administración, también es conocido como archivo de oficina; debe permanecer en esta fase durante cinco años.
- **Archivo administrativo**: es una organización administrativa de gran volumen y complejidad; en cuanto las oficinas dejan de utilizarlo llegan a este archivo.
- **Archivo intermedio**: la documentación ha perdido prácticamente toda la utilidad que había tenido; durante otros 15 años se valora, se selecciona y se expurga.
- **Archivo histórico**: la documentación se conserva indefinidamente, debe ser factible de transmitir a las futuras generaciones.(Contreras)

Los archivos históricos constituyen archivos de valor que tienen como misión la de garantizar la conservación de aquellos documentos considerados de interés y la de difundir la información contenida en ellos, tanto a los propios productores de los documentos como a los investigadores y al público en general.

Es importante destacar que la utilización de este módulo en instituciones archivísticas solo es aplicable para los archivos históricos.

1.2.2 ¿Qué es ISAD (G)?

La Norma Internacional General de Descripción Archivística constituye una herramienta primordial para la gestión de los archivos. ISAD (G) permite elaborar descripciones archivísticas estándares que pueden tener tanto nivel de detalle como se desee especificar. Cuenta con cuatro principios fundamentales:

1. La descripción del archivo se origina de lo general a lo específico.
2. La información debe ser relevante para el nivel de descripción.
3. Las descripciones deben estar vinculadas entre niveles.
4. No repetición de la información.

Además se conforma por 26 elementos agrupados en siete áreas de descripción:

1. Área de identificación.
2. Área de contexto.
3. Área de contenido y estructura.
4. Área de condiciones de acceso y uso.
5. Área de documentación asociada.
6. Área de notas.
7. Área de control de la descripción.

De todos los elementos que conforman la norma, constituyen esenciales para el intercambio internacional de la información descriptiva:

- El código de referencia.
- El título.
- El productor.
- La fecha (s).
- La extensión de la unidad de descripción.
- El nivel de descripción.

1.2.3 ¿Qué es EAD?

EAD (Descripción Archivística Codificada) es una norma de estructura de datos que permite conservar la jerarquía y determinar el contenido de las directrices descriptivas para los fondos archivísticos mundiales, por medio de SGML (Standard Generalized Markup Language) y XML. Dichas directrices son permanentes y pueden distribuirse por Internet.

EAD (2002) en su versión actual es compatible con ISAD (G); emplea XML, un metalenguaje muy simple pero estricto, que desempeña un papel fundamental en el intercambio de una gran variedad de datos, una adaptación de SGML usado en la primera edición de EAD, pero limitado a su aplicación en internet.

Hay tres documentos técnicos asociados a EAD:

1. La DTD: EAD DTD (Document Type Definition), adaptable a SGML o XML, con material complementario como archivos de entidades opcionales.
2. La biblioteca de etiquetas: documento de referencia sobre cada uno de los elementos y atributos definidos en EAD, con apéndices útiles como pasarelas (crosswalks) a otros estándares (ISAD (G) 2000).
3. Las directrices de aplicación. (Flores)

1.2.3.1 Estructura

A un nivel muy básico, un documento "instrumento de descripción" codificado utilizando EAD, consta de tres segmentos: uno que proporciona información sobre el instrumento de descripción en sí mismo (su título, compilador, fecha de compilación), <eadheader>; un segundo componente que incluye las cuestiones preliminares necesarias para la publicación formal del instrumento de descripción, <frontmatter>; y un tercero que proporciona la descripción del material archivístico en sí misma, además de la información contextual y administrativa asociada, <findaid>.

1) Cabecera EAD (*EAD Header*) **<eadheader>.** Contiene información similar a la de la portada de un texto impreso, pero relativa al documento electrónico, que identifica la versión electrónica del instrumento de descripción y puede también documentar las prácticas de descripción y codificación seguidas en su creación.

2) Material Inicial (*Front Matter*) **<frontmatter>**, opcional. Puede usarse para codificar estructuras como prefacios, dedicatorias u otro texto relativo a la creación, publicación o uso del instrumento de descripción, y para generar una portada adaptada a las necesidades o gustos locales.

3) Descripción Archivística (*Archival Description*) **<archdesc>**. Elemento hijo de **<findaid>** contiene la descripción archivística propiamente dicha, y puede ser de nivel único (un único fondo, serie, expediente) o contener una descripción multinivel con la descripción de componentes subordinados (las series de un fondo, las unidades documentales de éstas, en dos o más niveles). (Society and Archivist)

Los principales componentes informativos que constituyen el modelo se pueden representar mediante una estructura arbórea que refleja claramente las relaciones jerárquicas entre los elementos:

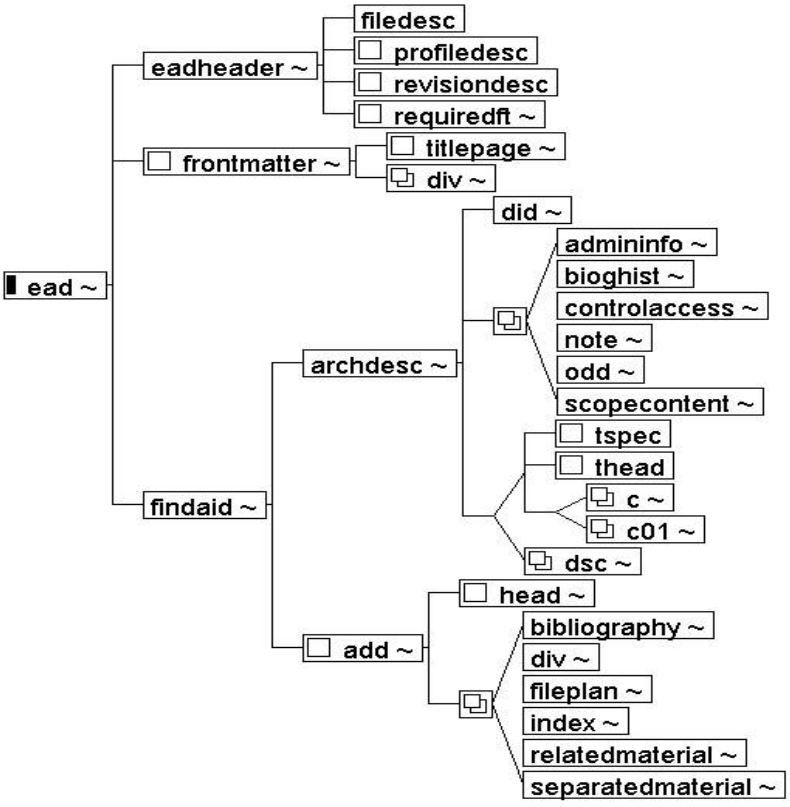

Figura 1.1 Estructura de un documento EAD.

1.2.3.2 ¿Por qué EAD?

Antes de que se desarrollara EAD muchos archivos utilizaban programas de interrelación de bases de datos internas, con el fin de crear instrumentos de descripción para colecciones complejas y obtener así una mayor capacidad de búsqueda y una indización y actualización más sencilla. Sin embargo, independientemente de lo útil que hayan resultado para los archivos individualmente, dichas bases de datos internas carecían de las estructuras de información normalizadas y de la plataforma de independencia que proporciona EAD y que resulta esencial para llevar a cabo una búsqueda entre los instrumentos de descripción de varios repositorios a la vez.

1.3 Lenguajes de programación Web

Desde los inicios de Internet fueron surgiendo diferentes demandas por los usuarios y se dieron soluciones mediante lenguajes estáticos. Con el transcurso del tiempo, las tecnologías fueron perfeccionándose y surgieron nuevos problemas para darles solución. Esto dio lugar a desarrollar lenguajes de programación dinámicos para la Web.(Valdés)

Las aplicaciones Web están compuestas fundamentalmente por una arquitectura cliente-servidor. El lado del cliente puede ser implementado por varios lenguajes de programación, entre los que se encuentran: HTML (HyperText Markup Language), Java Script, CSS (Cascading Style Sheets), que constituyen las tecnologías más usadas y son las encargadas de la representación visual de los componentes.

En el lado del servidor existen una gran cantidad de lenguajes, tales como ASP (Active Server Pages), Perl (Practical Extracting and Reporting Language) y PHP (Hypertext Preprocessor), este último seleccionado para realización del Módulo de Interconexión de Archivos Históricos.

1.3.1 ASP

ASP es un lenguaje script creado por la compañía Microsoft para la construcción de páginas Web dinámicas. No es de uso libre, ni multiplataforma, diseñado principalmente para entornos Windows. Su código no es abierto, lo que impide que muchos programadores del mundo hagan importantes aportes a su desarrollo; a pesar de ello posee buena documentación.

Si se utiliza un servidor web en entorno UNIX se debe comprar un producto desarrollado por la compañía Chilisoft, que permite interpretar el código ASP. (MOS)

1.3.2 Perl

Lenguaje Práctico para la Extracción e Informe, es de uso libre, creado por Larry Wall. Perl está creado a partir de lenguajes como C, sh, awk y sed, pero está diseñado para ser más práctico y fácil que estos últimos. Inicialmente fue implementado en entornos Unix, pero en la actualidad puede ser desarrollado en varios sistemas operativos.

No establece ninguna filosofía de programación concreta. No se puede decir que sea orientado a objetos, modular o estructurado, aunque soporta directamente todos estos paradigmas y su punto fuerte son las labores de procesamiento de textos y archivos.

Perl es un lenguaje de programación interpretado, al igual que muchos otros lenguajes de Internet como Java Script, ASP y PHP. Esto quiere decir que el código de los scripts en Perl no se compila, sino que cada vez que se quiere ejecutar se lee el código y se pone en marcha interpretando lo que hay escrito. Además es extensible a partir de otros lenguajes, debido a que desde Perl se pueden hacer llamadas a subprogramas escritos en otros lenguajes. También desde otros lenguajes se puede ejecutar código Perl.

1.3.3 PHP

PHP es un potente lenguaje de script multiplataforma y no propietario, soporta el paradigma de Programación Orientada a Objetos (POO). Fue creado en 1994 por Rasmus Lerdorf, diseñado originalmente para la creación de páginas Web dinámicas; hoy en día se pueden aprovechar sus grandes ventajas en la programación de escritorio.

Es un lenguaje de código abierto con abundante documentación que cada día gana más adeptos y millones de aplicaciones Web lo utilizan. La mayoría de su sintaxis es similar a C, Java y Perl, y es fácil de aprender. La meta de este lenguaje es permitir a los programadores desarrollar páginas dinámicas de una manera rápida y fácil, aunque no por ello deja de ser un lenguaje potente y seguro.

Para el desarrollo del módulo de interconexión de archivos históricos se ha elegido a PHP por sus potencialidades y las ventajas que presenta como software libre y gratuito.

1.4 Servidores Web

Un servidor web es un programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten usando el protocolo HTTP (Hypertext Transfer

Protocol) o el protocolo HTTPS (la versión cifrada y autenticada del protocolo HTTP). El servidor web va a ser fundamental en el desarrollo de las aplicaciones del lado del servidor, ya que además de contribuir a la seguridad de las aplicaciones Web, al mantener los ficheros en el lado del servidor, contribuye a que los clientes no necesiten de ninguna capacidad adicional para visualizar el contenido.

Entre los servidores Web más usados mundialmente se encuentran IIS (Internet Information Services) y Apache HTTP Server.

1.4.1 Apache HTTP Server

El servidor Apache HTTP es un servidor de código abierto y gratuito, desarrollado por la compañía Apache Software Foundation, y es el servidor Web más usado en Internet. Es utilizado en plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras. Goza de gran soporte técnico y documentación, debido a la gran comunidad que tiene a su disposición en foros, listas de correo, servidores de noticias, etc.

Apache es un software que está estructurado en módulos. La configuración de cada módulo se hace mediante la configuración de las directivas que están contenidas dentro del módulo. Los módulos del Apache se pueden clasificar en tres categorías:

1- Módulos Base: Módulos con las funciones básicas de Apache.

2- Módulos Multiproceso: Son los responsables de la unión con los puertos de la máquina, aceptando las peticiones y enviando a los hijos a atender las peticiones.

3- Módulos Adicionales: Cualquier otro módulo que le añada una funcionalidad al servidor.

El servidor Apache HTTP es altamente configurable en cualquier sistema operativo que se quiera utilizar, al igual que su instalación es muy sencilla.

1.4.2 IIS (Internet Information Services)

IIS es una serie de servicios para los ordenadores que funcionan con Windows. Originalmente era parte del Option Pack para Windows NT. Luego fue integrado en otros sistemas operativos de Microsoft destinados a ofrecer servicios, como Windows 2000 o Windows Server 2003, Windows XP Profesional incluye una versión limitada de IIS. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS. El ordenador que tenga instalado este servicio se convierte en un servidor de Internet,

debido a que puede publicar paginas web ya sea local o remotamente. Puede procesar distintos tipos de páginas debido a varios módulos que le dan esa funcionalidad, por ejemplo páginas ASP, ASP.NET, PHP o PERL.

1.5 Sistemas Gestores de Bases de Datos

En la actualidad existe un gran número de SGBD, todos con la principal función de preservar los datos. Entre los más usados están: Oracle, Informix, MySQL, PostgreSQL y Microsoft SQL Server. La elección de un tipo de gestor depende de muchos factores, como es el caso de los recursos materiales que se posea para adquirir una licencia, en el caso que esta sea propietaria; de la cantidad de datos a almacenar; y de la rapidez con la que se desea que se ejecute una consulta, ya sea para grandes cantidades de datos o para pocas, entre otras circunstancias especiales.

1.5.1 Acceso a los Sistemas Gestores de Bases de Datos

Muchas aplicaciones usan diferentes SGBD y la adhesión del módulo que se desea implementar conllevaría un problema si este se centrara solamente en un SGBD en específico; es por ello que se concibió la utilización de librerías genéricas que posibilitaran el acceso a un gran número de SGBD. Además, las funciones de acceso a los SGBD en PHP no están estandarizadas. Esto requiere una librería que elimine las diferencias entre cada API de base de datos.

ADOdb y PDO son un conjunto de librerías que proveen una capa de abstracción de acceso a datos, cada una proporciona los mecanismos necesarios para la comunicación con un gran número de SGBD.

1.5.1.1 ¿Por qué ADOdb?

ADOdb permite la conexión con muchas más bases de datos que PDO; aunque es más lento que este último se puede modificar y mejorar su funcionamiento. ADOdb es una clase grande y de alto rendimiento, debido a su diseño en forma de capas, con las funciones más rápidas en la capa más profunda. Usa principalmente estas funciones para mejorar su rendimiento:

Capa mas profunda
Connect, PConnect, NConnect Execute, CacheExecute SelectLimit, CacheSelectLimit MoveNext, Close qstr, Affected_Rows, Insert_ID

Figura 1.2 Funciones de rendimiento

1.6 Servicios Web

Los servicios Web son una colección de protocolos y estándares diseñados para soportar la interoperabilidad entre aplicaciones a través de una red, es decir, distintos softwares desarrollados en diferentes lenguajes de programación y ejecutados sobre cualquier plataforma, pueden utilizar los servicios Web para intercambiar datos en redes de ordenadores como Internet. Poseen una interfaz descrita en un formato que puede ser procesado por una máquina como WSDL (Web Services Description Languages), la cual es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios Web, o sea define los requisitos del protocolo.

1. 6.1 SOAP

1.6.1.1 ¿Qué es SOAP?

SOAP (Simple Object Access Protocol) no es más que un protocolo que se utiliza para la comunicación de aplicaciones mediante un formato de mensajes definido por él. Utiliza como lenguaje de codificación a XML, un protocolo de transporte para la comunicación. SOAP no rige el uso de un protocolo de transporte determinado, funciona sobre SMTP (Simple Mail Transfer Protocol) y FTP (File Transfer Protocol), entre otros, pero sí define cómo debe ser el transporte en el caso de HTTP. En fin, SOAP define un mecanismo de intercambio de información entre pares de aplicaciones en un entorno distribuido.

SOAP deriva de XML-RPC, protocolo creado por David Winer en 1998 que también utilizaba XML y HTTP como lenguaje de codificación y protocolo de transporte, respectivamente; con el transcurso del tiempo el proyecto iniciado por David Winer interesó a grandes compañías, como IBM y Microsoft, y a partir de ahí se desarrolló SOAP.

1.6.1.2 Mensajes SOAP

SOAP especifica el formato de los mensajes para la comunicación de las aplicaciones; por lo general estos mensajes son una forma de comunicación de única vía entre un emisor y un receptor (mensajes asíncronos), pero los mensajes pueden ser combinados implementando patrones de requerimiento/respuesta (mensaje sincrónico).

El mensaje SOAP es un documento XML que está constituido por un **envelope** (sobre), cuya estructura está compuesta por los siguientes elementos: **header** (cabecera) y **body** (cuerpo).

Figura 1.3 Estructura de un mensaje SOAP.

1.6.1.3 SOAP envelope

Envelope (envoltura): Es el elemento raíz del mensaje para describir su contenido y la forma de procesarlo, contiene el resto del documento XML, debe estar presente siempre y ser la primera sección del mensaje. Define los distintos NAMESPACES que son usados en el resto del mensaje.

```
<?xml version="1.0"?>

<soap:Envelope

xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
```

```
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
 ...
 Texto del mensaje
 ...
</soap:Envelope>
```

Figura 1.4 SOAP Envelope.

Los NAMESPACES se utilizan para garantizar la unicidad de los elementos y evitar ambigüedades.

1.6.1.4 SOAP Header

Header (encabezado): Es la información de identificación del contenido. Un grupo de reglas de codificación para expresar las instancias de tipos de datos definidos por la aplicación. Header es opcional y posee información específica sobre el mensaje SOAP (como la autenticación); es un mecanismo genérico para extender las características de los mensajes SOAP de una manera descentralizada y sin un acuerdo previo entre las partes que se comunican. En caso de estar presente, debe ser el primer hijo de la etiqueta envelope.

```
<SOAP-ENV:Header>
        <Información del usuario>
</SOAP-ENV:Header>
<SOAP-ENV:Header>
        <Información de transacción>
</SOAP-ENV:Header>
```

Figura 1.5 SOAP Header.

1.6.1.5 SOAP Body

Body (cuerpo): Es el contenido del mensaje. Una convención para representar las llamadas y las respuestas a procedimientos remotos, actúa como contenedor para la información que se envía al receptor del mensaje. Esta etiqueta debe aparecer siempre en los mensajes SOAP y ubicarse a continuación del header, o ser el primer hijo de envelope, si el header no está.

```
<SOAP-ENV:Body>
        <m1:NombredelMetodoRemoto xmlns:m1="URI">
                <arg1>value1</arg1>
                <arg2>value2</arg2>
        </m1:NombredelMetodoRemoto >
        <m2:NombredelMetodoRemoto xmlns:m2="URI">
      <arg1>resultado</arg1>
   </m2:NombredelMetodoRemoto >
</SOAP-ENV:Body>
```

Figura 1.6 SOAP Body.

1.6.1.6 ¿Por qué SOAP?

En la actualidad existen diferentes tecnologías de intercambio de información remota ampliamente difundidas y que funcionan correctamente, como Java RMI (Remote Method Invocation), CORBA (Common Object Request Broker Architecture) y DCOM (Distributed Component Object Model), por solo mencionar algunas, y cada una de ellas provee su propio protocolo de comunicación. Si se piensa en la evolución de los sistemas distribuidos hacia los Servicios Web, donde las aplicaciones distribuidas son accesibles a otros sistemas, puede verse que las distintas tecnologías de procesamiento distribuido existentes presentan una serie de limitaciones debido a:

Son dependientes de la plataforma y de fabricantes: DCOM es una tecnología propietaria de Microsoft, por lo que está ligado fuertemente al sistema operativo Windows y CORBA; a pesar de ser una arquitectura abierta, está controlado por determinados vendedores. Por lo tanto, hacer que los sistemas distribuidos, construidos usando DCOM o CORBA, sean accesibles a otros sistemas heterogéneos (escritos o funcionando en otra plataforma) es un problema bastante serio, a menos que

ambas partes tengan la misma plataforma y sistemas similares, lo cual implica que las aplicaciones tengan un alto acoplamiento.

Problemas con los firewalls: Tanto DCOM como CORBA asignan puertos en forma dinámica, lo cual dificulta su uso en Internet, donde los firewalls bloquean normalmente el acceso, excepto por puertos específicos.

Protocolos sofisticados: Implica la instalación de complejas librerías del lado del cliente; esto conlleva que el proceso de desarrollo de los clientes DCOM y CORBA sea complejo. Esto contradice a los servicios Web, cuya idea base es publicar servicios para que sean accesibles a otras aplicaciones. Además requieren de un gran soporte en tiempo de ejecución para funcionar correctamente.

Formatos propietarios de representación de los datos: DCOM utiliza un formato llamado NDR (Network Data Representation) y CORBA utiliza un esquema similar, pero incompatible, llamado CDR (Common Data Representation). Además dichos formatos son binarios y muy ligados a la arquitectura de los modelos de componentes respectivos.

En el caso de Java RMI, provee un simple mecanismo de aplicación distribuida, pero solamente se puede utilizar para comunicar aplicaciones codificadas en Java, o sea, no permite la interacción entre sistemas de otro lenguaje que no sea Java.

SOAP no es una tecnología creada para reemplazar las ya existentes, sino que permite la interoperabilidad de aplicaciones heterogéneas; incluso se puede decir que no es una nueva tecnología, sino la utilización de tecnologías existentes para internet con el fin de estandarizar la comunicación entre aplicaciones distribuidas a través de la Web. La utilización de SOAP provee de múltiples ventajas, tales como:

Manejo de los Firewall (Cortafuegos): Al utilizar generalmente HTTP como protocolo de transporte, posibilita pasar fácilmente a través de los firewalls, lo cuales dejan pasar habitualmente el tráfico que les llega por el puerto HTTP. La mayoría de las compañías bloquean casi todos los puertos excluyendo el HTTP, lo que no es un problema para el funcionamiento de SOAP.

Independiente de plataformas y lenguajes: Al estar fuertemente vinculado a HTTP y XML hace que SOAP sea completamente independiente de plataformas, sistemas operativos o lenguajes de programación.

No se encuentra fuertemente asociado a ningún protocolo de transporte: Un mensaje de SOAP no es más que un documento XML, por lo que puede transportarse utilizando cualquier protocolo capaz de transmitir texto.

1.6.2 XML

1.6.2.1 ¿Qué es XML?

XML (Extensible Markup Language) es un metalenguaje extensible de etiquetas desarrollado por el W3C (World Wide Web Consortium). Es un subconjunto simplificado del SGML que fue diseñado principalmente para la Web.Deja a los diseñadores crear sus propias etiquetas o tags, habilitando la definición, transmisión, validación y la interpretación de datos entre aplicaciones y entre organizaciones. XML tiene múltiples aplicaciones, entre las que se destaca su uso como estándar para el intercambio de datos entre diversos softwares con lenguajes privados, como es el caso de SOAP.

A partir de documentos XML se pueden programar buscadores, debido a que la información en dichos documentos está etiquetada por su significado de manera precisa, y es posible localizarla de forma mucho más clara que en documentos HTML.

1.7 Proyectos de gestión de archivos históricos

Los archivos históricos surgen a partir de la necesidad de conservar o guardar documentos que fueron realizados para dejar constancia de algún hecho en específico. Con el paso del tiempo estos documentos han variado su soporte material, desde el papiro hasta el papel, y han llegado a la era de los documentos microfotográficos.

Todos los países deben poseer archivos históricos, que no son más que instituciones o partes estructurales de ellas que realizan la recepción, conservación y organización de los documentos para su utilización, o sea, un depósito o local en el cual existen ciertas cantidades de archivos públicos y privados, algo similar a una biblioteca. En la actualidad, con las nuevas tecnologías informáticas, la gestión de archivística se ha hecho más flexible, debido a que la mayoría de estos ya tiene informatizada dicha gestión.

A nivel internacional son diversos los proyectos que se han llevado a cabo para lograr una mayor eficiencia y calidad en la gestión de archivos históricos. España posee gran protagonismo en el desarrollo de la archivística en Europa y a nivel mundial, debido a que su desarrollo en esta rama es muy grande. Entre sus productos más destacados se encuentra:

1.7.1 PARES

Portal de archivos españoles, es un proyecto del Ministerio de Cultura español destinado a la difusión en internet del patrimonio histórico documental. PARES ofrece un acceso libre y gratuito para todos los usuarios interesados en acceder a los documentos con imágenes digitalizadas. Permite realizar búsquedas sencillas y avanzadas de los archivos históricos a través de una interfaz amigable y simple. Admite además la interconexión con otros archivos. Es un sitio bien diseñado pero resulta imposible reutilizar su código, ya que este es cerrado.

1.7.2 ARNAC

En Cuba existen diversos archivos históricos, como el Archivo Nacional de la República de Cuba (**ARNAC**) y algunos de las provincias que conforman la división político-administrativa del país. Estos archivos no se encuentran comunicados o interconectados unos con los otros para la búsqueda de descripciones de archivos históricos.

1.7.3 PAPIRO

En la provincia Granma se desarrolló un producto informático basado en el estándar ISAD (G) denominado PAPIRO, el cual es de uso libre y realizado con la finalidad de conservar la documentación de valor histórico al evitar su manipulación directa. Está concebido solamente para funcionar en Windows en la versión 2000 o XP y tampoco posee la funcionalidad de búsqueda a través de varios archivos interconectados. (Delio G. Orozco González)

1.8 Metodología de desarrollo del software

La evolución de la industria del software y el gran desarrollo que han alcanzado las computadoras, han permitido que se construyan software más potentes y complejos. En el desarrollo del software cada día hay una mayor exigencia de calidad, funcionabilidad, con el menor costo y en el menor tiempo posible.

Todo desarrollo de software es riesgoso y difícil de controlar, pero si no se sigue una metodología, lo que se obtiene es clientes insatisfechos con el resultado y desarrolladores aún más insatisfechos.

Para guiar dicho proceso existen diferentes metodologías, todas enfocadas a mejorarlo; algunas de las más usadas mundialmente son:

- eXtreme Programing (XP)

- Rational Unified Process (RUP)

1.8.2 eXtreme Programing (XP)

XP es una metodología ágil de desarrollo de software, cuya particularidad es tener como parte del equipo al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. XP está diseñada para el desarrollo de proyectos cortos y que requieran de un grupo de programadores pequeños; se basa en una serie de metodologías de desarrollo de software en las que se da prioridad a los trabajos que aportan un resultado directo y que reducen la burocracia que hay en torno a la programación.

1.8.2.1 Fases de la metodología XP

Planificación:

XP plantea la planificación como un permanente diálogo entre las partes: la empresarial (deseable) y la técnica (posible). Las personas del negocio necesitan determinar:

- Inicialmente se definen las historias de usuario, las que tienen similitud con casos de uso. Estas en particular presentan dos o tres líneas con terminología del cliente, sin hacer mucho hincapié en los detalles técnicos.
- Después de tener las historias de usuario es necesario crear un plan de publicaciones, donde se indiquen las historias de usuario que se crearán para cada versión del programa, el tiempo que tardarán en desarrollarse y publicarse las versiones del programa, el número de personas que trabajarán en el desarrollo y cómo se evaluará la calidad del trabajo realizado.
- El desarrollo se divide en iteraciones, cada una de las cuales comienza con un plan de iteración para el que se eligen las historias de usuario a desarrollar y las tareas de desarrollo.
- En la planificación se estima la velocidad del proyecto, la cual da una medida del número de historias que se pueden desarrollar en las distintas iteraciones. La metodología XP aconseja la programación en parejas, pues incrementa la productividad y la calidad del software desarrollado.

Diseño:

- Se eligen los diseños más simples que funcionen.

- Usar glosarios de términos y una correcta especificación de los nombres de métodos y clases, facilitará sus posteriores ampliaciones y la reusabilidad del código.
- Nunca se debe añadir funcionalidad extra al programa aunque se piense que en un futuro será utilizada.
- Refactorizar: Es mejorar y modificar la estructura y codificación de códigos ya creados sin alterar su funcionalidad.
- El uso de las tarjetas C.R.C (Class, Responsabilities and Collaboration) permite al programador centrarse y apreciar el desarrollo orientado a objetos, olvidándose de los malos hábitos de la programación procedural clásica.

Codificación:

- En todas las fases de XP es de vital importancia la presencia del cliente; a la hora de codificar una historia de usuario su presencia es aún más necesaria. La codificación debe hacerse ateniendo a estándares de codificación ya creados. Programar bajo estándares mantiene el código consistente y facilita su comprensión y escalabilidad.
- Crear pruebas para analizar el funcionamiento de los distintos códigos implementados permitirá un buen desarrollo de dicho código. Se puede dividir la funcionalidad que debe cumplir una tarea a programar en pequeñas unidades, haciendo las pruebas más sencillas y manejables.
- XP sugiere un modelo de trabajo usando repositorios de código donde las parejas de programadores publican cada pocas horas sus códigos implementados y corregidos.
- Se usa la propiedad colectiva del código, lo que se traduce en que cualquier programador puede cambiar cualquier parte del código. El objetivo es fomentar la contribución de ideas por parte de todo el equipo de desarrollo.
- Se deja la optimización para el final.
- No se hacen horas extra de trabajo.

Pruebas:

Uno de los pilares de la metodología XP es el uso de pruebas para comprobar el funcionamiento de los códigos que se vayan implementando.

- Hay que crear las pruebas abstrayéndose del futuro código, de esta forma se asegura la independencia de las pruebas respecto al código que se evalúa.
- Ningún código puede ser publicado en el repositorio sin que haya pasado su prueba de funcionamiento; de esta forma se asegura el uso colectivo del código.
- Las pruebas permiten verificar que un cambio en la estructura de un código no tiene porqué cambiar su funcionamiento.

1.8.1 Rational Unified Process (RUP)

RUP es un proceso para el desarrollo de un proyecto de software que define quién, cómo, cuándo y qué debe hacer el proyecto. El Proceso Unificado propone un modelado visual de la estructura y los componentes, para ello utiliza UML (Lenguaje Unificado de Modelado).

1.8.1.2 Características fundamentales del RUP

Dirigido por casos de uso:

Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo, ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso (cómo se llevan a cabo).

Centrado en la arquitectura:

La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción.

Iterativo e Incremental:

RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros. Es práctico dividir el trabajo en partes más pequeñas o miniproyectos. Cada miniproyecto es una iteración que resulta en un

incremento. Las iteraciones hacen referencia a pasos en los flujos de trabajo, y los incrementos, al crecimiento del producto.

Ciclo de vida del RUP:

La vida de un sistema transcurre a través de ciclos de desarrollo desde su nacimiento hasta su muerte, en cada ciclo se repite el proceso unificado de desarrollo. Cada ciclo consta de cuatro fases:

Inicio: Se hace un plan de fases, se identifican los principales casos de usos y se identifican los riesgos. Se define el alcance el proyecto.

Elaboración: Se define la arquitectura del sistema y se obtiene una aplicación ejecutable que responde a los casos de uso que la comprometen.

Construcción: Se obtiene un producto listo para su utilización, el cual está documentado y tiene un manual de usuario. Se obtiene uno o varios entregables del producto que han pasado las pruebas.

Transición: El entregable ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

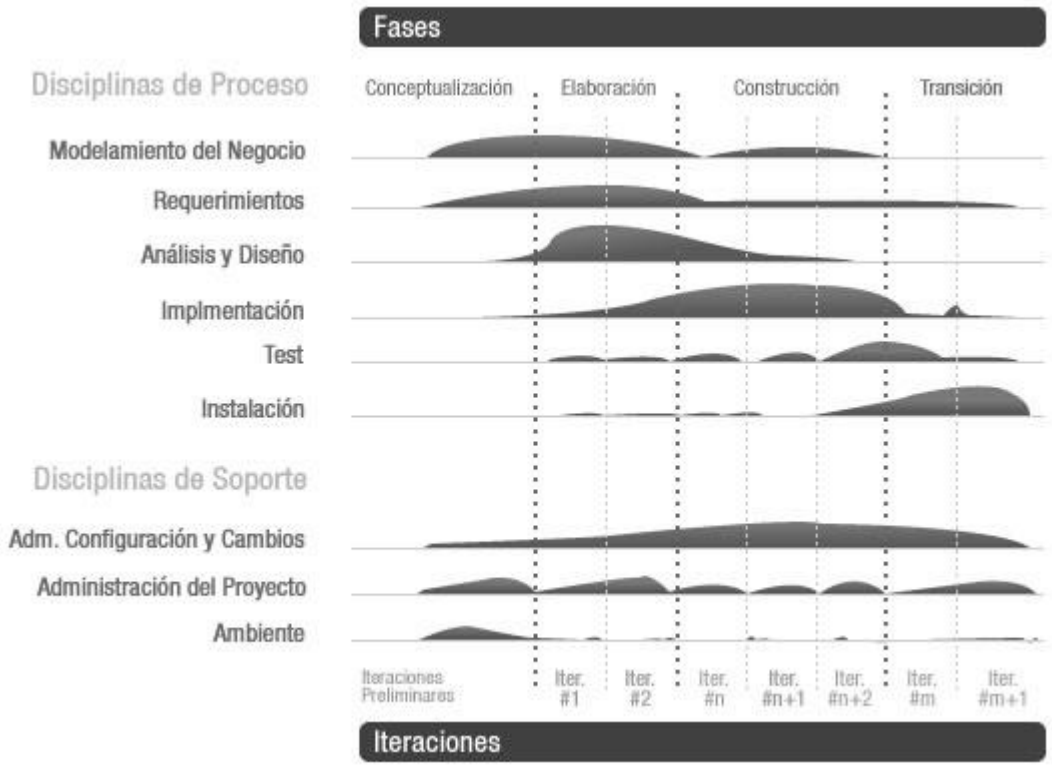

Figura 1.7 Características de RUP.

1.8.2.2 ¿Por qué la metodología RUP?

Se decidió utilizar la metodología RUP para guiar el proceso de desarrollo del módulo de interconexión de archivos históricos, debido a que está bien documentada y se ha probado su éxito en disímiles proyectos de desarrollo de software. Además, por las características específicas del presente trabajo dicha metodología facilita mucho su desarrollo debido a que presta especial atención al establecimiento temprano de una buena arquitectura que no se vea fuertemente impactada a cambios posteriores durante la construcción y el mantenimiento.

1.8.2.2 Lenguaje de modelado UML

Como lenguaje de modelado se utilizará UML que es un lenguaje que permite modelar sistemas de software mediante el paradigma orientado a objetos; así como visualizar, especificar, construir y documentar un sistema de software. UML está estrechamente relacionado a RUP y se ha convertido en un estándar a nivel mundial al poseer las siguientes características:

- Permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- Cubre las cuestiones relacionadas con los tamaños propios de los sistemas complejos y críticos.
- Es un lenguaje muy expresivo que cubre todas las vistas necesarias para desarrollar y luego desplegar los sistemas.
- Existe un equilibrio entre expresividad y simplicidad, pues no es difícil de aprender ni de utilizar.
- Es independiente del proceso, aunque para utilizarlo óptimamente se debería usar en un proceso que fuese dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental.

1.8.2.3 Herramienta CASE de modelado de UML

Las herramientas CASE (Computer Aided Software Engineering) suponen la automatización del desarrollo del software, contribuyendo a mejorar la calidad y la productividad en el desarrollo de sistemas de información y se plantean los siguientes objetivos:

- Permitir la aplicación práctica de metodologías estructuradas, las cuales al ser realizadas con una herramienta se consigue agilizar el trabajo.
- Facilitar la realización de prototipos y el desarrollo conjunto de aplicaciones.
- Simplificar el mantenimiento de los programas.
- Mejorar y estandarizar la documentación.
- Aumentar la portabilidad de las aplicaciones.
- Facilitar la reutilización de componentes software.
- Permitir un desarrollo y un refinamiento visual de las aplicaciones, mediante la utilización de gráficos.

Automatizar:

- El desarrollo del software
- La documentación
- La generación del código
- El chequeo de errores
- La gestión del proyecto

Permitir:

- La reutilización del software
- La portabilidad del software
- La estandarización de la documentación

1.8.2.3.1 Rational Rose

Rational Rose es una herramienta propietaria para el modelado visual, que forma parte de un conjunto más amplio de herramientas que juntas cubren todo el ciclo de vida de desarrollo de software. Esta herramienta propone la utilización de cuatro tipos de modelo para realizar un diseño del sistema, utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software.

Características de Rational Rose:

• Mantiene la consistencia de los modelos del sistema software.

• Chequeo de la sintaxis UML.

• Genera la documentación automáticamente.

• Generación de código a partir de los modelos.

• Ingeniería inversa (crear modelos a partir de código).

1.8.2.3.2 Visual Paradigm

Visual Paradigm es una herramienta CASE profesional que soporta el ciclo de vida completo del desarrollo de software y ayuda a optimizar la construcción de las aplicaciones; además permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

Características de Visual Paradigm:

- Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Los modelos y el código permanecen sincronizado en todo el ciclo de desarrollo.

- Disponibilidad de integrarse en los principales IDE.
- Disponibilidad en múltiples plataformas.

1.9 Conclusiones del capítulo

En el presente capítulo se expusieron conceptos y características imprescindibles para la comprensión de este trabajo de diploma. Se decidió la adopción de diferentes tecnologías en dependencia de sus ventajas, de acuerdo con otras existentes. Se propuso, además, una metodología de desarrollo de software para la realización eficiente del trabajo.

Capítulo 2. Características del sistema

2.1 Introducción.

En este capítulo se abordará brevemente el flujo actual de los procesos involucrados en el campo de acción del problema planteado y se hará un análisis crítico de como se ejecutan actualmente estos procesos.

Se describirán los pasos realizados mediante la metodología propuesta en el capítulo anterior (RUP) para la construcción del sistema. Se realizará un Modelo de Dominio para ayudar a la compresión del contexto del sistema, capturando los tipos más importantes de objetos. También será elaborada una lista de requerimientos no funcionales y funcionales que deberá tener el módulo de interconexión de archivos, y a partir de estos últimos se conformarán los casos de uso del sistema, de los cuales se ofrecerá una descripción textual, se definirán los actores del sistema y las relaciones entre ellos y se elaborarán los diagramas de casos de uso del sistema.

Además se detallará la solución propuesta para la realización del módulo de interconexión de archivos históricos y se expondrán algunos elementos relacionados con los prototipos de interfaz de usuario.

2.2 Modelo de Dominio

El desarrollo de un sistema software por pequeño que sea, necesita dividirse en piezas más pequeñas que permitan su comprensión. Estas piezas se pueden representar a través de modelos que facilitan su interpretación. Existen dos modelos que ayudan a una interpretación inicial del software que se quiere desarrollar: modelo de dominio y modelo de negocio.

El modelo del dominio captura los tipos más importantes de objetos que existen en el entorno donde estará el sistema y describe los conceptos más importantes del módulo, debido a que los procesos del negocio no son visibles. Los objetos del dominio representan "cosas" que existen o los eventos que suceden en el entorno en el que trabaja el sistema. Este modelo se representa a través de un diagrama UML y constituye un subconjunto del modelo de negocio.

2.3 Descripción del Problema de Dominio

El módulo de interconexión de archivos históricos constituye una aplicación que brindará la posibilidad de búsqueda y recuperación de las descripciones de los archivos históricos mediante la interconexión de instituciones archivísticas que tengan acceso a internet y/o intranet. El módulo podrá ser anexado a un sistema de archivo y los programadores o administradores de este sistema deberán realizar la configuración correspondiente con la funcionalidad que desee brindar.

La aplicación está formada por tres interfaces, una inicial que posibilitará la configuración predeterminada del módulo, otra destinada a la búsqueda y visualización de las descripciones archivísticas, y una última para la administración del módulo. La primera contiene tres opciones:

1. **Buscar solamente:** El sistema se ajustará solamente como un cliente de búsquedas, o sea, no brindará la posibilidad de que otras instituciones puedan realizar búsquedas en su institución.
2. **Permitir que busquen solamente:** Se comportará como un servidor para que otras instituciones puedan acceder a sus descripciones archivísticas, pero no podrá buscar en otras.
3. **Buscar y permitir búsquedas:** El sistema tendrá disponible todas las funcionalidades. Podrá realizar búsquedas en otras instituciones y servirá para que otras puedan acceder a sus descripciones archivísticas.

La segunda interfaz posee opciones de búsquedas simples y avanzadas, en cada una de ellas se brindará la posibilidad de buscar en alguna institución en específico o en todas. El resultado mostrará las descripciones archivísticas encontradas organizadas por instituciones.

La interfaz de administración contiene la gestión de instituciones, usuarios y una opción para volver a configurar el módulo. Aquí el administrador del sistema podrá adicionar, eliminar o modificar tanto usuarios como instituciones, esta última esencial para poder acceder a los servicios de búsquedas de otras instituciones.

El módulo brindará las siguientes funcionalidades:

- Configuración del sistema:

El administrador del sistema pueda seleccionar que funcionalidades tendrá el módulo.

- Realizar búsquedas de descripciones de archivos históricos:

Cualquier usuario puede realizar búsquedas de descripciones de archivos. Existe una búsqueda simple que se le proporcionan palabras claves que puedan identificar el archivo deseado y una búsqueda avanzada en el que se le pueden proporcionar diferentes criterios de búsqueda que restrinjan los posibles resultados a mostrar.

- Visualizar descripciones de archivos históricos:

Luego de realizar una búsqueda se mostraran los resultados, organizados por instituciones, en el que el usuario podrá elegir alguno de ellos y ver su descripción.

- Añadir instituciones archivísticas:

El administrador del sistema puede añadir el nombre y la dirección URL (Uniform Resource Locator) de alguna institución archivística que disponga del módulo de interconexión de archivos instalado, para poder acceder a sus servicios de búsquedas.

2.4 Diagrama del Modelo de Dominio

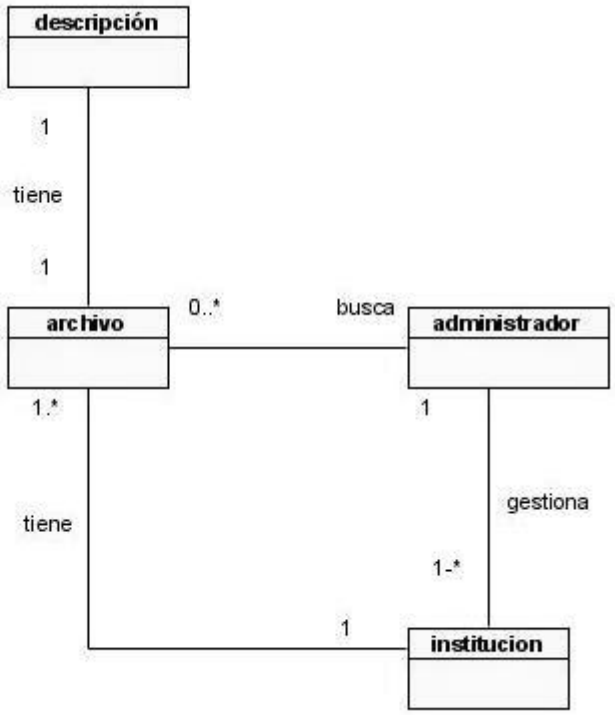

Figura: 2.1 Modelo de dominio

2.5 Requisitos del sistema.

Los requerimientos definen lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen. Los requisitos se pueden clasificar en: funcionales y no funcionales.

2.5.1 Requisitos funcionales.

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. Estos requisitos posteriormente serán modelados a través del diagrama de casos de uso del sistema.

R1 –Configurar módulo

R1.1 - Buscar solamente

R1.2 - Permitir que busquen solamente

R1.3 - Buscar y permitir búsquedas

R2 -Buscar descripciones archivísticas.

R2.1 Buscar por criterio

R2.2 Buscar por título

R2.3 - Buscar por productor

R2.4 - Fecha inicial

R2.5 - Código de referencia

R2.6 - Buscar en alguna institución archivística, o en todas

R3 - Mostrar descripción

R3.1 Listado de descripciones organizadas por instituciones

R3.2 Visualización de una descripción

R4 Gestionar institución

R4.1 - Adicionar institución

R4.2 - Eliminar institución

R4.3 - Modificar institución

R5- Autenticar usuario

R6- Gestionar usuario

R6.1 - Adicionar usuario

R6.2 - Modificar usuario

R6.3 - Eliminar usuario

R7 - Registrar usuario

2.5.2 Requisitos no funcionales.

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Son las características que hacen a la aplicación agradable, rápida, confiable, etc.

Requerimientos de Software:

- PHP 5.0.1 o superior
- PHP con la extensión SOAP activada.
- El sistema podrá utilizar diversos tipos de SGBD, soportados por la librería de abstracción de acceso a datos ADODB.
- Tener un navegador con posibilidades de solicitar y visualizar contenido HTML.
- Servidor Web que interprete código PHP.

Requerimientos de portabilidad:

- El módulo funciona en sistemas operativos Unix, Windows y Macintosh.

Requerimientos de seguridad:

- Restringir el acceso a los usuarios al área de administración.
- Los datos de los administradores serán almacenados encriptados.

Requerimientos de Ayudas y Documentación en línea:

- En el directorio de instalación del sistema será incluido un fichero nombrado LEEME.txt con las instrucciones necesaria para la instalación de la aplicación.
- El sistema incluirá un manual para el programador, para facilitar la utilización del sistema.

Requerimientos de apariencia o interfaz externa:

- Se debe tener en cuenta algunos elementos de diseño como gráficos de encabezamiento, estilos y formatos de texto.
- Utilización de CSS.

2.6 Modelo de casos de uso del sistema

2.6.1 Actores del sistema

Nombre del actor	Justificación
Usuario anónimo	Representa a cualquier persona. Tiene la posibilidad de realizar búsquedas de descripciones de archivos históricos. Puede autenticarse. Inicialmente es el encargado de la configuración del módulo.
Administrador del sistema	Es aquel que se encarga de adicionar, eliminar o modificar usuarios e instituciones archivísticas, además de la configuración del módulo.

2.6.2 Diagrama de casos de uso del sistema.

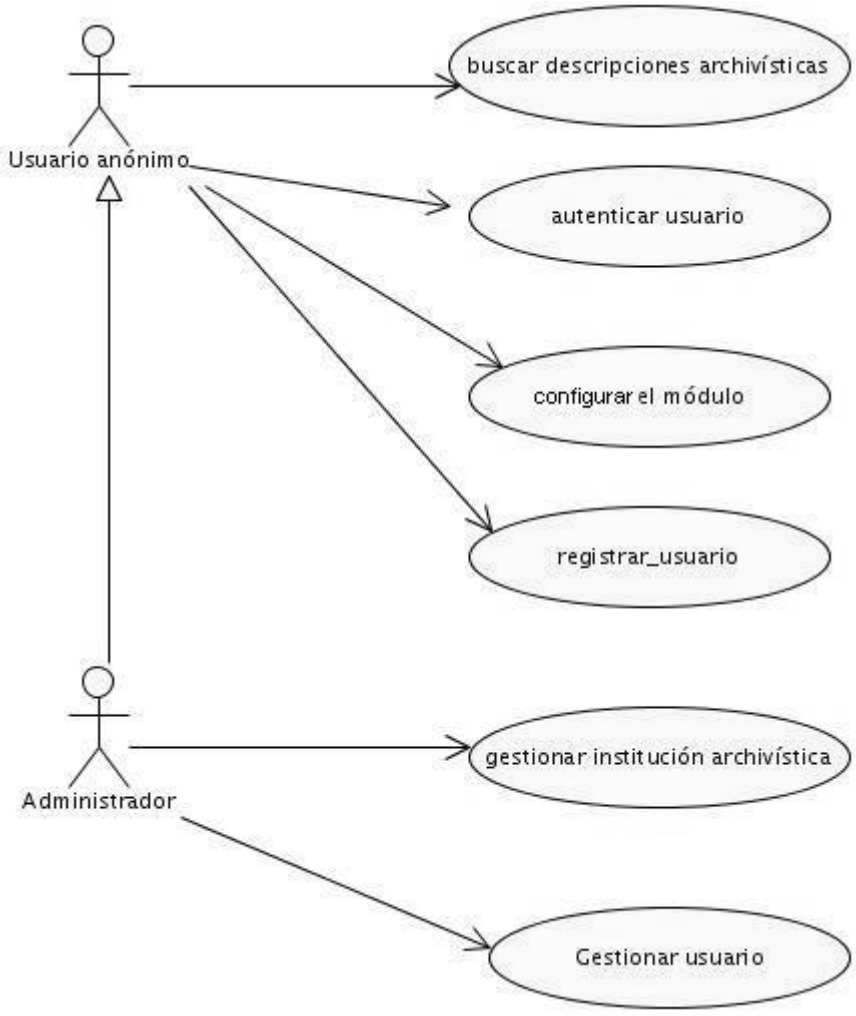

Figura 2.2: Diagrama de caso de usos del sistema

2.6.3 Descripción textual de los caso de uso del sistema

Caso de Uso:	Configurar el módulo
Propósito:	Permitir al usuario configurar el módulo.
Actores:	Usuario anónimo (Inicia)
Resumen:	El caso de uso inicia cuando el usuario anónimo accede por primera vez al módulo. Selecciona una de las tres opciones que brindará el sistema. Según la opción que seleccione el sistema la guiará en cada uno de los pasos hasta finalizar. El caso de uso culmina con la configuración del módulo.
Referencias:	R1
Precondiciones:	
Poscondiciones	Conclusión de la configuración.

Sección Buscar solamente	
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario accede por primera vez al sistema.	1.1 El sistema le brinda tres opciones de configuración.
2. El usuario selecciona la opción 1. 2.2 Introduce los datos especificados. 2.4 El usuario introduce los datos del usuario a crear.	2.1 Se solicita el gestor de base datos y el usuario y contraseña para acceder a este. 2.3 Se ofrece un formulario para la creación del primer usuario del sistema.
Flujo Alternativo	

Acción del Actor	Respuesta del Sistema
2. El usuario no selecciona una opción. 2.2 El usuario no suministra la información correcta. 2.4 El usuario introduce los datos del usuario a crear incorrectamente.	2.1 El sistema le notifica el error correspondiente. 2.3 El sistema le notifica el error correspondiente. 2.5 El sistema le notifica el error correspondiente.
Sección Permitir que busquen solamente	
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario accede por primera vez al sistema.	1.1 El sistema le brinda tres opciones de configuración.
2. El usuario selecciona la opción 2. 2.2 Introduce los datos especificados. 2.4 El usuario selecciona la base de datos. 2.6 El usuario selecciona la(s) tabla(s) que posean descripciones. 2.8 Selecciona que campos de la(s) tabla(s) corresponde con los elementos de la norma ISAD (G).	2.1 Se solicita el gestor de base datos y el usuario y contraseña para acceder a este. 2.3 Se listan las base de datos existentes. 2.5 Se muestran las tablas de la base de datos seleccionada. 2.7 Se muestran los campos de la(s) tabla(s) y los elementos de la norma ISAD(G).
Flujo Alternativo	
Acción del Actor	**Respuesta del Sistema**

2. El usuario no selecciona una opción. 2.2 El usuario no suministra la información correcta. 2.4 El usuario no selecciona una base de datos. 2.6 El usuario no selecciona la(s) tabla(s) que posean descripciones.	2.1 El sistema le notifica el error correspondiente. 2.3 El sistema le notifica el error correspondiente. 2.5 El sistema le notifica el error correspondiente. 2.7 El sistema le notifica el error correspondiente.
Sección Buscar y permitir búsquedas	
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1 El usuario accede por primera vez al sistema.	1.1 El sistema le brinda tres opciones de configuración.
2. El usuario selecciona la opción 3. 2.3 Introduce los datos especificados. 2.4 El usuario selecciona la base de datos. 2.6 El usuario selecciona la(s) tabla(s) que posean descripciones. 2.8 Selecciona que campos de la(s) tabla(s) corresponde con los elementos de la norma ISAD (G). 2.10 El usuario introduce los datos del usuario a crear.	2.1 Se solicita el gestor de base datos y el usuario y contraseña para acceder a este. 2.3 Se listan las base de datos existentes. 2.5 Se muestran las tablas de la base de datos seleccionada. 2.7 Se muestran los campos de la(s) tabla(s) y los elementos de la norma ISAD (G). 2.9 Se ofrece un formulario para la creación del primer usuario del sistema.

Flujo Alternativo	
Acción del Actor	**Respuesta del Sistema**
2. El usuario no selecciona una opción. 2.3 El usuario no suministra la información correcta. 2.4 El usuario no selecciona una base de datos. 2.6 El usuario no selecciona la(s) tabla(s) que posean descripciones. 2.10 El usuario introduce los datos del usuario a crear incorrectamente.	2.1 El sistema le notifica el error correspondiente. 2.3 El sistema le notifica el error correspondiente. 2.5 El sistema le notifica el error correspondiente. 2.7 El sistema le notifica el error correspondiente. 2.11 El sistema le notifica el error correspondiente.
Prioridad	Crítica
Prototipo de interfaz	

Caso de Uso:	Buscar descripciones archivísticas.
Propósito:	Permitir a un usuario la búsqueda de descripciones de archivos históricos.
Actores:	Usuario anónimo (Inicia)
Resumen:	El caso de uso inicia cuando el usuario anónimo decide realizar la búsqueda de la descripción de un determinado archivo histórico e introduce sus criterios. El caso de uso finaliza cuando se le muestra al usuario el resultado de su búsqueda y se procede a realizar otra acción.
Referencias:	R2
Precondiciones:	
Poscondiciones	Se muestran el resultado de la búsqueda.

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario anónimo accede a la interfaz de búsqueda.	1.1 El sistema le brinda diferentes opciones para la búsqueda.
2. El usuario anónimo introduce sus criterios de búsqueda.	2.1. El sistema realiza la búsqueda. 2.2. El sistema muestra los resultados de la búsqueda.
Flujo Alternativo	
Acción del Actor	**Respuesta del Sistema**
	2.1.1 Si los criterios de búsqueda del usuario anónimo no producen ningún resultado se le notifica al usuario. 2.1.2 Si introduce un criterio menor de tres

	letras, el sistema le notifica al usuario que su criterio es muy escueto.
Prioridad	Crítico
Prototipo de interfaz	

Caso de Uso	Gestionar institución
Actores	Administrador
Propósito	Permitir a un administrador adicionar, modificar o eliminar una institución.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción de adicionar, modificar o eliminar una institución, el sistema le solicita una serie de datos que debe introducir. Cuando el usuario envía estos datos el sistema los verifica y el caso de uso finaliza cuando se actualizan todos cambios realizados.
Referencias	R3
Precondiciones	-----
Poscondiciones	Se registran los cambios realizados por el administrador.
Sección Adicionar institución	

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El administrador selecciona la opción adicionar una institución.	1.1 El sistema le solicita una serie de datos que debe introducir.
2. El administrador introduce los datos.	2.1 El sistema verifica los datos. 2.2 El sistema verifica si la institución existe. 2.3 El sistema adiciona una institución a la base de datos y muestra un mensaje de notificación.
Flujo Alternativo	
Acción del Actor	**Respuesta del Sistema**
	2.1 Si los datos no son válidos el sistema emite un mensaje de error. 2.2 Si la institución existe el sistema envía un mensaje de error.
Sección Modificar institución	
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El administrador selecciona la institución a modificar.	1.1 El sistema le muestra los datos de la institución.
2. El administrador modifica los datos de la institución.	2.1 El sistema verifica si los datos son válidos 2.2 El sistema actualiza los cambios y muestra un mensaje de notificación.
Flujo Alternativo	

Acción del Actor	Respuesta del Sistema
	2.1 Si los datos no son válidos el sistema emite un mensaje de error.
Sección Eliminar institución	
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El administrador selecciona la institución que desea eliminar.	2. El sistema elimina la institución y muestra un mensaje de notificación.
Prioridad	Crítico
Prototipo de interfaz	

Caso de Uso:	Autenticar usuario
Propósito:	Permite al usuario anónimo acceder a la interfaz de administración.
Actores:	Usuario anónimo (Inicia)
Resumen:	El caso de uso inicia cuando el usuario anónimo desea acceder a la sección administrativa del sistema y proporciona su nombre y contraseña. El sistema verifica si son válidos, de no suceder así, le muestra un mensaje de error y le da la

	opción de acceder nuevamente, de lo contrario le da acceso al sistema. El caso de uso finaliza cuando el usuario anónimo se ha autentificado.
Referencias:	R4
Precondiciones:	El usuario debe estar registrado.
Poscondiciones	El administrador debe estar autentificado.
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario anónimo selecciona la opción de autenticarse.	1.1 El sistema solicita los datos necesarios (usuario/contraseña).
2. El usuario anónimo introduce sus datos.	2.1. El sistema verifica sus datos. 2.2. El sistema lo redirecciona a la interfaz de administración.
Flujo Alternativo	
Acción del Actor	**Respuesta del Sistema**
	2.1. Si el usuario no entra correctamente los datos se notifica un error.
Prioridad	Crítico

Prototipo de interfaz	

Caso de Uso	Gestionar usuario
Actores	Administrador
Propósito	Permitir a un administrador adicionar, modificar o eliminar un usuario.
Resumen	El caso de uso inicia cuando el administrador selecciona la opción de adicionar, modificar o eliminar un usuario, el sistema le solicita una serie de datos que debe introducir. Cuando el usuario envía estos datos el sistema los verifica y el caso de uso finaliza cuando se actualizan todos datos.
Referencias	R4
Precondiciones	-----
Poscondiciones	Se registran los cambios realizados por el administrador.
Sección Adicionar usuario	
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El administrador selecciona la opción adicionar usuario.	1.1 El sistema le solicita una serie de datos que debe introducir.
2. El administrador introduce los datos.	2.1 El sistema verifica los datos. 2.2 El sistema verifica si el usuario existe. 2.3 El sistema adiciona el nuevo usuario a la base de datos y muestra un mensaje de notificación.
Flujo Alternativo	
Acción del Actor	**Respuesta del Sistema**
	2.1 Si los datos no son válidos el sistema emite un

	mensaje de error. 2.2 Si el usuario existe el sistema envía un mensaje de error.
Sección Modificar usuario	
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El administrador selecciona el usuario a modificar.	1.1 El sistema le muestra los datos del usuario.
2. El administrador modifica los datos del usuario.	2.1 El sistema verifica si los datos son válidos 2.2 El sistema actualiza los cambios y muestra un mensaje de notificación.
Flujo Alternativo	
Acción del Actor	**Respuesta del Sistema**
	2.1 Si los datos no son válidos el sistema emite un mensaje de error.
Sección Eliminar usuario	
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El administrador selecciona el usuario que desea eliminar.	1.2 El sistema verifica que el usuario a eliminar no sea el que este autenticado en ese momento.
	2. El sistema elimina el usuario y muestra un mensaje de notificación.
Flujo Alternativo	

Acción del Actor	Respuesta del Sistema
	1.1 Si el usuario no existe envía un mensaje de aviso.
Prioridad	Crítico
Prototipo de interfaz	

Caso de Uso:	Registrar usuario
Propósito:	Permite crear el primer administrador del sistema.
Actores:	Usuario anónimo (Inicia)
Resumen:	El caso de uso se inicia cuando el usuario que va a administrar el módulo accede por primera vez. El sistema proporcionará un formulario para introducir sus datos de acceso, con los cuales tendrá permiso posteriormente para administrar el sistema. El caso de uso finaliza cuando el usuario es registrado
Referencias:	R6
Precondiciones:	
Poscondiciones	Se registra el usuario.
Flujo Normal de Eventos	

Acción del Actor	Respuesta del Sistema
1. El usuario anónimo accede por primera vez al sistema.	1.1 El sistema le brinda la posibilidad de registrarse.
2. El usuario anónimo introduce sus datos.	2.1 El sistema verifica los datos. 2.2. El sistema registra al usuario.
Flujo Alternativo	
Acción del Actor	**Respuesta del Sistema**
	2.1 Si el usuario no introduce correctamente los datos el sistema le notifica un mensaje de error.
Prioridad	Crítica
Prototipo de interfaz	

2.7 Conclusiones del capítulo:

En este capítulo se mostró la descripción del problema del dominio para un mayor entendimiento del negocio, y se describió la solución propuesta al problema de investigación. Además se especifican los

principales actores y casos de usos con que constará el sistema, así como la descripción detallada de los mismos

Capitulo 3. Análisis y Diseño del sistema

Para el buen desarrollo del módulo de interconexión de archivos históricos se necesitó en su fase de elaboración del análisis y diseño del mismo. A través del los diferentes diagramas modelados para cada caso de uso, fue posible obtener una mayor comprensión y guía para lograr una buena implementación.

3.1 Análisis del sistema

El análisis permite obtener una visión del sistema, en este flujo de trabajo solo interesan los requisitos funcionales para lograr un refinamiento de la funcionalidad del sistema.

3.1.1 Diagrama de Clases del Análisis

Diagrama de clases de Análisis: Autenticar usuario

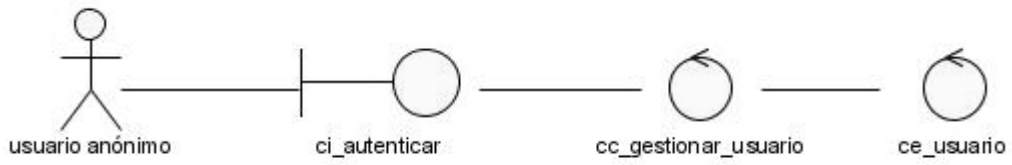

Figura 3.1 Diagrama de Clases del Análisis (CU Autenticar usuario)

Diagrama de clases de Análisis: Configurar módulo

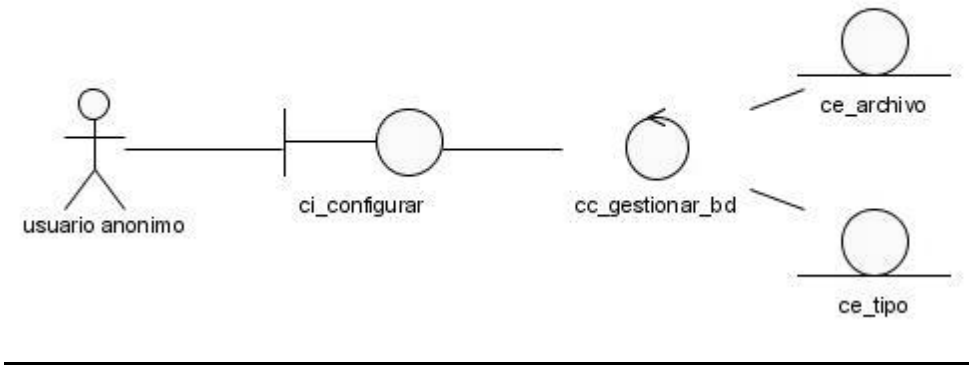

Figura 3.2 Diagrama de Clases del Análisis (CU Configurar módulo)

Diagrama de clases de Análisis: Buscar descripciones archivísticas.

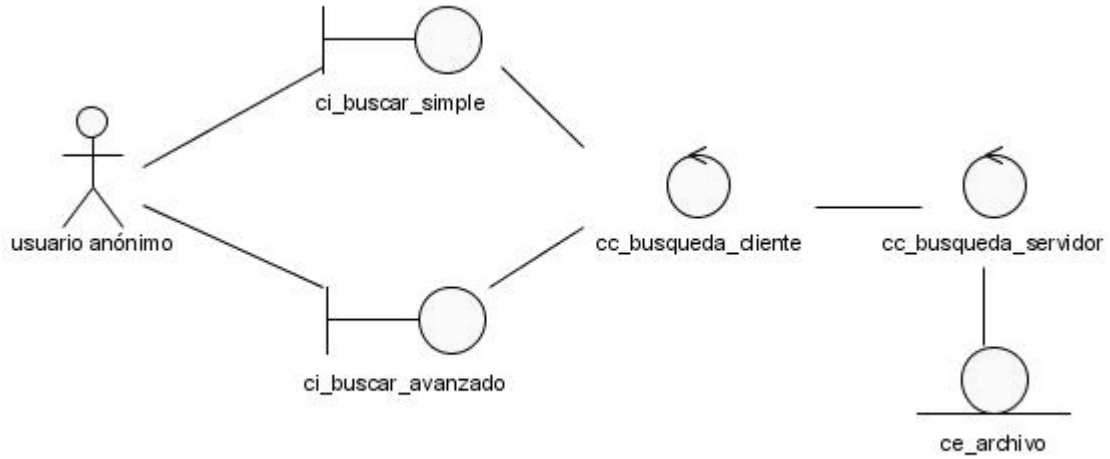

Figura 3.3 Diagrama de Clases del Análisis (CU Buscar descripción)

Diagrama de clases de Análisis: Gestionar institución

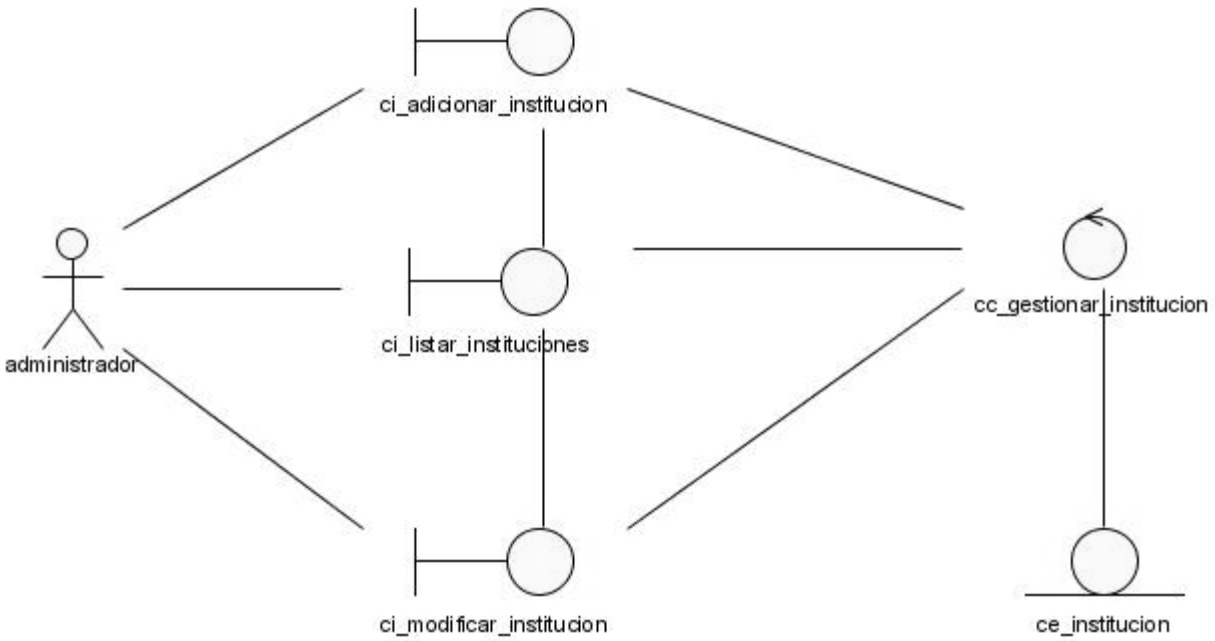

Figura 3.4 Diagrama de Clases del Análisis (CU Gestionar institución)

Diagrama de clases de Análisis: Registrar usuario

Figura 3.5 Diagrama de Clases del Análisis (CU Registrar usuario)

Diagrama de clases de Análisis: Gestionar usuario

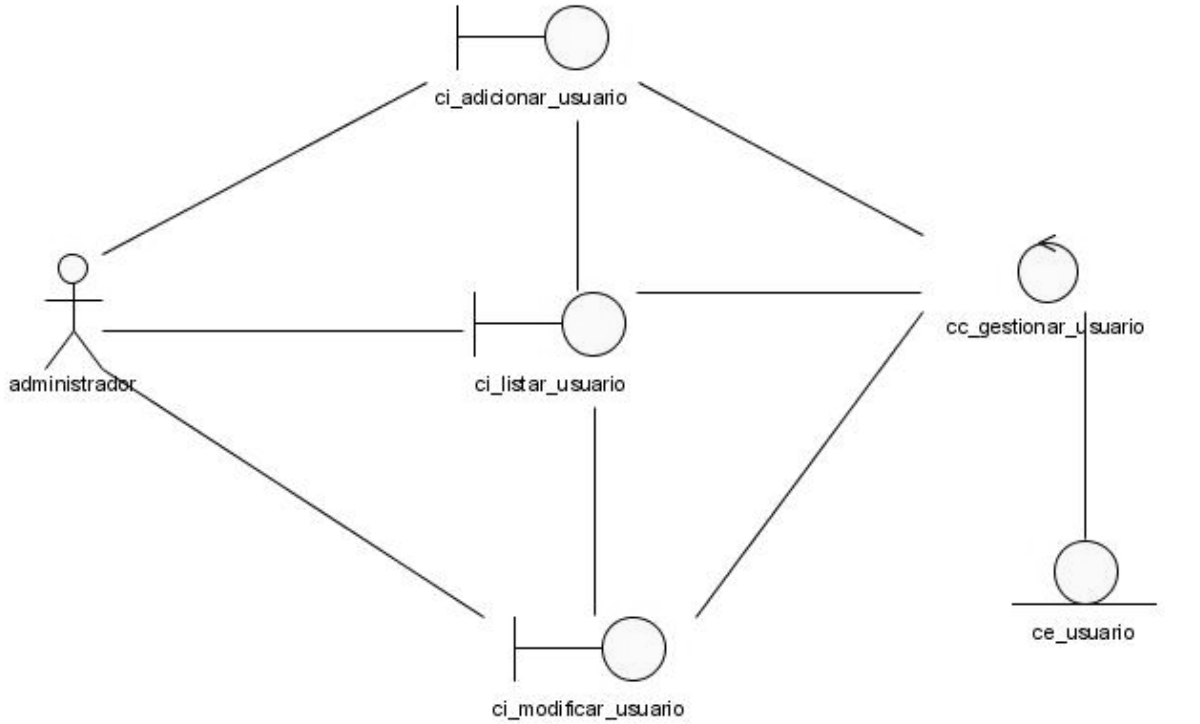

Figura 3.6 Diagrama de Clases del Análisis (CU Gestionar usuario)

3.1.2 Diagramas de interacción

Los diagramas de interacción modelan el comportamiento dinámico del sistema, así como la interacción entre objetos a través de mensajes. Existen dos tipos diagramas de interacción, los de colaboración y los de secuencia.

Los diagramas de colaboración destacan la organización estructural de los objetos que envían y reciben mensajes. A continuación se representan los diagramas de colaboración correspondientes a cada caso de uso del sistema.

Diagrama de colaboración: Autenticar usuario

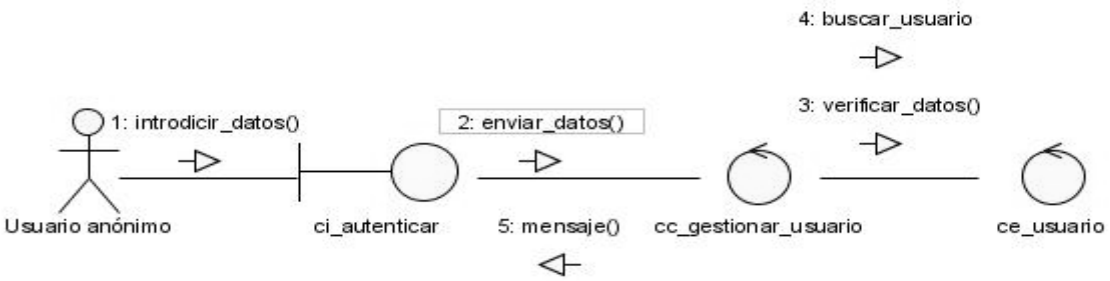

Figura 3.7 Diagrama de colaboración (CU Autenticar usuario)

Diagrama de colaboración: Gestionar institución

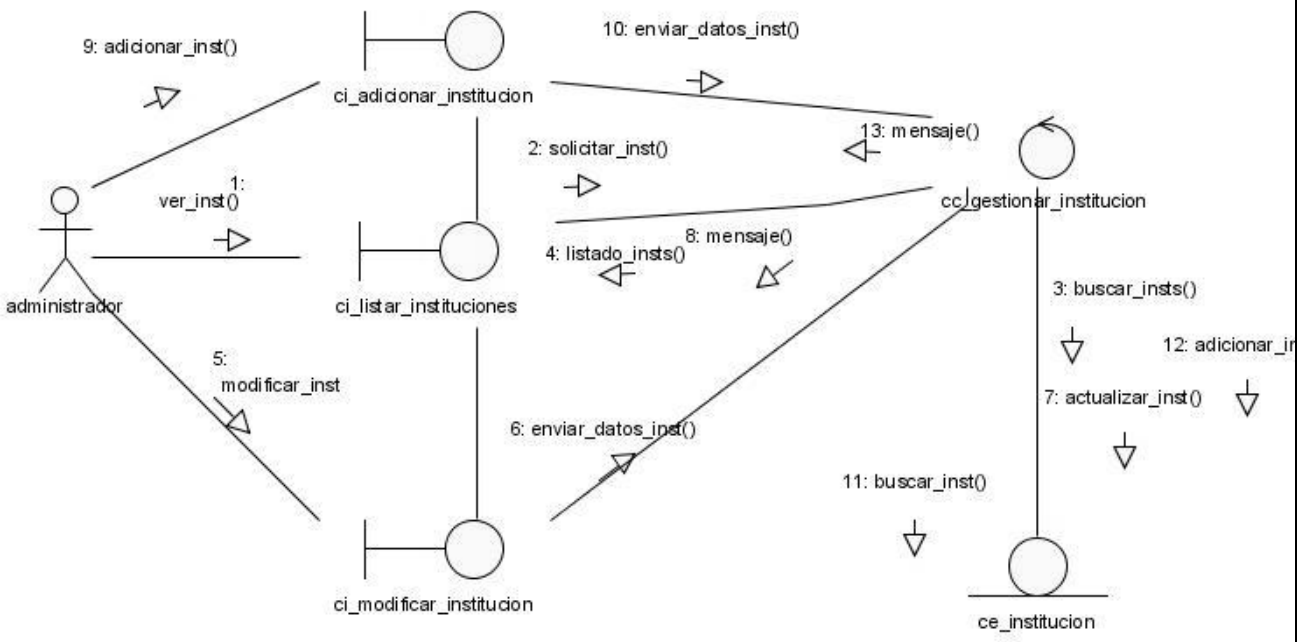

Figura 3.8 Diagrama de colaboración (CU Gestionar institución)

Diagrama de colaboración: Gestionar usuario

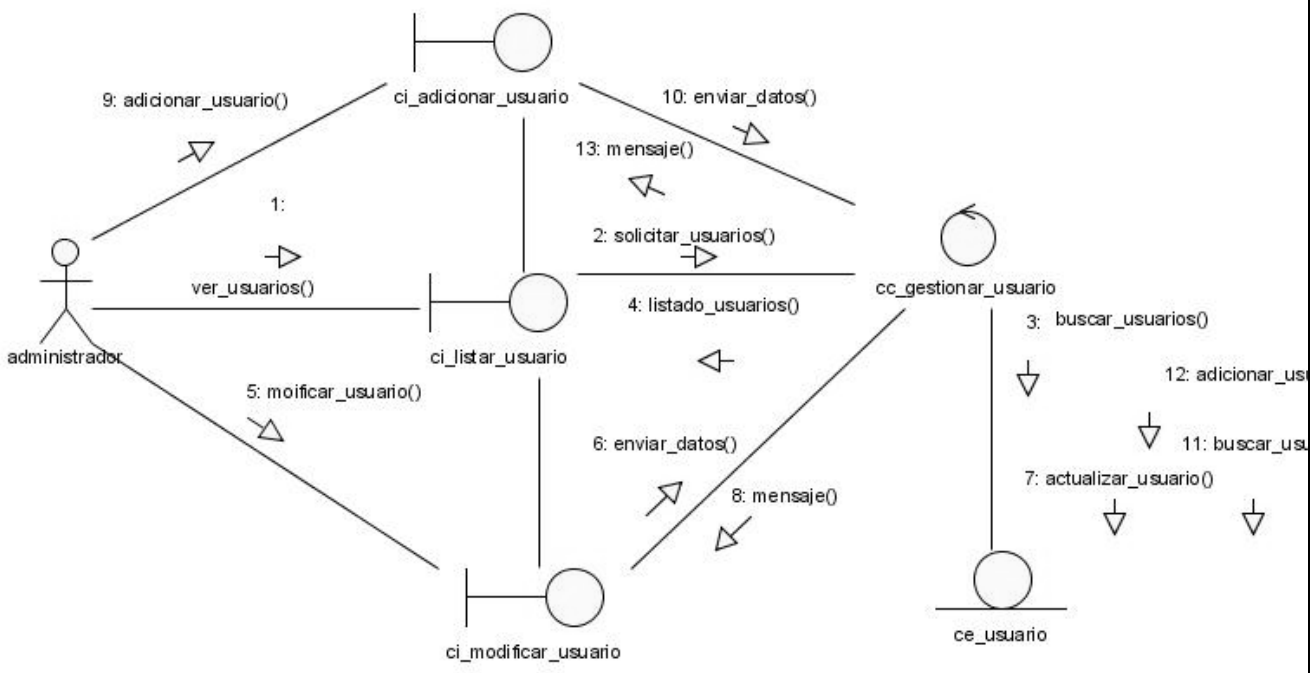

Figura 3.9 Diagrama de colaboración (CU Gestionar usuario)

Diagrama de colaboración: Buscar descripciones archivísticas.

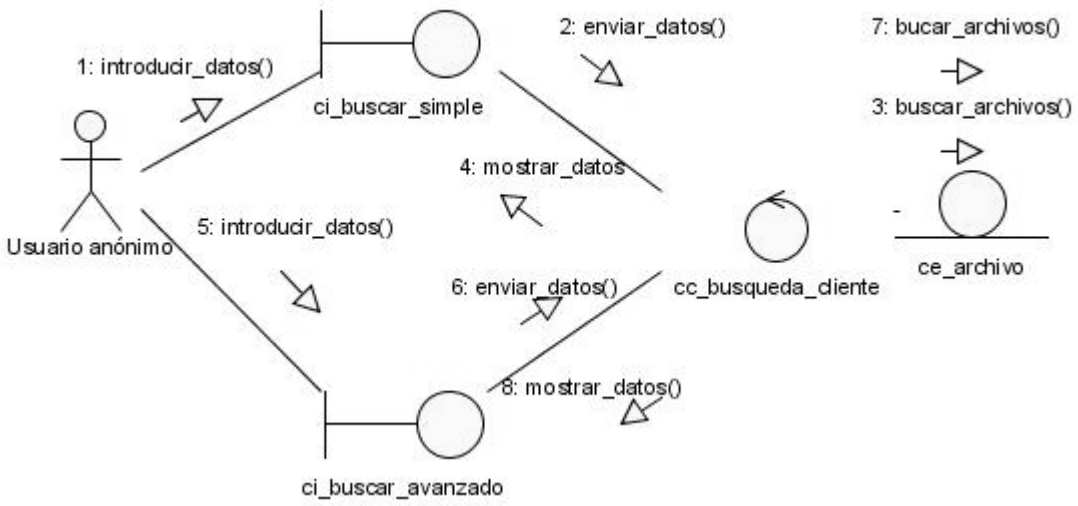

Figura 3.10 Diagrama de colaboración (CU Buscar descripción de archivos)

Diagrama de colaboración: Configurar módulo

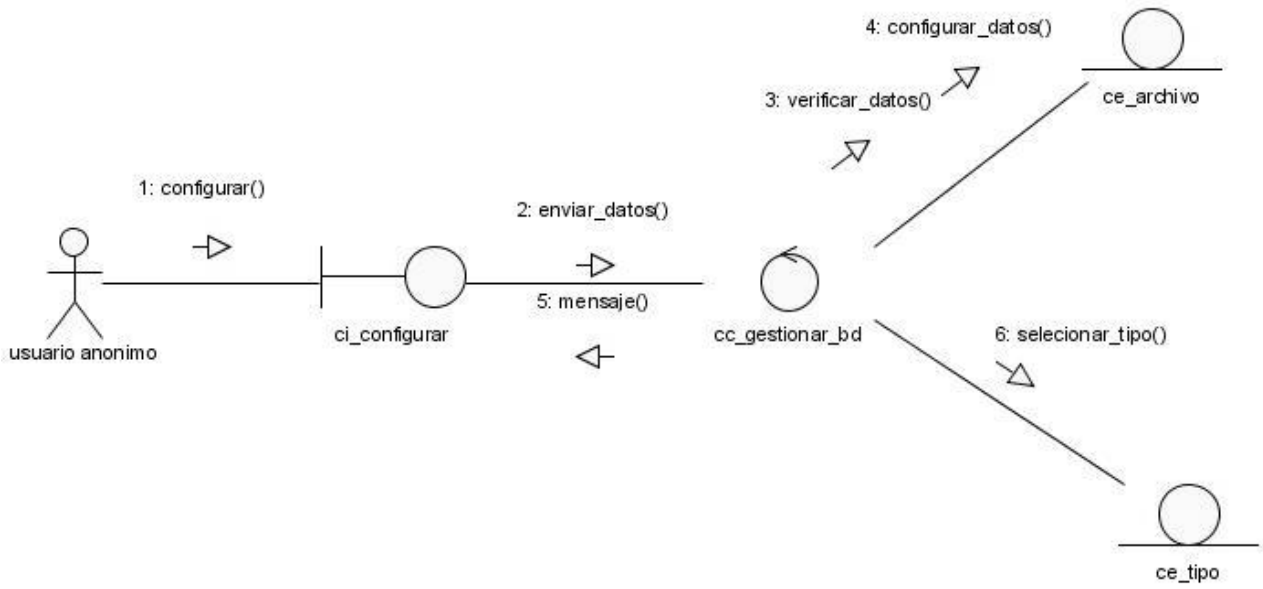

Figura 3. 11 Diagrama de colaboración (CU Configurar módulo)

Diagrama de colaboración: Registrar usuario

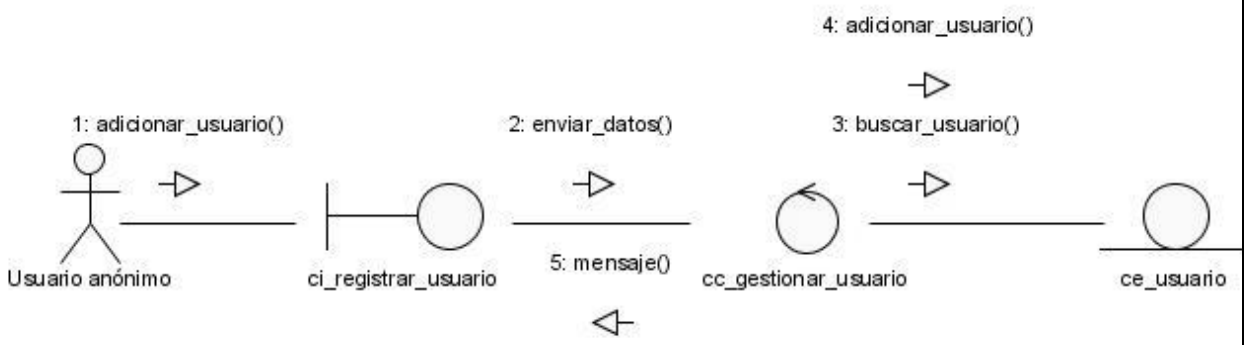

Figura 3.12 Diagrama de colaboración (CU Registrar usuario)

3.2 Diseño del sistema

El modelo de diseño permite adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales. Contribuye en gran medida a lograr una arquitectura estable y sólida, y crear un plano del modelo de implementación.

3.2.1 Diagrama de Clases del Diseño

Diagrama de clases: CU Configurar módulo

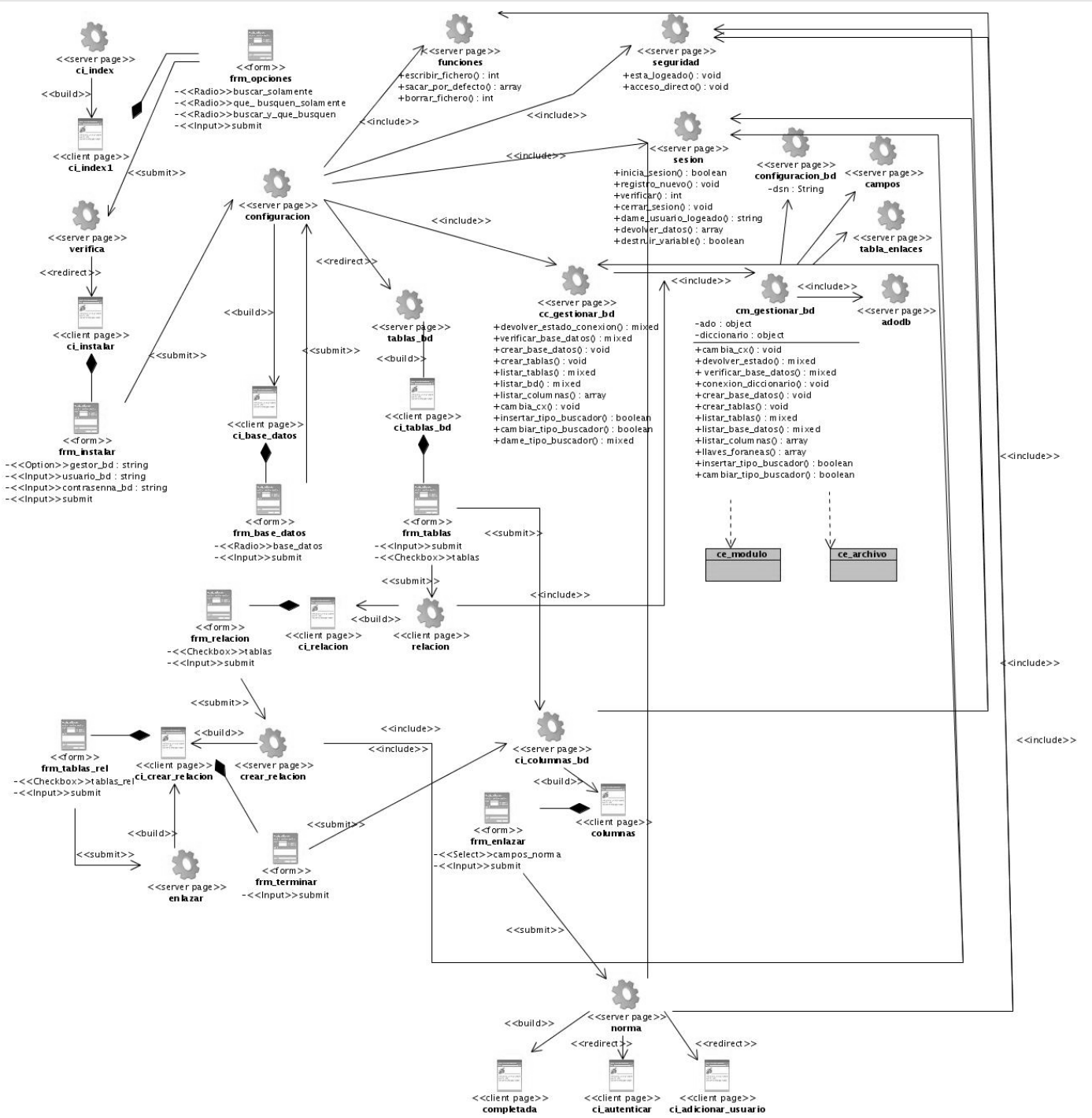

Diagrama de clases: CU Buscar descripciones archivísticas.

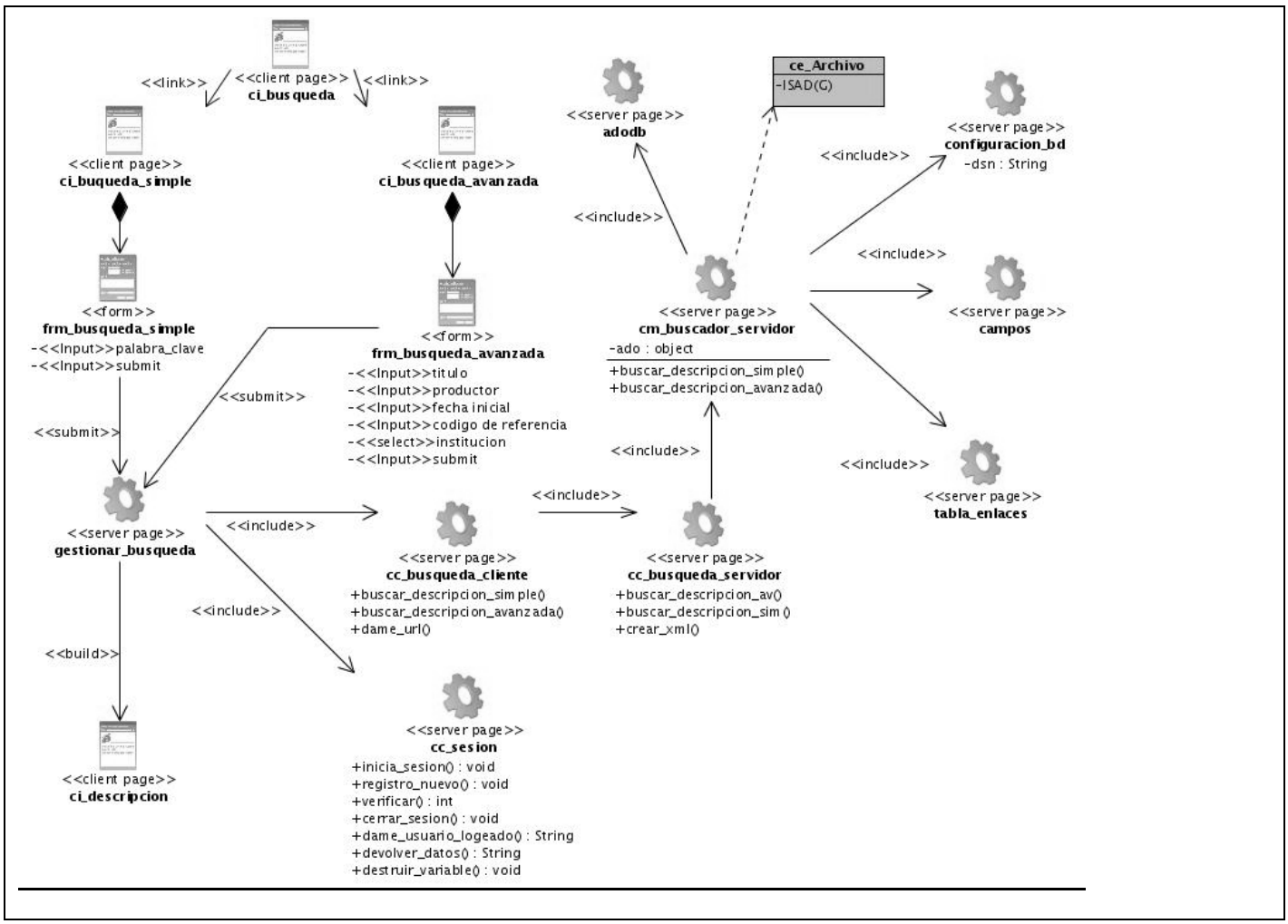

Figura 3.14 Diagrama de clases (CU Buscar descripción de archivos históricos)

Diagrama de clases: CU Gestionar institución

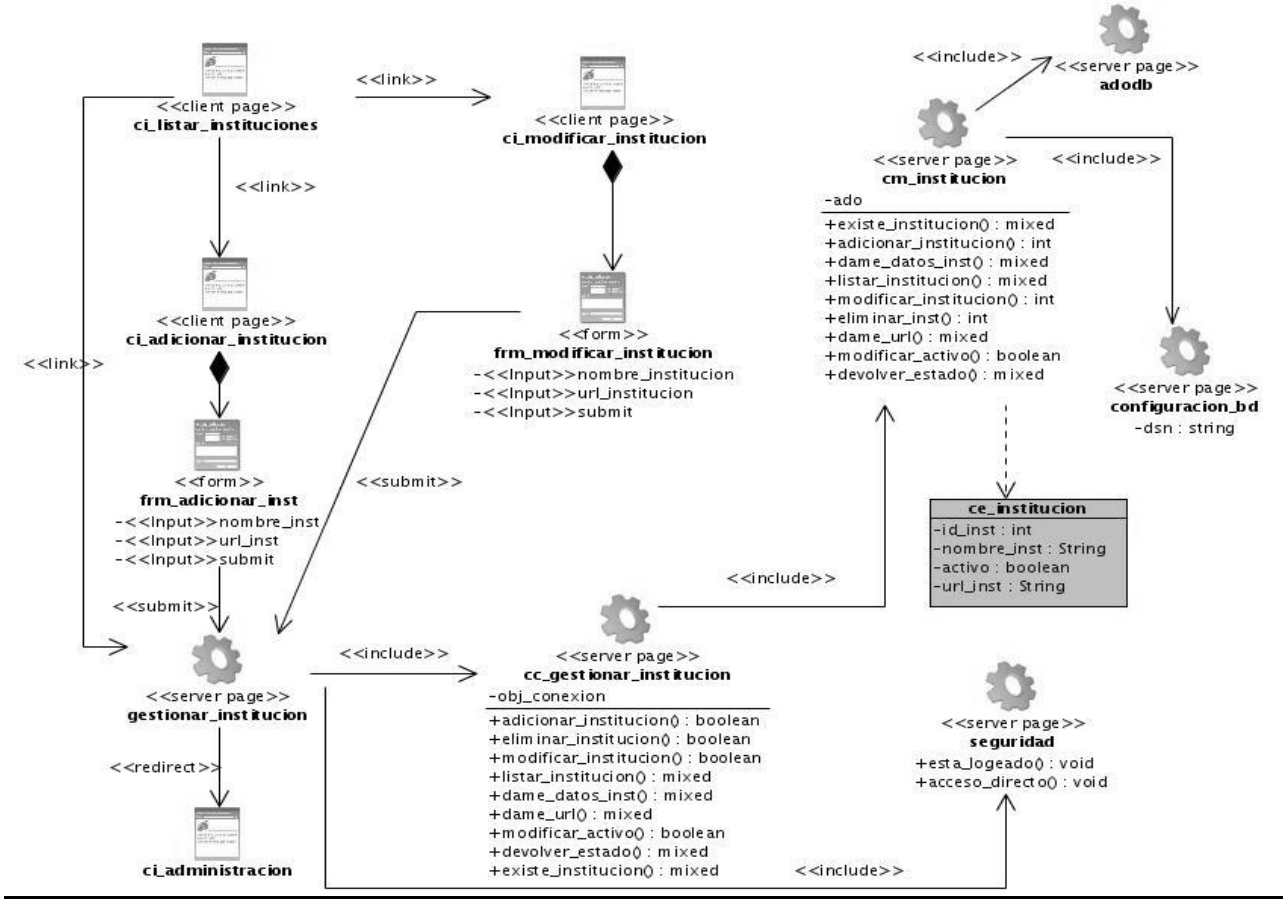

Figura 3.15 Diagrama de clases (CU Gestionar institución)

Diagrama de clases: CU Gestionar usuario

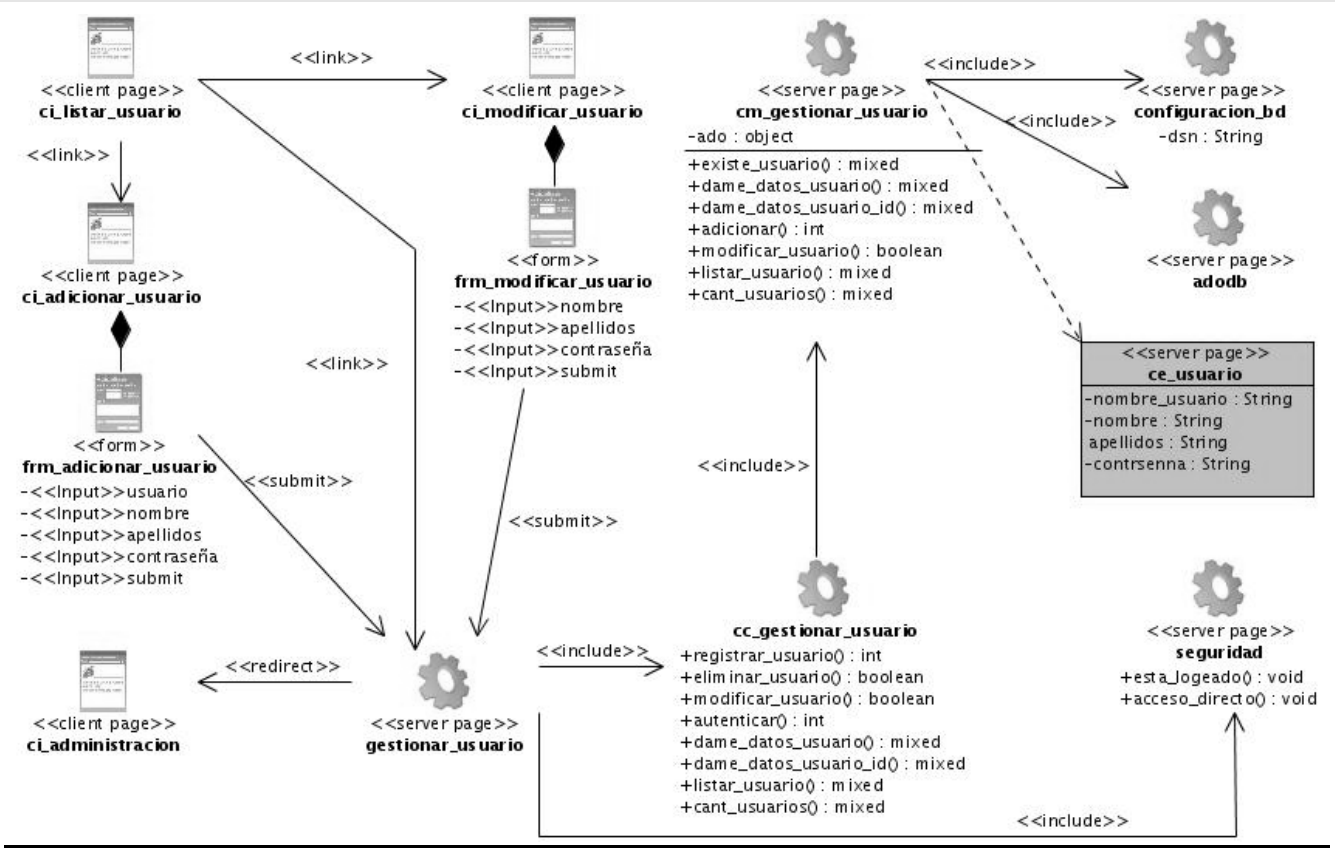

Figura 3.16 Diagrama de clases (CU Gestionar usuario)

Diagrama de clases: CU Registrar usuario

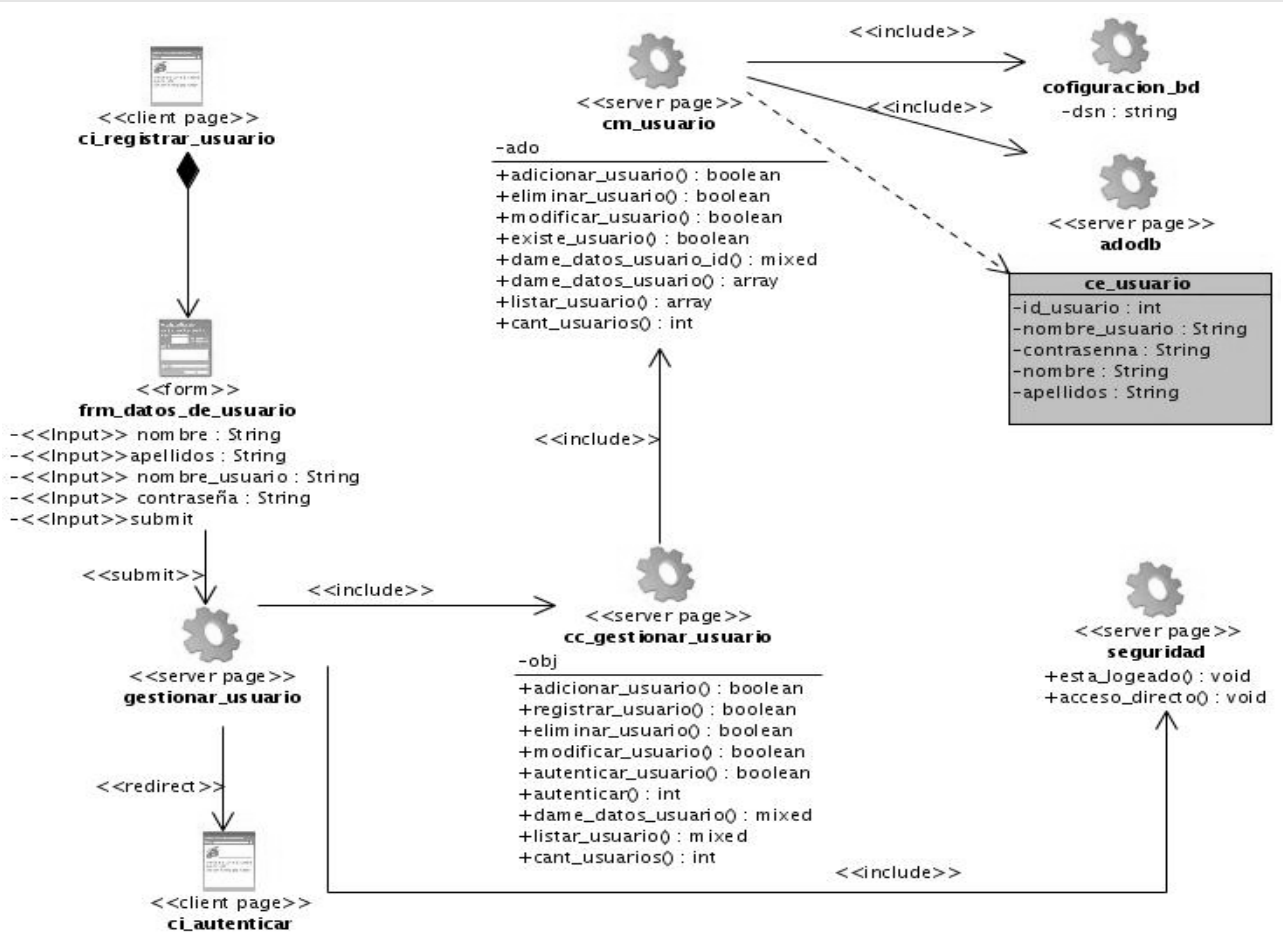

Figura 3.17 Diagrama de clases (CU Registrar usuario)

Diagrama de clases: CU Autenticar usuario

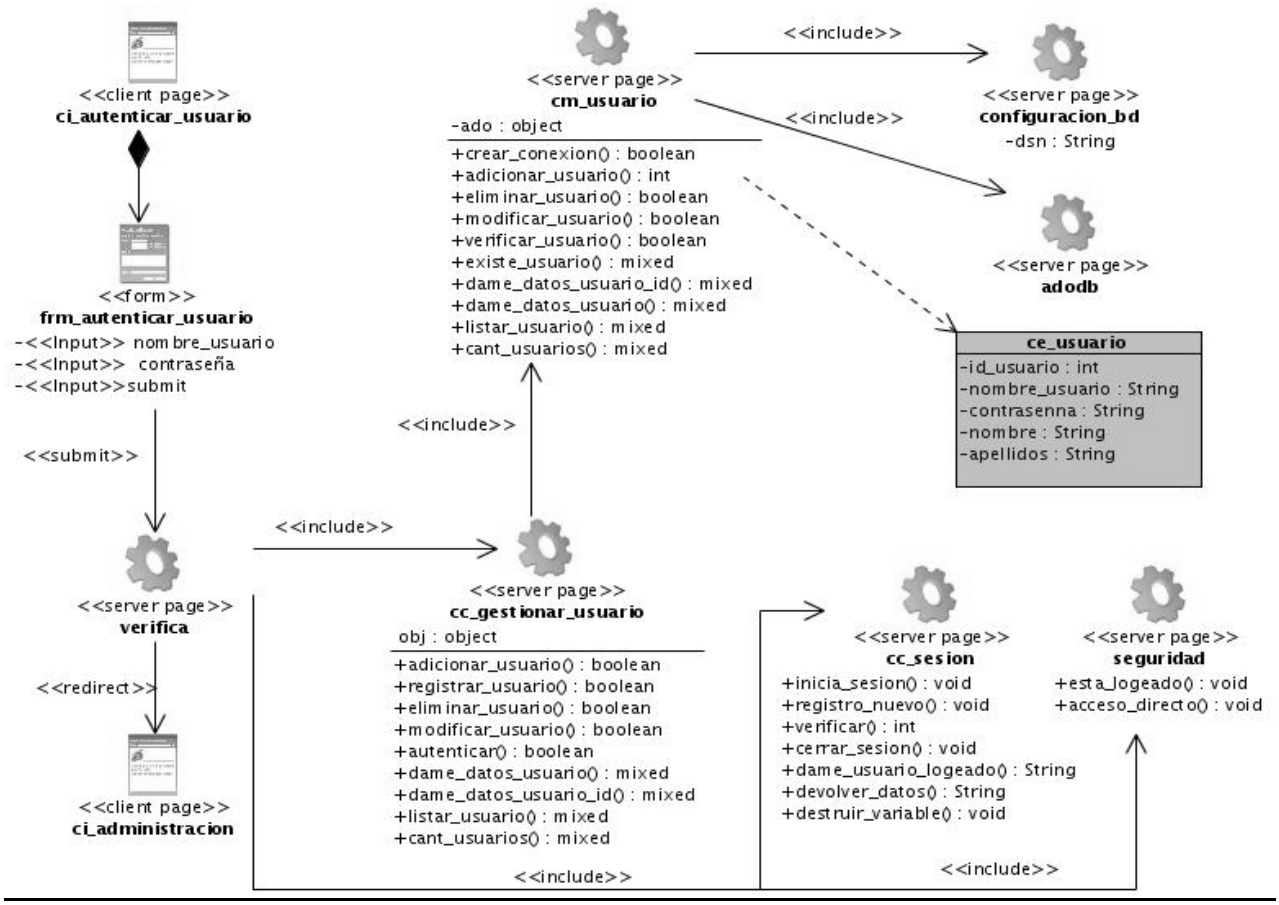

Figura 3.18 Diagrama de clases (CU Autenticar usuario)

3.3 Diseño de la base de datos

3.3.1 Diagrama de clases persistentes

Diagrama de clases persistentes

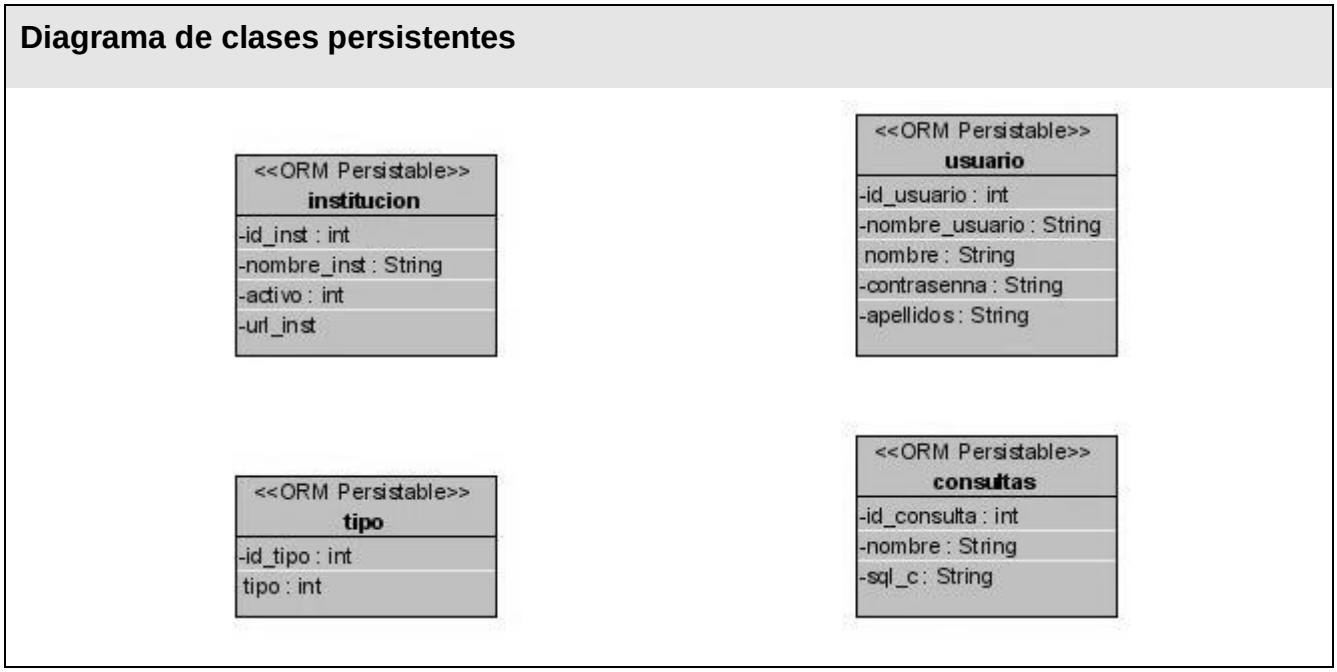

Figura 3.19 Diagrama de clases persistentes.

3.3.2 Modelo de datos

Modelo de datos

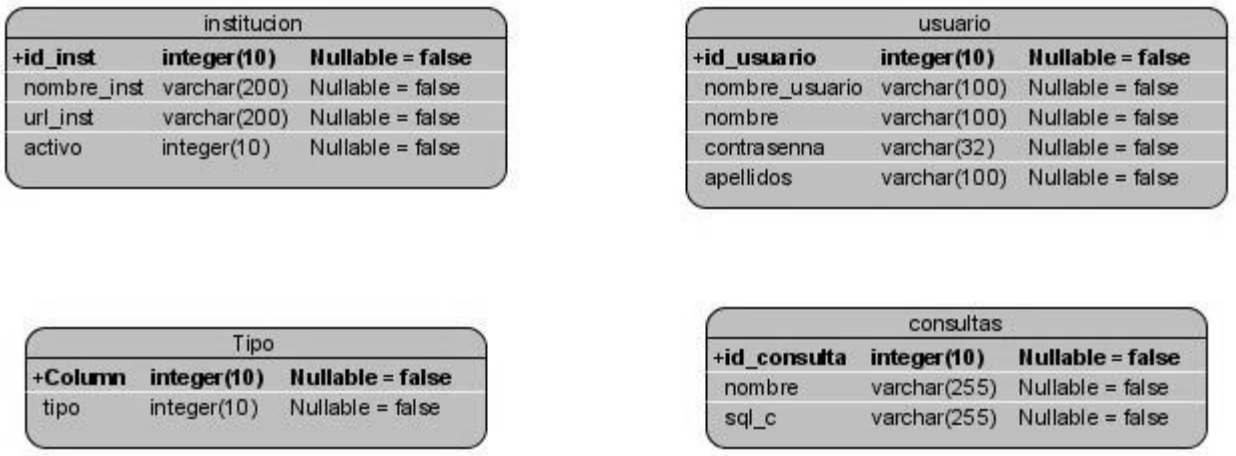

Figura 3.20 Modelo de datos.

3.2.3 Descripción de las tablas y atributos

Nombre: Usuario		
Descripción: En esta tabla se almacenan todos los usuarios con privilegios de administración.		
Atributo	**Tipo**	**Descripción**
id_usuario	Int(10)	Es el identificador de la tabla, que constituye un numero autogenerado de forma incrementar
nombre	varchar(100)	Nombre del usuario
nombre_usuario	varchar(50)	Usuario con el que se accede al módulo
apellidos	varchar(100)	Apellidos del usuario
contraseña	varchar(100)	Contraseña del usuario para acceder al módulo

Tabla3.1 Descripción de la tabla usuario

Nombre: Institución		
Descripción: En esta tabla se almacena el nombre y la dirección donde esta prestando el servicio la institución.		
Atributo	**Tipo**	**Descripción**
id_inst	Int(10)	Es el identificador de la tabla, que constituye un numero autogenerado de forma incrementar
nombre_inst	varchar(200)	Nombre de la institución
activo	Int(11)	Si la institución esta activa o no.
url_inst	varchar(200)	Dirección donde esta publicado el servicio web

Tabla3.2 Descripción de la tabla institución

Nombre: Tipo		
Descripción: En esta tabla se almacena el tipo de funcionalidad que estará disponible en el módulo		
Atributo	**Tipo**	**Descripción**
id_tipo	Int(10)	Es el identificador de la tabla, que constituye un numero autogenerado de forma incrementar
tipo	int(10)	Tipo de funcionalidad. 1-Buscar solamente 2-Permitir búsquedas solamente 3-Buscar y permitir búsquedas.

Tabla3.3 Descripción de la tabla tipo

Nombre: Consulta		
Descripción: En esta tabla se almacena el nombre de la institución y la consulta SQL correspondiente a ella.		
Atributo	**Tipo**	**Descripción**
id_consulta	Int(10)	Es el identificador de la tabla, que constituye un numero autogenerado de forma incrementar
nombre	varchar(200)	Nombre de la institución
sql_C	text	Consulta SQL generada a partir de los parámetros de búsqueda introducidos.

Tabla3.4 Descripción de la tabla consulta.

3.3 Diseño de la interfaz.

El diseño de la interfaz es algo fundamental a la hora de elaborar el sistema, debido a que esta será la capa que interactuará con el usuario final, y por lo tanto debe ser lo más amigable y comprensible posible.

En el diseño de las pantallas del sistema, se tuvo en cuenta dos principios fundamentales como son:

1- Lograr un sistema de fácil interacción.

2- Darle respuesta al usuario a cada acción que realice en el sistema.

Para lograr estos puntos se optimizará la cantidad de elementos en las pantallas y todas las páginas mostrarán la información con una misma estructura, además se guiará al usuario en la utilización del módulo mostrándole las indicaciones precisas para el uso del mismo, y dándole respuestas visuales a cada acción que realice.

3.4 Tratamiento de errores

Ningún sistema informático o software es 100% seguro y confiable, por lo tanto están propensos a que en un determinado momento produzcan un error. Para lograr una mayor calidad del software es preciso que estos errores se hayan tratado de antemano.

En este sistema pueden ocurrir dos tipos de errores. Cuando ocurre un fallo en la funcionalidad del sistema, que es cuando ocurre algún error en las operaciones con la base de datos como inserción, modificación y eliminación de algún registro; estos errores serán capturados y se le mostrará un mensaje de error al usuario notificándole lo ocurrido. El otro error posible es cuando el usuario realiza una acción que el sistema no espera.

Estos errores son capturados y la mayoría validados en el servidor, debido que en el cliente existe la posibilidad de no ser interpretadas si el navegador no tiene activado Java Script.

3.5 Concepción de la ayuda

El sistema cuenta con un manual para el programador, en que se detallan todos los métodos y clases, especificando, los parámetros, tipos de retorno y de datos, y una descripción de la función que realiza. El módulo cuenta además con una breve instrucción al usuario, en el que se le orienta como trabajar con el sistema para un óptimo desempeño.

3.6 Conclusiones

En este capitulo se modelaron los artefactos relacionados con el flujo de trabajo de Análisis y Diseño. Se realizaron los diagramas de clases del análisis, así como los de colaboración que facilitaron el desarrollo de los diagramas del diseño, estos últimos de vital importancia para la el flujo de trabajo de implementación. Además se definieron los detalles del diseño que se proponen, así como el modelo de datos para el desarrollo de la base de datos.

Capítulo 4. Implementación y Prueba

4.1 Introducción

En los siguientes flujos de trabajo para la elaboración del módulo se tomará el resultado del flujo de diseño y se implementará el sistema en términos de componentes; o sea, es la transición del diseño de clases a la creación de componentes (ficheros código fuente, scripts, código binario, ejecutables, etc.). Además se ejecutará el sistema bajo una serie de condiciones o requerimientos específicos donde los resultados serán observados y registrados, para evaluar la funcionalidad del software.

4.2 Diagrama de componentes

Un diagrama de componentes representa un grafo de componentes software unidos por relaciones de dependencia lógica, o sea muestra un conjunto de elementos del modelo tales como componentes, subsistemas de implementación y sus relaciones. Los elementos de modelado dentro de un diagrama de componentes serán componentes y paquetes. En cuanto a los componentes, sólo aparecen tipos de componentes, ya que las instancias específicas de cada tipo se encuentran en el diagrama de despliegue. Por lo general los diagramas de componentes se utilizan para modelar código fuente, versiones ejecutables y bases de datos físicas.

Código fuente: En el modelado de código fuentes se suelen utilizar para representar las dependencias entre los ficheros de código fuente, o para modelar las diferentes versiones de estos ficheros. Para ello se deben identificar el conjunto de archivos de código fuente de interés y estereotiparlos como archivos “file”, agruparlos en paquetes, utilizar valores etiquetados para la información de versiones y modelar las dependencias de compilación.

Código ejecutable: Se utiliza para modelar la distribución de una nueva versión a los usuarios. Para tal propósito se identifican el conjunto de componentes ejecutables que intervienen, se utilizan estereotipos para los diferentes tipos de componentes (ejecutables, bibliotecas, tablas, archivos, documentos, etc.), se consideran las relaciones entre dichos componentes que la mayoría de las veces incluirán interfaces que son exportadas (realizadas) por ciertos componentes e importadas (utilizadas) por otros.

Bases de datos física: UML permite el modelado de bases de datos físicas así como de los esquemas lógicos de bases de datos.

Se pueden utilizar estereotipos como <<link>> o <<compile>> para distinguir la distinta naturaleza de las dependencias. Igualmente se pueden definir estereotipos para las distintas clases de componentes. UML proporciona algunos estereotipos predefinidos como: <<file>>, <<library>>, <<executable>>, <<table>>, <<scripts>> y <<document>>.

Diagrama de componentes: Paquete Buscador

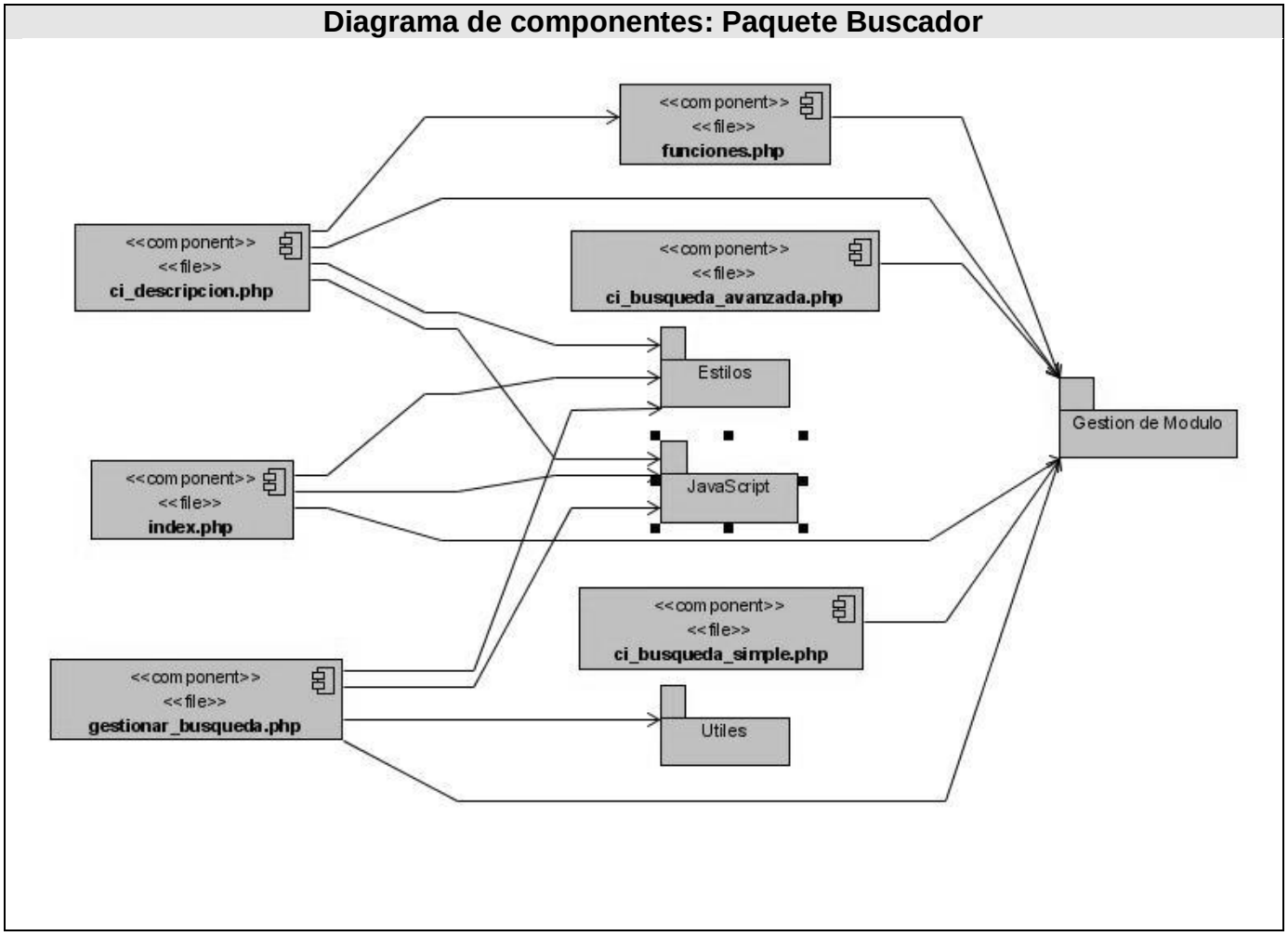

Figura 4.1 Diagrama de componentes: Paquete Buscador

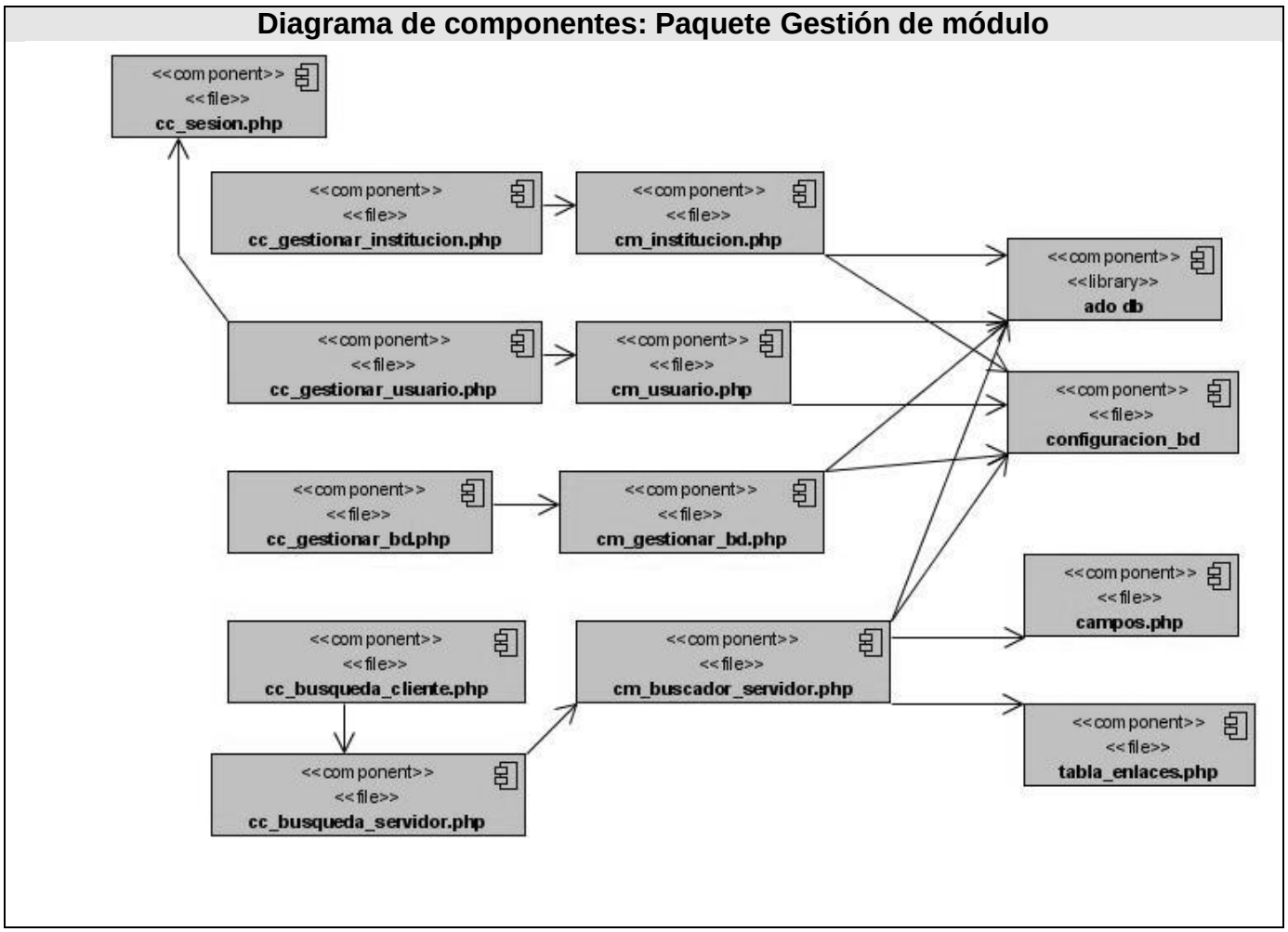

Figura 4.2 Diagrama de componentes: Paquete Gestión de módulo

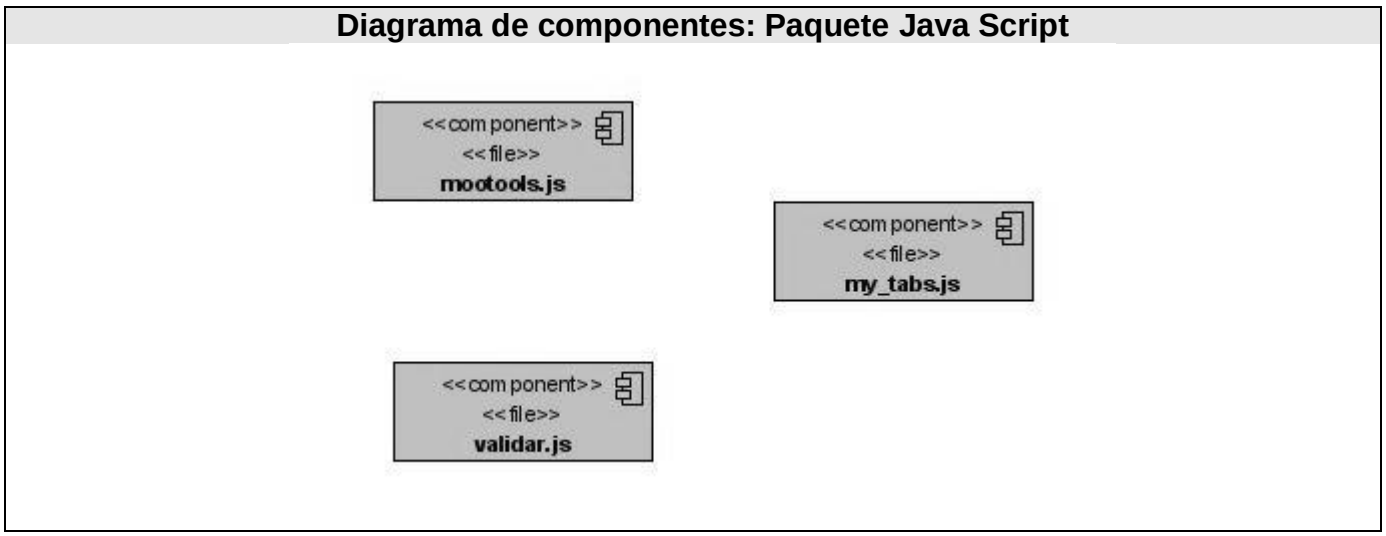

Figura 4.3 Diagrama de componentes: Paquete Java Script

Diagrama de componentes: Paquete Módulo

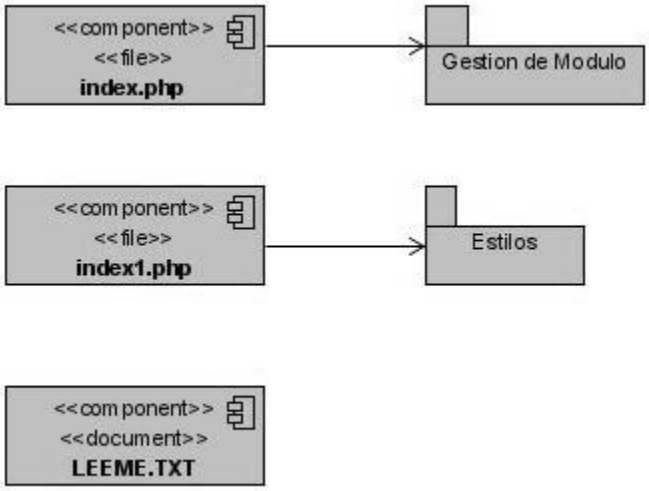

Figura 4.4 Diagrama de componentes: Paquete Módulo

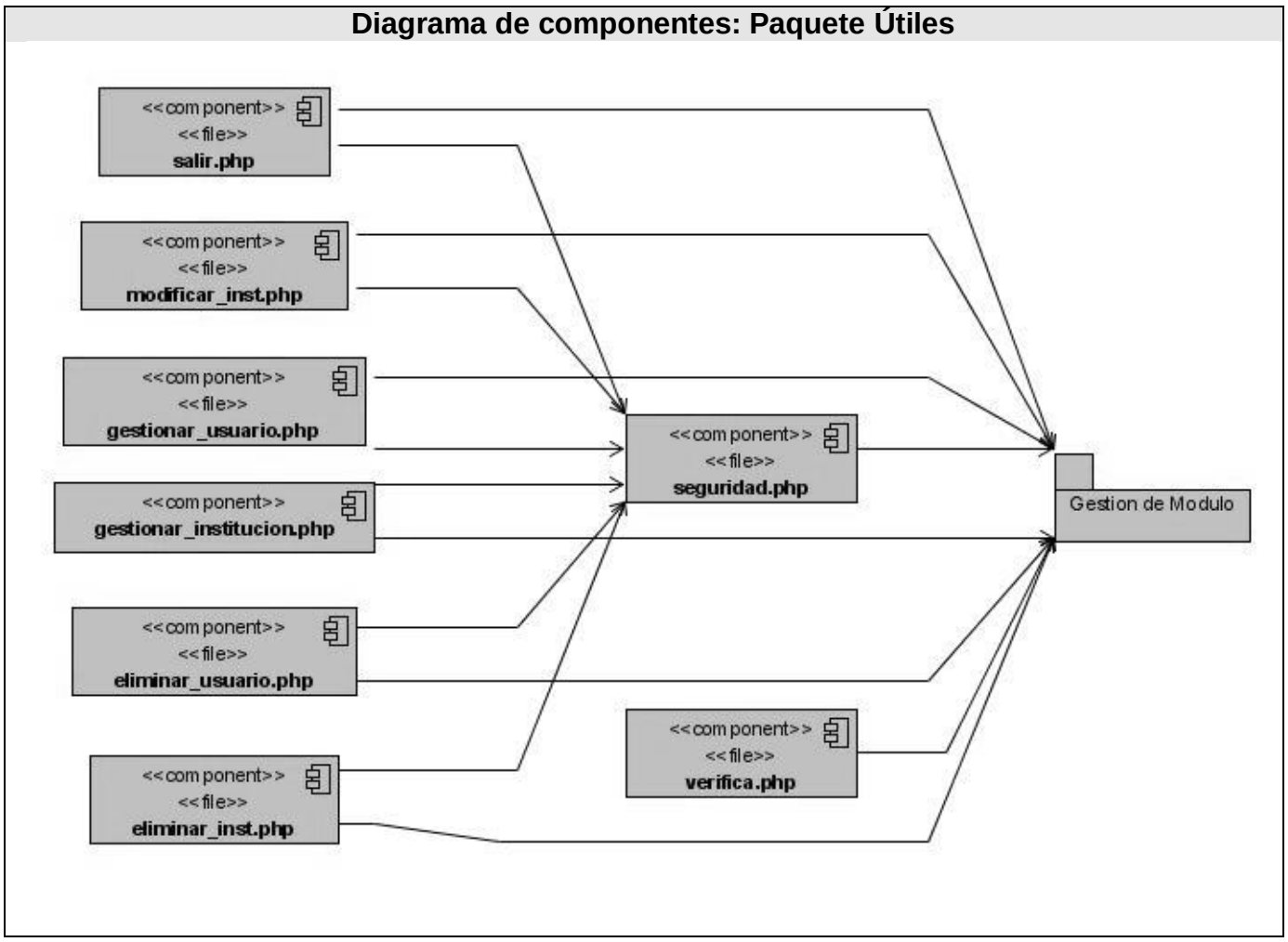

Figura 4.5 Diagrama de componentes: Paquete Útiles

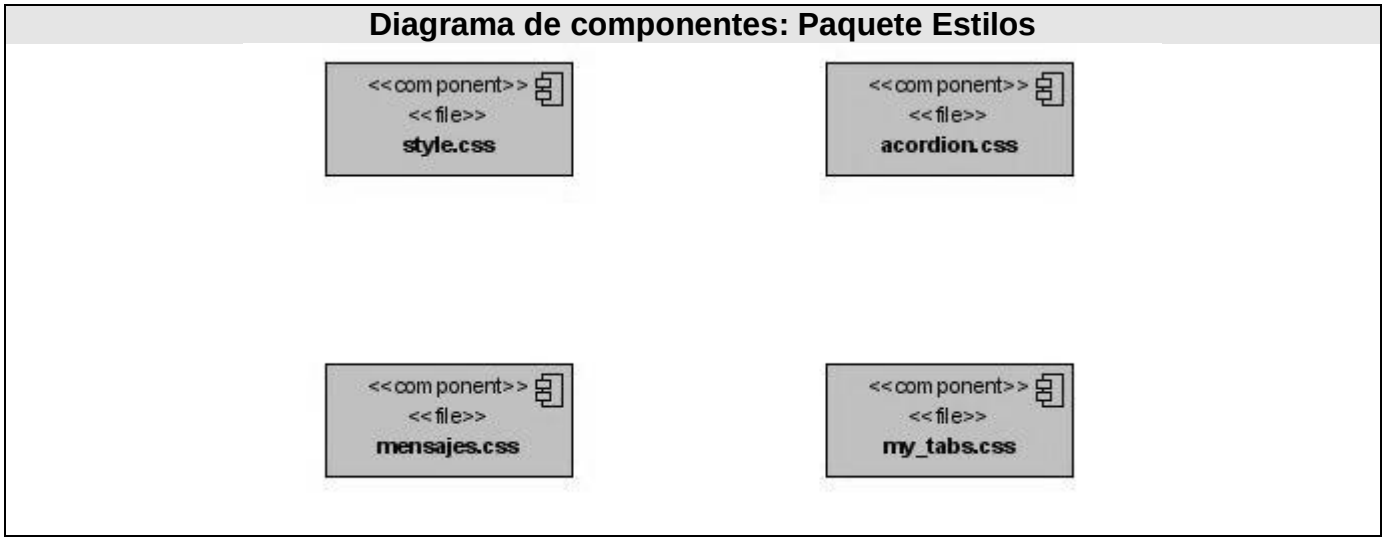

Figura 4.6 Diagrama de componentes: Paquete Estilos

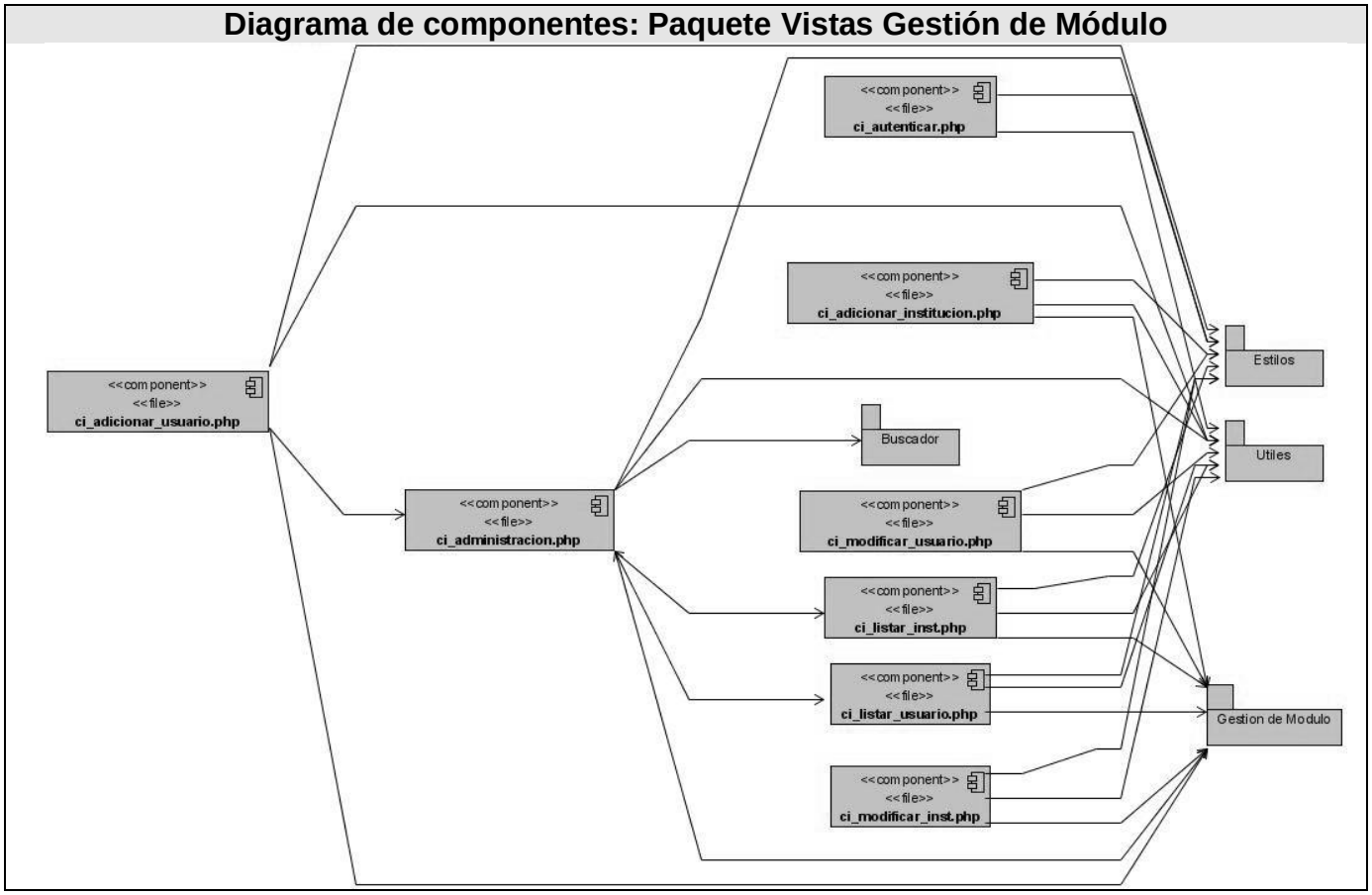

Figura 4.7 Diagrama de componentes: Paquete Vistas Gestión de Módulo

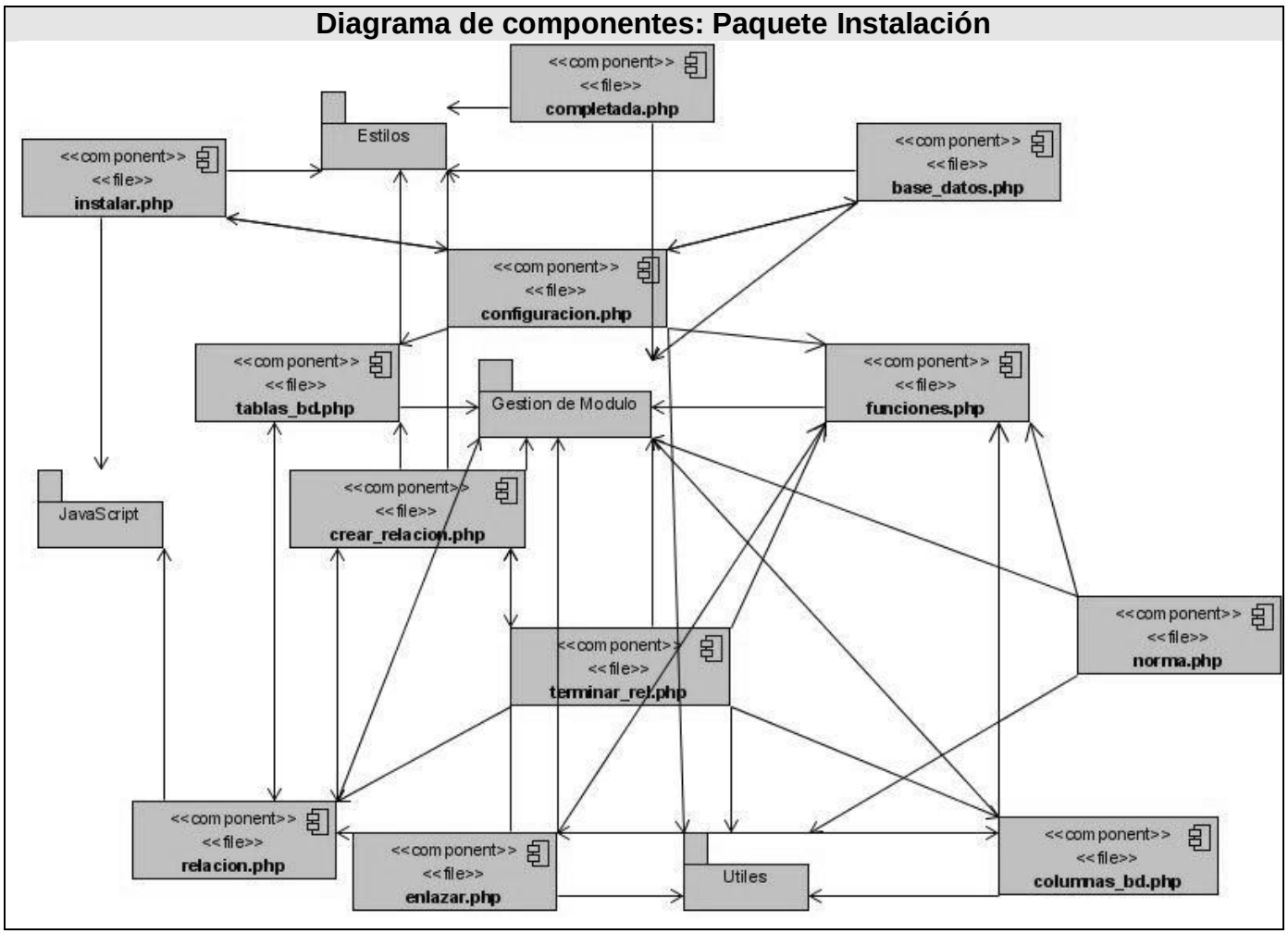

Figura 4.8 Diagrama de componentes: Paquete Instalación

4.3 Diagrama de despliegue

En el diagrama de despliegue se indica la situación física de los componentes lógicos desarrollados. Es decir se sitúa el software en el hardware que lo contiene, además se muestra las relaciones físicas entre los componentes hardware y software en el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software (procesos y objetos que se ejecutan en ellos). Estarán formados por instancias de los componentes software que representan manifestaciones del código en tiempo de ejecución (los componentes que sólo sean utilizados en tiempo de compilación deben mostrarse en el diagrama de componentes).

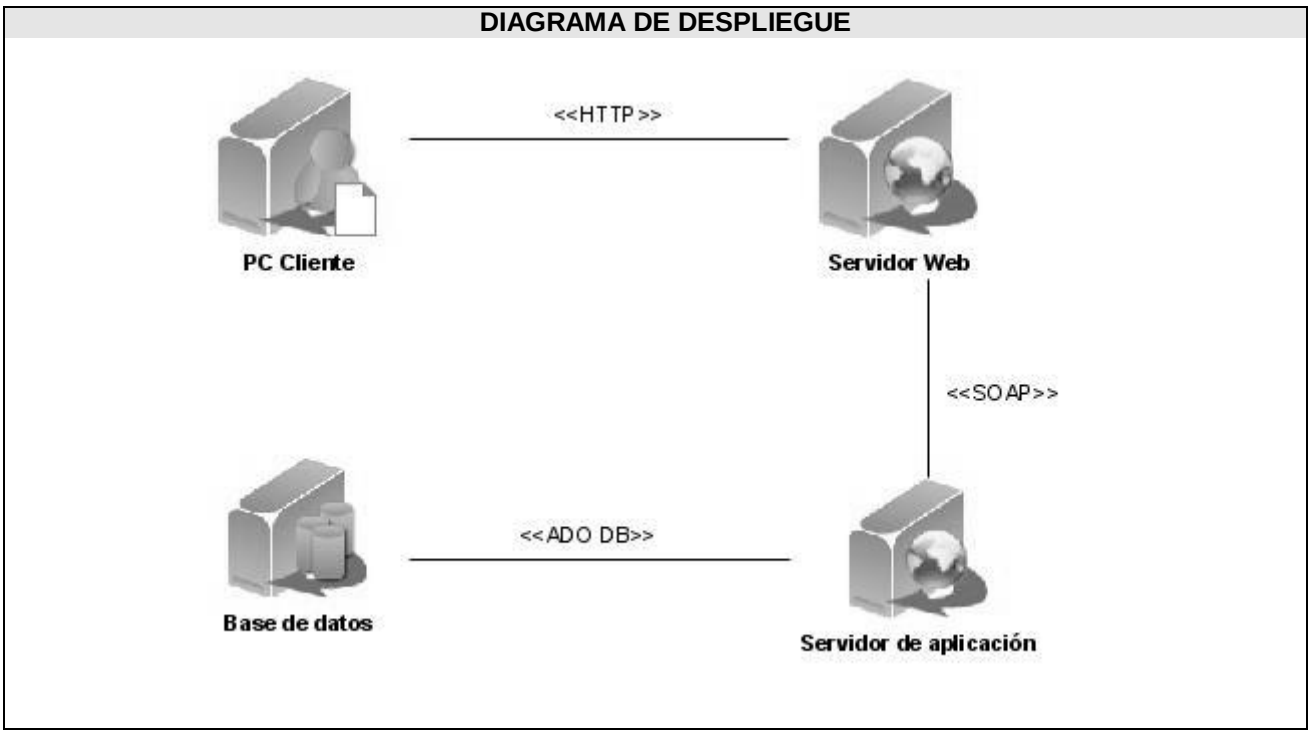

Figura 4.9 Diagrama de despliegue

4.4 Prueba

Las pruebas que se realizan a un software no aseguran que esté libre de errores, sin embargo, nos ayudan a encontrar deficiencias existentes en la implementación, por lo que las pruebas son elementos críticos para la garantía de la calidad del software

El modelo de pruebas describe cómo se prueban los componentes ejecutables en el modelo de implementación con pruebas de integración y de sistema. Además, es la colección de casos de pruebas, procedimientos de prueba y componentes de prueba.

Los casos de prueba especifican una forma de probar el sistema, incluyendo precondiciones, valores de entrada, resultados esperados y poscondiciones.

Caso de Uso Buscar Descripción.

Las pruebas realizadas a este caso de uso son:

- Buscar descripción simple.
- Buscar descripción avanzada.

- Datos incorrectos: criterio de búsqueda con longitud menor.
- Datos incorrectos: campos de búsqueda vacíos.

Buscar descripción simple.

Precondiciones: El usuario deberá seleccionar la opción de búsqueda simple.

Valores de Entrada:

- Introducimos “titulo” en el campo criterio.
- Seleccionamos “Todas” en el campo institución.
- Pulsamos el botón “Buscar”.

Resultados esperados:

- El sistema recupera la información del servidor, y muestra las instituciones que poseen archivos relacionados con el criterio de búsqueda.
- El sistema intenta recuperar información, pero no existen instituciones que posean archivos relacionados con el criterio introducido, por lo que mostrará un mensaje informándole al usuario lo ocurrido.

Poscondiciones:

- El usuario realizó la búsqueda con éxito.
- La búsqueda no tuvo éxito.

Buscar descripción avanzada

Precondiciones: El usuario deberá seleccionar la opción de búsqueda avanzada.

Valores de Entrada:

- Introducimos “titulo” en el campo título.
- Introducimos “Juan Pérez” en el campo productor.
- Seleccionamos “Todas” en el campo institución.

- Pulsamos el botón “Buscar”.

Resultados esperados:

- El sistema recupera la información del servidor, y muestra las instituciones que poseen archivos relacionados con los valores de los campos introducidos.
- El sistema intenta recuperar información, pero no existen instituciones que posean archivos relacionados con los valores de los campos introducidos, por lo que mostrará un mensaje informándole al usuario lo ocurrido.

Poscondiciones:

- El usuario realizó la búsqueda con éxito.
- La búsqueda no tuvo éxito.

Datos incorrectos: criterio de búsqueda con longitud menor.

Precondiciones: El usuario deberá seleccionar la opción de búsqueda simple o avanzada.

Valores de Entrada:

- Introducimos “ti” en el campo criterio.
- Seleccionamos “Todas” en el campo institución.
- Pulsamos el botón “Buscar”.

Resultados esperados:

- El sistema muestra un mensaje indicando que el criterio es demasiado pequeño.
- El proceso de búsqueda termina.

Poscondiciones:

- El usuario no pudo realizar la búsqueda.

Datos incorrectos: campos de búsqueda vacíos.

Precondiciones: El usuario deberá seleccionar la opción de búsqueda simple o avanzada.

Valores de Entrada:

- Dejamos todos los campos de búsqueda vacíos.
- Seleccionamos “Todas” en el campo institución.
- Pulsamos el botón “Buscar”.

Resultados esperados:

- El sistema muestra un mensaje indicando que no se pueden dejar campos vacíos.
- El proceso de búsqueda termina.

Poscondiciones:

- El usuario no pudo realizar la búsqueda.

4.5 Conclusiones

Con la culminación de este capitulo queda implementada la aplicación que se planteo en los objetivos trazados de este trabajo. Se modeló el diagrama de componentes divididos en paquetes para una mayor comprensión, además del diagrama de despliegue. Se desarrollaron las pruebas del software que posibilitó un mejor acabado del módulo de interconexión de archivos.

Conclusiones

Con este trabajo se realizó un estudio de los principales estándares existentes para la descripción de documentos de archivos históricos. Se investigaron aplicaciones web relacionadas con la gestión de archivos históricos. Finalmente y gracias a la investigación y el estudio realizado se logró intercambiar información entre sistemas remotos, posibilitando así la interconexión de distintas instituciones archivísticas, dando cumplimiento a los objetivos planteados.

Gracias al desarrollo de este trabajo los autores adquirieron un conjunto de conocimientos del mundo archivístico. De la importancia que tienen las descripciones de los archivos históricos, así como la posibilidad de búsquedas en distintas instituciones archivísticas.

Se consideran cumplidos cada uno de los objetivos propuestos al inicio de este trabajo, posibilitando la búsqueda y recuperación de archivos históricos basados en estándares internacionales.

Recomendaciones

Como recomendaciones para el desarrollo futuro del proyecto proponemos:

- Profundizar los conocimientos en el área de la archivística a nivel nacional e internacional.
- Lograr un mayor desarrollo en los proyectos de Gestión Archivística en nuestro país.
- Continuar el desarrollo del sistema propuesto a partir de nuevos requisitos que puedan surgir con su utilización.
- Instalar el sistema en todas las instituciones archivísticas del país, para lograr la intercomunicación entre ellos.
- Crear el Portal de Archivos de Cuba el cual utilizará el sistema y de esa forma tener acceso a todos los archivos históricos del país mediantes un solo sitio web.

Glosario de Términos

Descripción archivística**:** La creación de una representación precisa de una unidad de descripción y sus partes componentes, por el proceso de captura, cotejo, análisis, y organización de cualquier información que sirva para identificar el material de archivo y explicar el contexto donde se producen.

Institución archivística**:** instituciones o partes estructurales de ellas que realizan la recepción, conservación y organización de los documentos para su utilización.

Sistema de Archivo**:** Sitio Web para la gestión de archivos históricos.

Unidad de descripción**:** Documento o conjunto de documentos, cualquiera que sea su forma física, tratado como un todo y que como tal contribuye a la base de la única descripción.

Módulo Web: Pequeña aplicación web fácilmente integrable que añaden funcionalidades a su página con una mínima configuración.

IDE: Entorno de Desarrollo Integrado (Integrated Development Enviroment) programa compuesto por un conjunto de herramientas para un programador.

Herramienta CASE: diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.

DTD: descripción de estructura y sintaxis de un documento XML o SGML.

SGML: Lenguaje de marcado generalizado normado, superconjunto de HTML. Lenguaje que define a otros lenguajes con tags, base del HTML utilizado en la Web.

Namespace: Conjunto de nombres en el que todos, son únicos.

Referencias Bibliográficas

"CONCEPTO DE ARCHIVO EN GENERAL Y
DE ARCHIVO HISTÓRICO EN PARTICULAR: SUS FUNCIONES." [Disponible en:
http://www.gipuzkoakultura.net/centenario/conceptodearchivo.htm.
"Extreme Programming." [Disponible en:
http://www.planetacodigo.com/wiki/glosario:extreme_programming
. "Introduction to SOAP." [Disponible en: http://www.w3schools.com/soap/soap_syntax.asp.
. "Módulos web." [Disponible en:http://es.arumeinformatica.es/web/módulo s/.
. "SOAP (Simple Object Access Protocol)." [Disponible en:
http://www.desarrolloweb.com/articulos/1557.php.
. "Visual Paradigm for UML." [Disponible
en:http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_14720_p/.
Contreras, L. N. "ARCHIVÍSTICA, ARCHIVO, DOCUMENTO DE ARCHIVO...
NECESIDAD DE CLARIFICAR LOS CONCEPTOS."
Delio G. Orozco González, V. F. B. "Un sistema de conservación, digitalización, gestión
y socialización de información documental para los
archivos en Cuba."
Flores, J. T. N. "XML aplicado a los instrumentos de descripción archivística
EAD (Encoded Archival Description) Version 2002." 2008, [Disponible en:
http://rayuela.uc3m.es/~nogales/cursos/ead.html.
Fuster, F. "Archivística." [Disponible en: http://es.wikipedia.org/wiki/Archiv%C3%ADstica.
M, J. O. (2005). "EAD: UNA ESTRUCTURA PARA LA DESCRIPCIÓN DEL
PATRIMONIO CULTURAL." 48.
MOS "¿Qué es ASP?" [Disponible en: http://www.aspfacil.com/articulos/278001.asp
Society, E. A. D. W. G. d. l. and o. A. Archivist Descripcion archivistica codificada, directrices de
aplicación. F. H. Tavera.
Valdés, D. P. Los diferentes lenguajes de programación para la web.
Consejo Internacional de Archivos, Norma Internacional General de descripción archivística, Madrid,
2000.

Nogales, T. XML aplicado a los instrumentos de descripción archivística EAD (Encoded Archival Description) Version 2002. 22 de diciembre de 2006 (en línea) http://rayuela.uc3m.es/~nogales/cursos/ead.html .2007

Oxley, J. EAD: UNA ESTRUCTURA PARA LA DESCRIPCIÓN DEL PATRIMONIO CULTURAL. Serie Bibliotecología y Gestión de Información N° 5 julio, 2005. (en línea)
http://eprints.rclis.org/archive/00004780/01/serie_5.pdf. 2007

Surós, A. Pernía, R. Repositorio de Objetos de Aprendizaje para la reutilización de contenidos en plataformas de teleformación. Trabajo para optar por el título de ingeniero informático, Universidad de las Ciencias Informáticas, junio 2006.

Cid, A. Surós, A. Pernía R. Herramienta para la descripción digital de documentos de archivo. Sitio2007. 2007.