Universidad de las Ciencias Informáticas

Facultad 2

Subsistema de Administración Informática y Configuración Inicial del Sistema Xabal Arkheia

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

Autores:

Danima Martínez Paez.

Leydis Herrera Fernández.

Tutor:

Ing. Frank Ernesto Verdecia Rodríguez.

La Habana, Junio de 2013.

"Si los jóvenes fallan, todo fallará. Es mi más profunda convicción que la juventud cubana luchará por impedirlo. Creo en ustedes."

Fidel Castro Ruz

Declaración de Autoría

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

Para que así conste, firmamos la presente a los____ días del mes de_____ del año_____.

Danima Martínez Paez

_____

Firma del Autor

Leydis Herrera Fernández

_____

Firma del Autor

Ing. Frank Ernesto Verdecia Rodríguez

_____

Firma del Tutor

Agradecimientos

A mi mamá y mi papá por siempre estar a mi lado en los buenos y malos momentos de mi vida, por cuidarme, aconsejarme, por hacer de mí la persona que soy hoy, por todo lo que me demuestran a diario y por ser los mejores padres del mundo.

A mi hermana Daineris por sus consejos a pesar de ser menor que yo, por tener un corazón tan grande y demostrarme a diario cuanto me quiere.

A mis hermanas Daimeris y Rosdaimis por apoyarme siempre y mostrar preocupación en todos estos años, también a mi sobrina Lorelita por hacer de mí su manicure y extrañarme tanto cuando no puedo ir a verla.

A mi abuela Juana por darme su amor siempre, por educarme como me educó, por siempre estar a mi lado, cuidarme y hacerme sentir una mejor persona cada año que pasa en mi vida.

A mis abuelos Gladys y Damián Ordener, por su constante preocupación, por su apoyo, sus consejos y por sentirse tan orgullosos de mí.

A mis tíos Marelis, Anén y José por quererme como una hija y demostrarme que siempre están ahí para ayudarme.

A mis tías y tíos, primas y primos por preocuparse incluso cuando las cosas estuviesen bien.

A Leydis, que aunque de vez en cuando estemos fajadas, es la amiga y compañera de tesis que todos quisieran tener.

A Yordan por ser uno de nuestros revisores y preocuparse por que todo saliera bien en cada uno de los cortes.

A mi papito por confiar en mí y ser la persona que es, por demostrarme que sí se puede aunque se pase por malos momentos, pues siempre se ve la luz al final del camino, por su apoyo incondicional día a día y ser para mí un ejemplo a seguir, por ayudarme como lo ha hecho siempre, por su amor y por sobre todas las cosas por ser el hombre de mi vida.

A todos muchas gracias. **Danima Martínez Paez.**

Agradecimientos

A mi madre, por estar siempre y apoyarme en los buenos y malos momentos de mi vida, sin ella no hubiese podido triunfar.

A mi padre que siempre estuvo orgulloso de mi sueño de ser ingeniera informática.

A toda mi familia, que la adoro y les agradezco a todos por preocuparse y estar pendientes de mí.

A mi novio Yordan, que conocerlo es lo más hermoso que me ha podido pasar durante la carrera, gracias por ser quien eres mi amor.

A la familia de mi novio, especialmente a mis suegros Denia y Jorge gracias por contar con su apoyo.

A Danima, que la considero una gran amiga, gracias por tener paciencia y haber trabajado juntas hasta el final.

A mi tutor Frank, por contar con su guía y apoyo para lograr el triunfo de este trabajo.

A todos mis compañeros de clases, amistades de la UCI siempre los recordaré.

A los compañeros del cuarto de mi novio, a Albin, Sabrina, Felipe, Maylevis, Yosbel, Dairelys.

Leydis Herrera Fernández.

Dedicatoria

Primero que todo a mi esposo que sin él no hubiese sido posible seguir adelante, Te Amo mi amor.

A mi mamá, mi papá y mis hermanas por apoyarme en todo y siempre sentirse orgullosos de mí.

A mis abuelos por ser siempre tan buenos conmigo.

A mi familia por su preocupación en todo momento.

Al profesor de Matemática Fernando Rafael Rodríguez Marzo por ayudarme como lo hizo y por confiar en mí.

A mis amigos, que siempre han estado apoyándome y guiándome con sus consejos.

Danima Martínez Paez.

Dedicatoria

Dedico este trabajo a:

Mi madre, mi padre y mi hermano ya que sin ellos no hubiese logrado este sueño, por depositar toda su confianza en mí y por estar siempre a mí lado.

A mi novio Yordan, por ayudarme en todo momento y brindarme tanto amor cuando más lo necesité, nunca te voy a olvidar, Te amo mi bb.

A mis abuelos que están muy orgullosos de mí.

A mis tíos, primos, a toda mi familia este trabajo se lo dedico también a ustedes.

Todo este gran esfuerzo es para ustedes.

Leydis Herrera Fernández.

Resumen

El sistema Xabal Arkheia permite la informatización de los principales procesos en la gestión de los archivos históricos. Además, brinda un conjunto de acciones y medidas que deben considerarse y establecerse para garantizar un adecuado acceso, uso, mantenimiento, protección y conservación de los documentos de archivo.

La administración informática es una herramienta que permite la toma de decisiones al observar eventos que no son comunes que ocurran a diario, además de gestionar el acceso a la información de los usuarios que trabajan con el sistema.

En el presente trabajo se especifica el desarrollo del Subsistema de Administración Informática y Configuración Inicial del Sistema Xabal Arkheia, el cual está compuesto por dos módulos. El módulo de Administración Informática permite la gestión de usuarios, auditoría de las acciones de los usuarios, la administración de parámetros generales y la gestión de recursos de memoria; el módulo de Configuración Inicial, encargado de la configuración del cuadro de clasificación y la estructura de ubicación física, logra que el sistema se adapte a las características de cada institución donde se utilice.

Palabras Claves

Administración informática, auditoría, configuración, nomencladores, permiso, rol, reporte, subsistema, usuario.

Índice

Introducción

Antes de que la Archivología[1] fuera considerada como una ciencia, las instituciones solo se dedicaban a conservar los documentos y expedientes, agrupándolos sin hacer uso de normas específicas, solo se tenía en cuenta la fecha de creación para su conservación y usualmente lograr su organización no era un objetivo preferencial.

A partir de la Segunda Guerra Mundial no solo existían los archivos históricos exclusivos de estudio, sino también los administrativos, los privados y los de empresa, a los que se le dedicaban mayor cuidado y protección, siendo necesario establecer criterios, aplicar normas homogéneas y establecer etapas que permitieran la organización gradual y sistemática de estos archivos.

En países como Francia y España la administración de archivos se centra en los archivos administrativos o de oficina y los históricos. En países de Europa del Este, se acostumbraba a guardar los archivos en un fondo estatal único, compuesto por los archivos provisionales y los de estado o históricos. En los Estados Unidos, se realiza un seguimiento de los archivos desde su creación hasta su destrucción o conservación permanente, a través del Servicio Nacional de Archivos y Documentos. (1)

En Venezuela durante el período colonial, los documentos de valor histórico, político, económico y social, se depositaban en las escribanías públicas, en los despachos eclesiásticos y en los archivos particulares. Sin embargo en 1836 se creó la institución del Registro dividiendo estas oficinas en dos partes, en el Registro y en el Archivo Público. La actividad archivística venezolana, está normada por la Ley de Archivo Nacional desde 1945, siendo actualizada en el 2001 con la aprobación de la Ley Orgánica de Administración Pública para establecer entre otros aspectos, las normas básicas sobre los archivos y registros de la Administración Pública. (2)

En Cuba, no es hasta el Triunfo de la Revolución que se implementa una política cultural orientada conscientemente a la conservación del patrimonio histórico documental de la nación y a la potenciación de la actividad archivística. Durante el año 1959 se crean los archivos históricos regionales, transformándose con la división político–administrativa acaecida en 1976, en archivos históricos municipales y provinciales, adscritos a la Academia de Ciencias de Cuba. Ambas instituciones comienzan a formar parte de una Red Nacional, quedando subordinadas al Archivo Nacional hasta 1985 en que ocurre una descentralización (unos pertenecerían a Cultura, algunos a Justicia y otros a las oficinas de los Historiadores o Conservadores de la Ciudad) aunque metodológicamente continuarían siendo atendidos por el Archivo Nacional.

Con el transcurso del tiempo y el desarrollo de la tecnología, se han logrado sofisticados sistemas que contribuyen a una mayor eficiencia en la gestión de archivos históricos. Los

[1] Disciplina que estudia los archivos en todos sus aspectos (58).

sistemas informáticos para la gestión de archivos históricos brindan la posibilidad de tener un control de los datos asociados a sus archivos, eliminando el contacto directo de las personas con el archivo físico, disminuyendo así el deterioro de los documentos. La organización física y lógica en las entidades varían según sus necesidades de organización. Sin embargo cada organización realiza un control de su información de forma similar, pero con características propias, como son: el acceso a la información y la gestión de valores nomenclados para lograr una homogeneidad entre todas las personas que trabajan en la institución.

En la actualidad el tema de la seguridad informática es muy importante a la hora de implementar un sistema informático, estar debidamente asegurado contra ataques cibernéticos a la institución permite no dejar escapar información sensible. Los lugares que no presenten un correcto mecanismo de acceso a la información pueden llevar a grandes pérdidas de datos, poniendo en riesgo su patrimonio histórico. Según una encuesta de la empresa Kroll Ontrack[2], asegura que para un 60% de las empresas el impacto económico de la pérdida de datos fue igual o superior a 50.000 dólares, y para un 3%, de más de 1 millón de dólares.

El centro CENIA[3] de la facultad 1 de la Universidad de las Ciencias Informáticas realizó un producto para el Archivo General de la República Bolivariana de Venezuela (ArchivenHIS, aplicación web para gestionar los documentos históricos que se custodian en los archivos), esta solución ha sido de interés para varias empresas tanto nacionales como internacionales debido a las funcionalidades que ofrece. Sin embargo, no es personalizable a los requerimientos de nuevos clientes por estar desarrollado de acuerdo a las necesidades específicas del cliente venezolano. Entre sus principales deficiencias es posible encontrar que ArchivenHIS no presenta funcionalidades de administración capaz de configurar los parámetros generales de la aplicación, la gestión de permisos, configuración del cuadro de clasificación y la internacionalización del sistema. Esto conlleva a que se pierdan posibles contratos o demoras en la entrega de la solución solicitada porque se deben desarrollar las funcionalidades o características que necesita el nuevo cliente.

El centro ISEC de la facultad 2 de la Universidad de las Ciencias Informáticas como ente productor de software, ha decidido desarrollar un sistema genérico que se inserte en el mercado de la gestión de archivos: el Sistema Xabal Arkheia, software cuyo propósito fundamental, es el de servir como plataforma de gestión de archivos históricos, que con un esfuerzo mínimo del equipo de desarrollo se adapte a las características propias de cada institución, ante esta situación surge el siguiente **problema a resolver:** ¿Cómo proveer un mecanismo para la restricción de acceso y configuración de parámetros generales del Sistema Xabal Arkheia?

Con el objetivo de guiar el curso de la investigación se tomó como **objeto de estudio** los mecanismos de restricción de acceso y configuración en sistemas de gestión de archivos.

[2] Empresa especializada en el uso en la recuperación y administración de datos.
[3] Centro de Informatización Universitaria.

Esta investigación tiene como **objetivo general** desarrollar el Subsistema de Administración Informática y Configuración Inicial para administrar los parámetros generales, la gestión de usuarios y la configuración de los subsistemas “Incorporación y Organización Documental” y “Almacenamiento y Conservación” del Sistema Xabal Arkheia.

Se enmarca el **Campo de acción** en los procesos de administración informática y configuración de parámetros generales en sistemas de gestión de archivos.

Para el cumplimiento del objetivo general se definen los siguientes **objetivos específicos:**

- ✓ Determinar la fundamentación teórica de la investigación.
- ✓ Definir el Subsistema de Administración Informática y Configuración Inicial del Sistema Xabal Arkheia.
- ✓ Validar los requerimientos de software definidos en el Subsistema de Administración Informática y Configuración Inicial del Sistema Xabal Arkheia.

Las **tareas de investigación** propuestas para dar cumplimiento a los objetivos trazados son:

- ✓ Analizar los principales conceptos de gestión de archivo y administración informática y definir los que guiarán el desarrollo de la investigación.
- ✓ Seleccionar sistemas de gestión de archivo como referencia para el desarrollo de la investigación.
- ✓ Caracterizar las herramientas, tecnologías y metodología de desarrollo seleccionadas por el equipo de arquitectura del Sistema Xabal Arkheia.
- ✓ Definir las pautas de codificación a utilizar en el desarrollo del Subsistema de Administración Informática y Configuración Inicial del Sistema Xabal Arkheia.
- ✓ Definir los métodos de pruebas para la validación de sistemas web.

Para apoyar el desarrollo de la investigación se emplean los siguientes métodos científicos:

Métodos Empíricos.

Los métodos empíricos posibilitan revelar las relaciones esenciales y las características fundamentales del objeto de estudio, accesibles a la detección de la percepción, a través de procedimientos prácticos con el objeto y diversos medios de estudio. La investigación empírica permite al investigador hacer una serie de investigación referente a su problemática, retomando experiencia de otros autores, para de ahí partir con su exploración, también conlleva efectuar el análisis preliminar de la información, así como verificar y comprobar las concepciones teóricas. (3) (4) (5)

- ✓ **Entrevista:** Es la comunicación interpersonal establecida entre el investigador y el sujeto de estudio a fin de obtener respuestas verbales a las interrogantes planteadas sobre el problema propuesto.

A lo largo de la presente investigación se realizan varias entrevistas con los clientes para conocer los requerimientos y funcionalidades que desean tener en el Subsistema de Administración Informática y Configuración Inicial del Sistema Xabal Arkheia. **(VerAnexo I).**

Métodos Teóricos.

Los métodos teóricos potencian la posibilidad de realización del salto cualitativo que permite ascender del acondicionamiento de información empírica a describir, explicar, determinar las causas y formular la hipótesis investigativa.

- ✓ **Histórico-Lógico:** Este método facilita el estudio de las etapas por las que ha transcurrido la gestión de archivos. Se obtiene un conocimiento histórico de forma cronológica desde su surgimiento hasta nuestros días, tanto a nivel nacional como internacional, así como las tendencias actuales.
- ✓ **Modelación:** Este método permite realizar un modelo informático de la vida real, sirviendo de guía en cada una de las fases del ciclo de vida del proyecto, con la ayuda de la metodología de desarrollo seleccionada.
- ✓ **Analítico-Sintético:** Este método permite realizar una definición de los conceptos principales de la investigación a partir de distintas fuentes bibliográficas.

El presente trabajo de diploma consta de 3 capítulos donde se describe todo el proceso de la investigación:

Capítulo 1 *Fundamentación Teórica:* En este capítulo se profundizarán los conceptos fundamentales del negocio para el desarrollo del Subsistema de Administración Informática y Configuración Inicial del Sistema Xabal Arkheia. El análisis de los sistemas de gestión de archivos a nivel nacional e internacional seleccionados como referencia. Además, la selección del modelo de control de acceso en la implementación del sistema, así como, la caracterización de las herramientas y metodología de desarrollo seleccionadas por el grupo de arquitectura del proyecto, sirviendo como base para el desarrollo de la investigación.

Capítulo 2 *Análisis y Diseño:* En este capítulo se realizará el análisis y diseño del subsistema, mediante la descripción de los casos de uso, diagramas de interacción y diagramas de clases, además de la propuesta de solución del subsistema, permitiendo una guía para su desarrollo.

Capítulo 3 *Implementación y Pruebas:* En este capítulo se realizará el desarrollo del subsistema, guiado por los artefactos generados en el análisis y diseño. Como resultado se obtienen el diagrama de componentes, el diagrama de despliegue y la selección de los métodos de pruebas, para realizar una evaluación del estado de calidad de la solución.

Capítulo 1

1 Fundamentación Teórica.

En este capítulo se profundizarán los conceptos fundamentales del negocio para el desarrollo del Subsistema de Administración Informática y Configuración Inicial del Sistema Xabal Arkheia. El análisis de los sistemas de gestión de archivos a nivel nacional e internacional seleccionados como referencia. Además, la selección del modelo de control de acceso en la implementación del sistema, así como, la caracterización de las herramientas y metodología de desarrollo seleccionadas por el grupo de arquitectura del proyecto, sirviendo como base para el desarrollo de la investigación.

1.1 Conceptos Fundamentales.

Para una mejor comprensión del contexto en el que se desarrolla la investigación, se definen los conceptos principales de la administración informática.

Subsistema: permiten dividir el sistema entero en partes más manejables. Cada subsistema tiene aspectos del sistema con propiedades comunes, siendo un conjunto de elementos ordenados para cumplir con un propósito, estas partes deben cumplir con determinadas condiciones hasta que se complementen formando el sistema. Mediante este concepto se definieron las características y funcionalidades que tendrá la solución propuesta. (6) (7)

Usuario: es la persona o sistema informático que trabajará con el sistema, asignándole uno o varios roles, dependiendo del cargo que tenga en la institución.

Permiso: según el diccionario de la Real Academia: *"Es la autorización que le es asignada a una persona para realizar una determinada acción"*. Los permisos se les asignarán a los usuarios para restringir el nivel de acceso en el sistema.

Rol: es un conjunto de permisos y privilegios agrupados, mediante el cual se asigna acceso, a los usuarios, a las funcionalidades del sistema, dependiendo de sus responsabilidades.

Auditoría: es el proceso de recoger, agrupar y evaluar evidencias para determinar si un sistema de información salvaguarda el activo empresarial, mantiene la integridad de los datos, lleva a cabo eficazmente los fines de la organización y utiliza eficientemente los recursos. (8)

Nomencladores: los nomencladores son grupos de valores significativos para el negocio de la aplicación, compuesto por una descripción y un identificador. Estos valores usualmente se utilizan para ser seleccionados de una lista desplegable o múltiple, disminuyendo los errores de tipiado de palabras por el usuario.

Configuración: son los datos que determinan el valor de las variables de un programa o Sistema Operativo, valores constantes que no se modifican sistemáticamente, permitiendo establecer valores que se apliquen a distintas reglas del negocio.

Administración Informática: es un conjunto de actividades coordinadas que combinan e implementan recursos y capacidades para arrojar un resultado, el cual crea valor directo o indirectamente a un cliente externo o patrocinador. (9)

1.1 Estado del Arte.

El estado del arte es la etapa en la que se recopila información acerca del tema abordado en la investigación, permitiendo sistematizar la información bibliográfica consultada.

En este epígrafe se analizarán las características de la gestión documental y sistemas gestores de archivos tanto en el ámbito nacional como internacional, para conocer las fortalezas y debilidades, que presentan, en el área de la administración informática.

1.1.1 Gestión Documental.

Es el conjunto de normas, procedimientos y tecnologías que se utilizan para administrar la información dentro de una organización. (10)

Entre las características de un Gestor Documental se tienen las siguientes:

- ✓ Presentar una interfaz intuitiva con un entorno de interfaz gráfica, sencillo y no requerir largos períodos de aprendizaje.
- ✓ Realizar una búsqueda precisa de la información.
- ✓ El software de Gestión Documental es capaz de gestionar, clasificar, pasar los OCR[4] incluso en sus documentos adjuntos para poder realizar después búsquedas precisas.
- ✓ Resguarda la información almacenada y brinda seguridad en la integridad de los datos.

La implementación de un software de Gestión Documental brinda beneficios como:

- ✓ Ahorro monetario tangible, rapidez y precisión.
- ✓ Protección en el tratamiento de documentos y en el acceso a la información, resguardándolo de amenazas internas y externas.
- ✓ Reduce los tiempos de clasificación de los documentos en su búsqueda y tratamiento.
- ✓ Ahorro de espacio físico en almacenar los documentos, menos personas para realizar las mismas tareas.

1.1.2 Sistemas Gestores de Archivos seleccionados.

A continuación se realiza un análisis de los puntos fundamentales de la administración informática en los sistemas elegidos:

Software especializado para la administración automatizada de documentos (SCAV).

El sistema SCAV, es un software especializado para la Gestión Documental, diseñado para manejar grandes volúmenes de información. Es una herramienta que ofrece la oportunidad de

[4]Software de Reconocimiento Óptico de Caracteres aplicado a la Gestión Documental.

manipular y controlar electrónicamente documentos sin importar el formato (Word, Power Point, Email, Video, Página Web, entre otros). Es un sistema amigable que permite ser adaptado a las necesidades de cada empresa mediante las opciones de diseño y configuración. Permite crear, capturar, organizar, rastrear, distribuir y almacenar todos los contenidos digitales de una empresa. (10)

Dentro de las facilidades que brinda SCAV se encuentran:

- ✓ Permitir crear un sistema de archivo perdurable en el tiempo y regido por un manual de normas y procedimientos.
- ✓ Posibilitar ampliar el espacio de almacenamiento del archivo.
- ✓ Disponer de un inventario del contenido del archivo.
- ✓ Reducir drásticamente los costos de impresión y fotocopiado.
- ✓ Permitir administrar y controlar eficientemente los procesos que involucran grandes volúmenes de información en papel.

SCAV opera en los centros de documentación, administración de expedientes de clientes y recursos humanos, procesos de pago y en las solicitudes de crédito. Brinda ventajas como:

- ✓ La capacidad para múltiples catálogos de datos, diseñados directamente por el usuario a través de Interfaces gráficas y suministro de modelos para aplicaciones comunes.
- ✓ La capacidad para manejar cantidad ilimitada de documentos en línea sin límite de páginas por documento.
- ✓ El correo electrónico incluido en la aplicación y compatibilidad con los correos electrónicos más populares. Envío y recepción de correo electrónico (Internet) directamente desde la aplicación.

AvilaDoc.

Aplicación web desarrollada por la Empresa Nacional de Software (DESOFT), con el objetivo de agilizar el trabajo con la documentación en empresas y entidades. Creada con tecnologías libres con una base de datos centralizada. Está destinada a la gestión, tramitación y resguardo de archivos electrónicos y digitales; facilitando la búsqueda o recuperación de información de forma rápida y sencilla. (11)

Permite controlar la entrada y salida de documentos, así como gestionar su tránsito interno, constituyendo un archivo digital que agiliza las tareas de registro, búsqueda, reproducción y distribución de los documentos. Este sistema permite ser enlazado a Microsoft Outlook para generar automáticamente tareas a cumplir, notificaciones e información asociada al documento. Ofrece un historial del documento desde que fue creado hasta su eliminación.

Incorpora el fichado de la documentación en un expediente como punto de partida, simulando el flujo de la documentación en una entidad. Permite adjuntar los documentos digitalizados. El

objetivo con el cual fue implementado, abrió las puertas al desarrollo y comercialización de este producto. Con la implementación del sistema en las distintas entidades se logra:

- ✓ La obtención de un archivo digital centralizado con información única, congruente y confiable.
- ✓ Centralizar la información y disponer de ella a todos los niveles.
- ✓ Obtener uniformidad en el proceso de Gestión Documental.
- ✓ Permitir acceso a la información necesaria según los niveles de acceso que debe tener cada funcionario y trabajador dentro del proceso de la Gestión Documental.
- ✓ Agilizar la toma de decisiones con la implantación de un proceso de Gestión Documental.
- ✓ Definir el flujo de información legible y dócil.

Nuxeo.

Es una plataforma de ECM[5] basada en software libre, extensible, configurable y modular. Sus implantadores fueron Francia/París representando soluciones en diferentes lugares pertenecientes a Boston, Nueva York y San Francisco. También brinda soluciones modulares como Nuxeo Document Management (DM) y Digital Asset Management (DAM). Ofrece una plataforma empresarial de fuente abierta para la administración de contenido que permite a arquitectos y desarrolladores, crear, desplegar y ejecutar con facilidad las mejores aplicaciones de negocio. Brinda mayor flexibilidad de manera que su aplicación de gestión de contenidos responda a la perfección de sus necesidades técnicas y corporativas.

Incluye diversas soluciones de administración de documentos y activos digitales Presenta módulos funcionales para responder a las necesidades genéricas del negocio: gestión de documentos (DM) para la gestión de contenidos, colaboración social para un proyecto centrado en el usuario y la gestión de contenidos, la gestión de activos digitales (DAM) para la gestión de contenido rico en medios de comunicación (incluyendo imágenes, audio y video), y la sentencia del Marco de Gestión (CMF) para la gestión de casos. (12)

Alfresco.

Es el líder en el mercado de código abierto para la gestión de contenidos empresariales. Fundada en el año 2005 por un grupo de veteranos de la gestión documental, que incluye al co-fundador de Documentum (empresa de gestión de contenidos), John Newton y John Powell. Alfresco está ubicada en Londres, con representación en España. Utiliza una arquitectura flexible para proveer gestión de documentos, gestión de contenido web y software colaborativo a más de 700 clientes empresariales a nivel mundial.

Posee dos versiones: la versión *Community,* es gratuita pero no ofrece garantías y la versión *Enterprise* que ofrece soporte, pero hay que pagar la licencia. Hoy en día, Alfresco ya cuenta

[5]Enterprise Content Management (Gestión de Contenidos Empresariales).

con su cuarta versión principal y ha sido descargada más de 3,5 millones de veces por desarrolladores y organizaciones. Se ha demostrado su escalabilidad en empresas con millones de documentos y decenas de miles de usuarios. Funciona en servidores físicos, servidores virtuales, nubes privadas, EC2[6] y la nube pública. Presta servicios tales como: gestión de documentos, gestión de registros, administración, estándares abiertos y seguridad. (13)

Athento.

Solución de software de Gestión Documental desarrollado por la empresa malagueña Yerbabuena, radicada en Silicon Valley, EEUU, en la que existe la eficiencia, seguridad, movilidad e inteligencia. Integra tecnología de última generación, extrae contenido de forma automática de cualquier documento. Permite clasificar los documentos en la carpeta correcta sin intervención humana y deduciendo el tipo de documento dentro de los tipos pre-establecidos. También reduce de 8 minutos a 5 segundos los tiempos de extracción de datos en un documento. Integra firma digital utilizando los certificados más estándares e incorporando los procesos de firma a los flujos de trabajo, tales como revisión y firma de contratos por varias partes. (14)

Athento presenta las siguientes funcionalidades de administración informática:

- ✓ Multiusuario.
- ✓ Acceso al gestor mediante login y password.
- ✓ Soporte para Single Sign-On.
- ✓ Soporte para LDAP y Base de Datos propia para dar de alta usuarios desde LDAP.
- ✓ Creación de usuarios y grupos.
- ✓ Creación de roles.
- ✓ Administración de permisos.
- ✓ Estadísticas en tiempo real de uso del sistema.
- ✓ Pista de auditoría (logs de eventos).

Greenstone.

Es un paquete de software desarrollado por el proyecto Biblioteca Digital de Nueva Zelanda en la Universidad de Waikato para crear y distribuir colecciones de bibliotecas digitales, cuya salida presenta una apariencia típica. Este producto se distribuye en colaboración con la UNESCO[1] y la ONG HumanInfo[1].. Brinda una notable flexibilidad, permitiendo su reconfiguración a la medida de las necesidades y objetivos de cada institución. Ofrece una nueva forma de organizar la información y publicarla en Internet o en CD ROM. Es de código

[6]Elastic Compute Cloud, es diseñado para facilitar a los desarrolladores recursos informáticos escalables y basados en web.

abierto, software multilingüe, publicado bajo los términos de la Licencia Pública General GNU[7]. La interfaz completa de Greenstone y toda la documentación, está disponible en Inglés, francés, español, ruso y kazajo y en muchos otros idiomas. (15)

ArchivenHIS.

El sistema ArchivenHIS fue creado en la Universidad de las Ciencias Informáticas para el Archivo General de la República Bolivariana de Venezuela, solución que ha sido de interés para varias empresas tanto nacionales como internacionales debido a las soluciones que ofrece. Sin embargo, no es personalizable a los requerimientos de nuevos clientes por estar desarrollado de acuerdo a las necesidades específicas del cliente venezolano. Está compuesto por varios módulos que permiten realizar las actividades comunes en los archivos históricos tales como descripción de documentos, creación de la estructura física de organización de la documentación, creación de la estructura lógica de organización de la documentación mediante los niveles de descripción y solicitud de documentos físicos.

ArchivenHIS no presenta funcionalidades de administración informática, solo presenta una gestión básica de usuarios con permisos ya definidos previamente y no gestionables. Además no posee una administración de parámetros generales de la aplicación como son nomencladores, valores de configuración y auditoría de las acciones de los usuarios.

¿Qué aporta el estudio de los sistemas seleccionados?

El estudio de estos sistemas brinda una primera visión de las funcionalidades necesarias que debe tener la administración informática en la realización de sistemas de gestión de archivos. A continuación se realizará una clasificación de las funcionalidades atendiendo a conceptos de administración informática: gestión de usuarios, auditoría, configuración general y administración de recursos.

Sistemas.	Gestión de usuarios.	Auditoría.	Configuración general.	Administración de recursos.
SCAV			Capacidad para múltiples catálogos.	Amplía el espacio de almacenamiento de los archivos.
AvilaDoc	Limita el acceso al sistema a partir de niveles de acceso.			

[7]Es la licencia más ampliamente usada en el mundo del software y garantiza a los usuarios finales la libertad de usar, estudiar, compartir y modificar el software.

Nuxeo		Muestra el historial del trabajo con el documento.		
Alfresco	Integración con LDAP.	Historial de las versiones de los documentos.		
Athento	Multiusuario. Soporte para Single Sign-On. Creación de usuarios y grupos. Soporte para LDAP. Creación de roles.	Traza de las acciones de los usuarios. Estadísticas de uso del sistema en tiempo real.		
Greenstone			Presenta una internacionalización de la interfaz en distintos idiomas como: Inglés, francés, español, entre otros.	
ArchivenHIS	Creación de usuarios a partir de roles predefinido.			

Tabla 1 Clasificación de las funcionalidades atendiendo a conceptos de administración informática.

1.1.3 Seguridad Informática

La Seguridad Informática es un conjunto de características y condiciones de sistemas de gestión de la información, para garantizar su confidencialidad, integridad y disponibilidad.

La autorización es el procedimiento para determinar si el usuario o proceso previamente identificado y autenticado tiene permitido el acceso a los recursos. Se implementa con uno de los siguientes modelos de control de acceso:

Control de Acceso Obligatorio (MAC): en este modelo es el sistema quien protege los recursos, comparando las etiquetas del sujeto que accede frente al recurso accedido, o sea, la autorización para que un sujeto acceda a un objeto depende de los niveles de seguridad que tenga, ya que estos indican, qué permiso de seguridad tiene el sujeto y el nivel de sensibilidad del objeto. (16) (17) (18)

Control de Acceso Discrecional (DAC): este modelo se ha venido usando de forma abundante en sistemas operativos de propósito general, en la mayoría de los sistemas de base de datos y en sistemas de comunicaciones de propósito comercial. Un usuario bien identificado, por lo general el creador o propietario del recurso, decide cómo protegerlo estableciendo cómo compartirlo, mediante controles de acceso impuestos por el sistema. Lo fundamental es que el propietario del recurso puede cederlo a un tercero. (16) (17) (18)

Control de Acceso Basado en Roles (RBAC): este modelo es un intento de unificar los modelos clásicos DAC y MAC, logrando un sistema que impone el control de acceso, pero sin las restricciones rígidas impuestas por las etiquetas de seguridad. (16) (17) (18)

En este modelo los usuarios son asignados a uno o varios roles, mientras que los permisos y privilegios se le asignan a estos roles. Por tanto, las políticas de control de acceso basado en roles regulan el acceso de los usuarios a la información en términos de sus actividades y funciones de trabajo (roles), representándose así de forma natural la estructura de la organización. También permite la construcción jerárquica de las políticas de acceso, por herencia o especialización, teniendo el potencial de reducir la complejidad y el coste de la administración de seguridad en entornos heterogéneos. Dada la alta integración entre los roles y las responsabilidades de los usuarios, se pueden seguir los principios del mínimo privilegio y de la separación de las responsabilidades. Estos principios son vitales para alcanzar el objetivo de integridad, al requerir que a un usuario no se le otorguen mayores privilegios que los necesarios para efectuar su trabajo.

Luego de realizar un estudio de los modelos de control de acceso se decide utilizar el RBAC por tener alta integración entre los roles y las responsabilidades de los usuarios, siguiendo los principios del mínimo privilegio y la separación de responsabilidades.

En la implementación del RBAC se deben tener en cuenta los siguientes principios:

- ✓ Un usuario tiene acceso a las funcionalidades de acuerdo al rol que tenga asignado.
- ✓ Los roles son definidos en base a funciones de trabajo.
- ✓ Los permisos son definidos en función de la autoridad y responsabilidad que se asume en una función de trabajo.

- ✓ La ejecución de las funcionalidades son invocadas de acuerdo a los permisos.

1.2 Herramientas, tecnologías y metodología de desarrollo.

En el presente epígrafe se realiza un análisis de las herramientas, tecnologías y metodologías de desarrollo seleccionadas por el equipo de arquitectura del Sistema Xabal Arkheia, basado en el aporte que brindan en el desarrollo de la solución propuesta.

1.2.1 Lenguajes y tecnologías de desarrollo.

Los lenguajes de programación que se utilizan para la realización de aplicaciones web, se denominan lenguajes de programación web, dividiéndose en lenguajes de programación del lado del cliente y del lado del servidor. Los lenguajes de programación del lado del cliente se ejecutan en el navegador del usuario, tienen entre sus funciones la validación de los datos de entrada y los cambios en la apariencia que se necesiten realizar. Los lenguajes de programación del lado del servidor realizan la conexión entre la base de datos que se va a utilizar y el servidor, encargándose de los aspectos relacionados con la funcionalidad del sistema.

Los lenguajes y framework de desarrollo seleccionados del lado del cliente son:

HTML (Hyper Text Markup Language): es un lenguaje de hipertexto que permite escribir textos de forma estructurada. Compuesto por etiquetas que marcan el inicio y el fin de cada elemento del documento permitiendo describir las páginas web. Los documentos HTML deben tener la extensión html o htm, para que puedan ser visualizados en los navegadores, los que se encargan de interpretar ese código y mostrar a los usuarios las páginas resultantes.

HTML consta de varios componentes entre los que están los elementos y sus atributos, tipos de dato y la declaración de tipo de documento. Describe la estructura y el contenido en forma de texto empleando etiquetas (<Body>…</Body> o <P>…</P>) para ello. Debido a que es un estándar independiente de fabricantes y marcas puede ser interpretado por todos los navegadores, siendo sus aplicaciones muy rápidas, con mucho desarrollo y ocupan poco espacio. (19)

CSS: es un lenguaje de hojas de estilos creados para controlar la presentación de los documentos electrónicos definidos con HTML y XHTML[8], siendo la mejor forma de separar los contenidos y su presentación y muy necesario para crear páginas web complejas. El separar la definición de los contenidos y la definición de su aspecto, mejora la accesibilidad del documento, reduciendo la complejidad de su mantenimiento y garantizando que sus páginas web tengan un estilo coherente en todo el sitio. (20)

JavaScript: es un lenguaje de programación interpretado, orientado a objetos y basado en prototipos. Se utiliza para acceder a los objetos en aplicaciones permitiendo el desarrollo de

[8]Extensible HyperText Markup Language (Lenguaje de marcado de Hipertexto Extensible).

interfaces de usuario mejoradas y páginas web dinámicas. JavaScript se diseñó con una sintaxis similar a C, aunque adopta nombres y convenciones del lenguaje de programación Java, no guarda ninguna relación directa con este y es interpretado por todos los navegadores modernos. Un problema que presenta es que el código es visible y puede ser leído por cualquier usuario. También los script poseen capacidades limitadas por razones de seguridad, es necesario usarlo junto a otros lenguajes más seguros, como Java. (21)

Bootstrap 2.0: es un framework que simplifica el proceso de creación de diseños web combinando CSS y JavaScript. Es desarrollado por Twitter que liberó su versión 2.0. Su mayor ventaja es que se pueden crear interfaces que se adapten a los distintos navegadores. Se integra perfectamente con las principales librerías JavaScript, por ejemplo JQuery. (22)

Los lenguajes y framework de desarrollo seleccionados del lado del servidor son:

Máquina Virtual de Java (JVM[9]) 7: es la encargada de ejecutar las aplicaciones que se escriben en java, el compilador no realiza una compilación a código de máquina, sino, a un código intermedio llamado *bytecode*, el cual es interpretado por la JVM garantizando la correcta ejecución en distintas plataformas. (23)

Groovy 2.0: es un lenguaje de programación orientado a objetos implementado sobre la plataforma Java, es un lenguaje en el que se escriben las aplicaciones en Grails[10], dinámico ágil que incluye sobrecarga de operadores, sintaxis nativa para la manipulación de listas y mapas, soporte nativo para expresiones regulares, iteración polimórfica, expresiones embebidas dentro de strings. Groovy aporta una sintaxis que aumenta enormemente la productividad dentro del equipo de desarrolladores. (24)

Grails 2.2.1: es un framework para el desarrollo de aplicaciones web libres, desarrollado sobre el lenguaje de programación Groovy[11]. El objetivo de Grails es brindar al usuario un entorno de alta productividad, extensible y fácil de utilizar. También ofrece el balance adecuado entre consistencia y funcionalidades potentes. Grails se basa en la arquitectura MVC (Modelo-Vista-Controlador) e integra los frameworks java de código abierto como:

- ✓ Spring: es un framework de código abierto de desarrollo de aplicaciones para la plataforma Java. Cuenta con varios módulos que proveen un amplio rango de servicios entre los cuales se pueden encontrar la inversión de control, acceso a datos, manejo de transacciones, entre otros.

- ✓ Hibernate: es una herramienta de Mapeo objeto-relacional (ORM) para la plataforma Java. Es el framework de persistencia más adoptado en aplicaciones web. Facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de

[9] Java Virtual Machine.
[10] Es un framework para aplicaciones web libre desarrollado sobre el lenguaje de programación Groovy.
[11] Lenguaje de programación orientado a objetos implementado sobre la plataforma Java.

objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los beans de las entidades que permiten establecer estas relaciones. (25)

- ✓ SiteMesh: es un framework de decoración y layout para páginas web. Apunta a la creación de grandes sitios que contienen una gran cantidad de páginas. Se trata de un framework para manejar la disposición y contenido de páginas web basándose en la extracción e integración de la información.

Para la realización del sistema se utilizará como lenguaje de programación del lado del cliente HTML por ser ampliamente utilizado, fácil de aprender y usar, y compatible en cada navegador; CSS porque separa el contenido de su presentación, accesibilidad y estructuración, y optimización de los tiempos de carga y de tráfico en el servidor; JavaScript por ser utilizado en la mayoría de los navegadores web modernos (ejemplo: Internet Explorer, Mozilla Firefox, Opera, Chrome); Bootstrap por su simplificación en el proceso de creación de diseños web, los componentes que trae predefinidos y permitir crear interfaces que se adapten a distintos navegadores y dispositivos. Como lenguaje de programación del lado del servidor Groovy por ser un lenguaje dinámico sobre la plataforma Java; y Grails por tener alta productividad al basar su configuración en la convención de nombres, eliminando así extensos ficheros XML e integrando herramientas probadas de código abierto basadas en Java como Spring, SiteMesh e Hibernate.

1.2.2 Entorno de Desarrollo Integrado.

Es una herramienta para desarrollar aplicaciones, ayudando al programador a cometer menor cantidad de errores al escribir el código de la aplicación.

IntelliJ IDEA 12.0: es un entorno integrado de desarrollo inteligente enfocado en la productividad de los desarrolladores. Ofrece herramientas para el desarrollo de aplicaciones web basada en el marco de trabajo Grails, dando soporte al lenguaje Groovy, Java, HTML, JavaScript además de integrar servidores de versiones svn[12]. (26)

Para la realización del sistema se utilizará como entorno de desarrollo integrado IntelliJ IDEA debido a que es una herramienta que permite desarrollar las aplicaciones basadas en Grails, encontrándose entre los principales entornos de desarrollo que brindan soporte a este marco de trabajo. Integra el servidor svn que se utiliza para desarrollar la aplicación y se adapta a las condiciones reales que existen para desarrollar el sistema como son: estaciones de trabajo con 1 GB de memoria RAM.

1.2.3 Herramientas CASE.

Las herramientas CASE son aplicaciones informáticas que permiten aumentar la productividad en el desarrollo de software, minimizando el costo en tiempo y dinero. También ayudan en

[12]Herramienta de control de versiones open source basada en un repositorio.

aspectos de todo el ciclo de vida del software, en tareas como: realizar el diseño del proyecto, cálculo de costes, compilación automática, documentación o detección de errores, entre otras.

Visual Paradigm. 8.0: es una herramienta CASE[13] que propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. (7)

Posee características gráficas muy cómodas que facilitan la realización de los diagramas de modelado que sigue el estándar de UML que son:

- ✓ Diagramas de clase.
- ✓ Casos de Uso.
- ✓ Comunicación.
- ✓ Secuencia.
- ✓ Estado.
- ✓ Actividad.
- ✓ Componentes.

Se decide trabajar con Visual Paradigm para la realización del sistema por su usabilidad, portabilidad y robustez. También por utilizar UML como lenguaje de modelado y permitir la integración con herramientas de desarrollo.

1.2.4 Metodología de Desarrollo.

Una metodología de desarrollo de software es un conjunto de pasos y procedimientos que deben seguirse para el desarrollo de un proyecto.

Proceso Unificado de Desarrollo de Software (RUP): es una metodología de desarrollo de software que está basado en componentes e interfaces bien definidas. Junto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

El Proceso Unificado de Desarrollo de Software es un marco de desarrollo de software que se caracteriza por estar:

***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.

[13] Ingeniería de Software Asistida por Computación.

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, describiendo los elementos del modelo que son más importantes para su construcción. RUP se desarrolla mediante iteraciones, comenzando por los casos de uso relevantes desde el punto de vista de la arquitectura. El modelo de arquitectura se representa a través de vistas (4+1)[14], o sea, perspectivas del sistema que logran una abstracción particular en cada uno de los casos.

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. Las iteraciones hacen referencia a pasos en los flujos de trabajo, los incrementos y al crecimiento del producto. (27)

Las fases del ciclo de vida del proceso de desarrollo unificado son las siguientes:

1. Inicio: en esta fase se realiza un plan de fases, se identifican los principales casos de uso, los principales riesgos del sistema y se propone una visión general de la arquitectura del software.
2. Elaboración: en esta fase se hace un plan de proyecto, se seleccionan los casos de uso que van a definir la arquitectura del software, se realiza la especificación de los casos de uso que fueron seleccionados y se eliminan los riesgos encontrados en el sistema.
3. Construcción: esta fase se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario.
4. Transición: en esta fase se asegura que el software se encuentre disponible para los usuarios finales. Se verifica que el software cumpla con las especificaciones de los usuarios. Como consecuencia de esto suelen surgir nuevos requisitos.

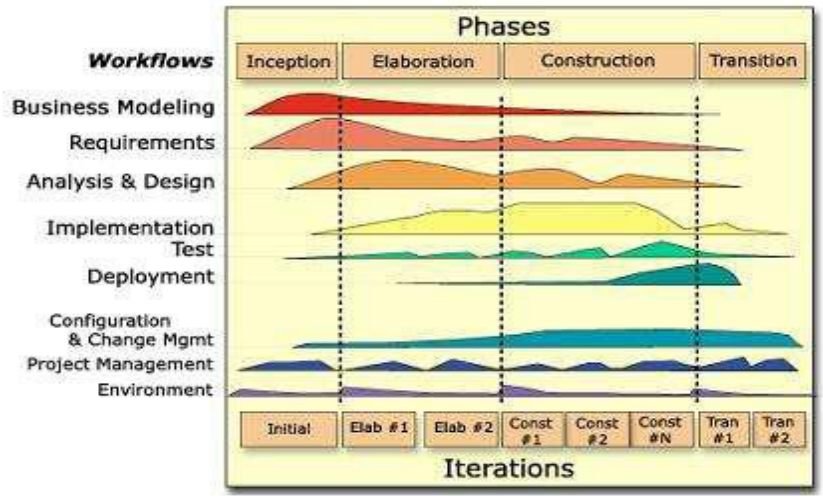

Imagen 1 Fases e iteraciones de la metodología RUP.

[14] 4+1 vistas de la arquitectura: Vista de casos de uso, vista lógica, vista de procesos, vista de componentes y vista de despliegue

Para el desarrollo del sistema se seleccionó la metodología de desarrollo RUP porque brinda la posibilidad de realizar una extensa documentación, permitiendo que en un futuro, otro equipo de desarrollo pueda brindar soporte a la aplicación o realizar nuevas versiones.

1.2.5 Sistema Gestor de Base de Datos.

Un SGBD es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y la aplicación. Está compuesto por un lenguaje de definición de datos y manipulación de datos y consulta, garantizando su seguridad e integridad.

PostgreSQL 9.0: es un SGBD[15] relacional orientado a objetos. Es publicado bajo la licencia BSD[16], que incluye características de la orientación a objetos: la herencia, tipos de datos, funciones, restricciones, disparadores, entre otros. Utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. (28)

Entre las características principales de PostgreSQL se encuentran:

- ✓ Escalabilidad.
- ✓ Implementa el uso de rollback's, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz.
- ✓ Capacidad de comprobar la integridad referencial.
- ✓ Almacena procedimientos en la base de datos.

Se seleccionó trabajar con el SGBD PostgreSQL porque es posible ejecutarlo en los sistemas operativos: Linux, Unix, Windows, entre otros. Posee una documentación organizada, pública y libre, con comentarios de los mismos usuarios y altamente adaptable a las necesidades del cliente. Además, presenta buen rendimiento a la hora de procesar gran cantidad de datos.

1.3 Conclusiones.

Con el análisis de los sistemas seleccionados, se puede observar que la gestión de documentos es un tema que ha venido desarrollándose alrededor del mundo por la necesidad de las instituciones de organizar y preservar sus archivos. La forma de cómo se hace, en cada lugar varía según sus necesidades. Para lograr que un sistema se adapte a los requerimientos de cada entidad, es necesario que su mecanismo de administración brinde un instrumento de apoyo a las frecuentes transformaciones que puedan ocurrir. Basados en estos sistemas, se definirán los requerimientos funcionales del presente trabajo.

La integridad y el acceso a los datos tienen importancia vital, por ser los documentos la vida de las entidades donde se implementará la solución para la gestión de archivos. Mediante la caracterización de los modelos de control de acceso se realizó la selección del modelo Control

[15] Sistema de Gestión de Base de Datos.
[16] Berkeley Software Distribution.

de Acceso Basado en Roles (RBAC) por ser un método flexible en la asignación de los permisos a los usuarios del sistema, aportando así un elemento más en el alcance del objetivo que se persigue con el desarrollo del Sistema Xabal Arkheia: ser adaptable a nuevas características de los clientes.

Además en este capítulo se abordaron los principales conceptos de la administración informática y la gestión de archivos, para una mejor comprensión de la presente investigación. Con el análisis de los sistemas de gestión de archivos seleccionados, se realizó un desglose de las principales características en el área de la administración informática, aportando una visión inicial de los requerimientos del subsistema.

Teniendo en cuenta el análisis realizado previamente, se desarrollará el sistema con el lenguaje de programación del lado del servidor Groovy 2.0 y framework de desarrollo Grails 2.2.1, del lado del cliente HTML, CSS, JavaScript y como framework de presentación Bootstrap 2.0. Se propone además, para garantizar la realización de la aplicación de forma adecuada, el uso de la herramienta CASE Visual Paradigm 8.0.

Capítulo 2

2 Análisis y Diseño.

En este capítulo se expondrán las diferentes características del sistema propuesto con el objetivo de cumplir con las expectativas del cliente y usuarios finales. Se explica el por qué se realizó un modelo de dominio, exponiéndose los requisitos funcionales, no funcionales, los casos de uso del sistema, y realizándose el análisis y diseño de lo que constituirá el subsistema.

2.1 Modelo de dominio.

El modelo de dominio permite la representación de un conjunto de conceptos, y las relaciones que se establecen entre ellos ayudarán tanto a usuarios como a desarrolladores, a usar un vocabulario común, que les permita entender el contexto en que se desarrolla el sistema.

A continuación se detallan los conceptos principales del modelo de dominio:

- ✓ **Usuario:** persona que tiene acceso para trabajar con el sistema.
- ✓ **Administrador:** usuario con permisos de administración encargado de realizar la gestión de usuarios y roles, la gestión de nomencladores y la configuración del sistema.
- ✓ **Rol:** papel que desempeña un usuario en el sistema.
- ✓ **Permisos:** autorizaciones que les serán concebidas a los diferentes roles, a través del cual se permita realizar solo las funcionalidades necesarias por los roles.
- ✓ **Reporte:** documento en formato pdf con información estadística y tabular de los datos almacenados en la base de datos.
- ✓ **Auditoría:** almacena los valores de las acciones realizadas por los usuarios desde su puesto de trabajo.
- ✓ **Nomenclador:** entidad que almacena los valores de los nomencladores del sistema.
- ✓ **Configuración:** entidad que almacena los parámetros del sistema.
- ✓ **Cuadro de Clasificación:** define la ubicación lógica de los documentos.
- ✓ **Estructura de Ubicación Física:** define la ubicación física de los documentos.
- ✓ **Nuxeo:** define los parámetros de acceso al servidor de gestión documental Nuxeo.
- ✓ **Cuenta de Administración:** cuenta creada con permisos y privilegios al módulo de administración informática.

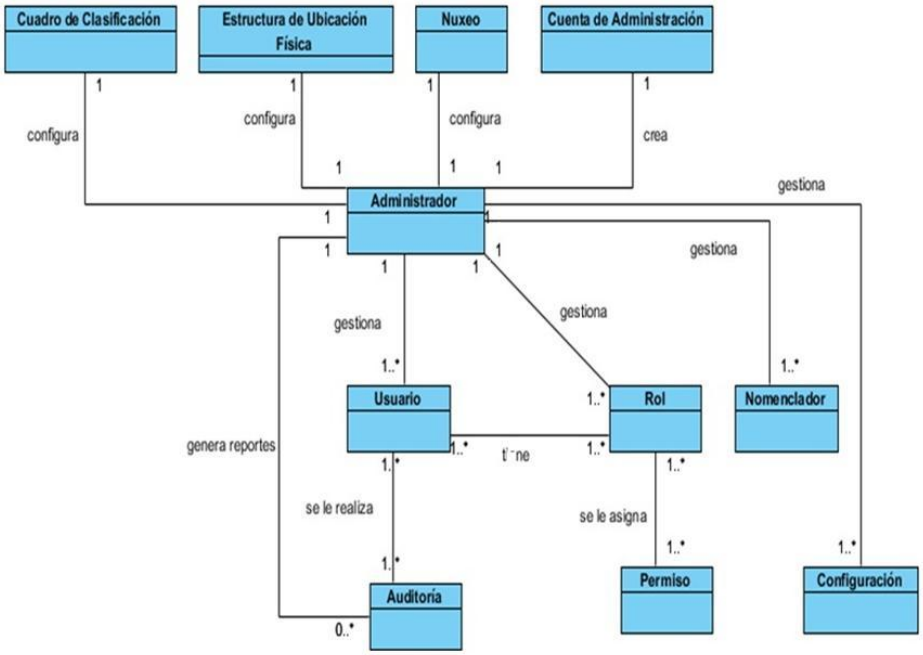

Imagen 2 Diagrama de clases del dominio.

2.2 Reglas del negocio.

Las reglas del negocio son aquellas condiciones, validaciones y normas que se deben cumplir y controlar dentro de la aplicación y que son definidas de acuerdo con el comportamiento esperado.

En la realización del subsistema se tendrán en cuenta algunas reglas que deben cumplirse para su buen funcionamiento. Las reglas son las siguientes:

- ✓ Solo tiene acceso a la información el personal autorizado.
- ✓ La información debe ser precisa y persistente.
- ✓ Los nombres de usuarios deben ser únicos.
- ✓ Los usuarios deben tener carnet de identidad único.
- ✓ No puede existir dos roles con el mismo nombre ni con los mismos permisos.
- ✓ Los usuarios con auditorías asociadas no pueden ser eliminados.
- ✓ El administrador creado en el módulo de configuración inicial debe tener acceso a todas las funcionalidades de administración.

2.3 Especificación de los requisitos de software.

El propósito fundamental del flujo de trabajo de los requisitos es guiar el desarrollo hacia el sistema correcto. Esto se consigue mediante una descripción de los requisitos del sistema (las condiciones o capacidades que el sistema debe cumplir) suficientemente buena como para que pueda llegarse a un acuerdo entre el cliente (incluyendo a los usuarios) y los desarrolladores sobre que debe y que no debe hacer el sistema. (29)

Los requisitos de software se clasifican en 2 categorías: los requisitos funcionales y los requisitos no funcionales.

2.3.1 Requisitos Funcionales

Un requisito funcional es una característica requerida del sistema que expresa una capacidad de acción del mismo (una funcionalidad); generalmente expresada en una declaración en forma verbal. Se traducen directamente en casos de uso.

RF1 Autenticar Usuario.

El sistema debe permitir que los usuarios se autentiquen introduciendo sus credenciales (usuario y contraseña).

RF2 Ver Perfil de Usuario.

El sistema debe permitir al usuario ver su perfil.

RF3 Cerrar Sesión.

El sistema debe permitir cerrar sesión del usuario autenticado.

RF4 Registrar Usuario.

El sistema debe permitir registrar un usuario especificando la información necesaria para su registro. Los campos a introducir son:

- ✓ Nombre(s).
- ✓ Primer apellido.
- ✓ Segundo apellido.
- ✓ Carnet de identidad.
- ✓ Sexo.
- ✓ Dirección.
- ✓ Cargo.
- ✓ Correo Electrónico.
- ✓ Teléfono.
- ✓ Nombre de Usuario.
- ✓ Contraseña.
- ✓ Confirmación de Contraseña.
- ✓ Roles.

RF5 Modificar Usuario.

El sistema debe permitir modificar los datos del usuario.

RF6 Mostrar Usuario.

El sistema debe mostrar los usuarios obtenidos, solicitando los siguientes criterios de búsqueda:

- ✓ Carnet de identidad.
- ✓ Nombre(s).
- ✓ Primer apellido.
- ✓ Segundo apellido.
- ✓ Nombre de usuario.
- ✓ Rol.

RF7 Establecer Contraseña.

El sistema debe permitir que el administrador pueda establecer la contraseña del usuario seleccionado, al introducir los campos:

- ✓ Nueva Contraseña.
- ✓ Confirmar Contraseña.

RF8 Cambiar Contraseña.

El sistema debe permitir que el usuario tenga la posibilidad de cambiar la contraseña, al introducir los campos:

- ✓ Contraseña Anterior.
- ✓ Nueva Contraseña.
- ✓ Confirmar Contraseña.

RF9 Activar/Desactivar Usuario.

El sistema debe permitir modificar el estado de activación del usuario y lo almacena en la Base de Datos como activado/desactivado.

RF10 Registrar Rol.

El sistema debe permitir registrar un Rol al introducir los siguientes datos:

- ✓ Rol.
- ✓ Listado de módulos.
- ✓ Listado de permisos.

RF11 Modificar Rol.

El sistema debe permitir modificar los datos del rol seleccionado, usando el id del mismo. Los datos a introducir son:

- ✓ Rol.
- ✓ Listado de permisos.

RF12 Eliminar Rol.

El sistema debe permitir eliminar completamente el rol del sistema.

RF13 Mostrar Rol.

El sistema debe mostrar una vista con los datos del rol seleccionado.

RF14 Registrar Nomenclador.

El sistema debe permitir registrar un Nomenclador al introducir el campo:

- ✓ Valor.

RF15 Modificar Nomenclador.

El sistema debe permitir modificar un Nomenclador al introducir los campos:

- ✓ Valor.

RF16 Mostrar Nomenclador.

El sistema debe mostrar los nomencladores.

RF17 Crear Cuenta de Administración.

El sistema debe permitir crear un administrador inicial.

RF18 Mostrar Auditoría.

El sistema debe mostrar la auditoría del usuario al introducir los siguientes datos:

- ✓ Fecha Inicial.
- ✓ Fecha Final.
- ✓ Usuario.

RF19 Registrar Auditoría.

El sistema debe registrar las acciones realizadas por los usuarios.

RF20 Generar Reporte de Auditoría.

El sistema debe permitir generar un documento en formato pdf con los datos de las acciones realizadas por los usuarios.

RF21 Generar Reporte de Usuarios.

El sistema debe permitir generar un documento en formato pdf con los datos de los usuarios registrados en el sistema.

RF22 Mostrar Usuarios Conectados.

El sistema debe mostrar los usuarios que se encuentran conectados. Los datos a mostrar por cada usuario conectado son:

- ✓ Nombre(s) y Apellidos.
- ✓ Usuario.
- ✓ Dirección IP.

RF23 Registrar Parámetros de Configuración.

El sistema debe permitir registrar los parámetros de configuración.

RF24 Modificar Parámetros de Configuración.

El sistema debe permitir modificar los parámetros de configuración.

2.3.2 Requisitos No Funcionales.

Un requisito no funcional es una propiedad o cualidad que el producto debe tener. Es importante para que clientes y usuarios puedan valorar las características no funcionales del producto, ya que si se conoce que el sistema cumple con la toda la funcionalidad requerida, las propiedades no funcionales como cuán usable, seguro, conveniente y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

Requerimientos de Software.

1. Del lado del Servidor: sistema operativo Windows o Linux, Máquina Virtual de Java 7 y Tomcat 7.23.
2. Del lado del cliente: navegador web Mozilla Firefox 17 o superior, Google Chrome 23 o superior y Lector de pdf Adobe Reader 10 o Superior.

Requerimientos de Hardware.

3. Servidor: requerimientos mínimos: 512 Mb RAM.
4. Cliente: requerimientos mínimos: 128 Mb RAM.

Restricciones en el diseño y la implementación.

En este requerimiento se especifica la construcción del sistema, siendo las restricciones ordenadas y deben ser cumplidas estrictamente. Ejemplo:

5. Framework de desarrollo Grails 2.2.1.
6. Framework de Presentación Bootstrap 2.0.
7. Arquitectura Modelo Vista Controlador.

Requerimientos de apariencia o interfaz externa.

Este requerimiento estará regido por el estándar de diseño Xabal definido por el departamento de diseño de la Universidad de las Ciencias Informáticas, que define:

8. El color predominante será el rojo.
9. Las acciones de los usuarios estarán representadas en la parte superior derecha.
10. Menú en dos niveles: el primer nivel representa el nombre del módulo y el segundo nivel las funcionalidades específicas de cada módulo.
11. Mostrar un encabezado de la página con el nombre del módulo y el título de la funcionalidad activa.
12. Los formularios no pueden exceder de 4 componentes de forma horizontal.

Requerimientos de Seguridad.

Este requerimiento provoca grandes riesgos si no se maneja correctamente. La seguridad puede ser tratada en tres aspectos diferentes:

13. Confidencialidad: el acceso al sistema está restringido al personal autorizado, dándole un usuario, contraseña y un rol para el acceso a solo las funcionalidades autorizadas.

14. Integridad: el sistema audita las acciones de los usuarios en todo momento.

15. Disponibilidad: significa que los usuarios autorizados se les garantizará el acceso a la información y que los dispositivos o mecanismos utilizados para lograr la seguridad no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado.

Requerimientos de Soporte.

En este requerimiento se llevan a cabo las acciones a tomar una vez que se ha terminado el desarrollo del sistema con motivo de lograr su mejoramiento progresivo y evolución en el tiempo para que el cliente se sienta satisfecho. El soporte del sistema incluye:

16. Pruebas Pilotos: se realizan para probar el software en el ambiente real en un determinado tiempo.

17. Peticiones de cambio: cuando se realizan las pruebas pilotos pueden existir las peticiones de cambio, lo cual es una solicitud del cliente para la modificación de alguna funcionalidad del sistema.

18. Mantenimiento: es la solución de problemas que se presenten en el software por errores de la implementación del código.

19. Instalación y Configuración: el equipo de desarrollo se encarga de la instalación y configuración del servidor web y servidor de base de datos.

20. Internacionalización: el sistema está preparado para configurarse en varios idiomas.

Requerimientos de rendimiento.

El sistema debe ser eficiente y con una velocidad de procesamiento, tiempo de respuesta y recuperación óptimo, así como un buen aprovechamiento de los recursos del servidor.

21. La respuesta del sistema es en no más de 1 minuto.

Requerimientos de portabilidad.

22. El sistema podrá ser usado bajo cualquier sistema operativo, ejemplo: Linux o Windows.

Requerimiento de usabilidad.

En este requerimiento se describe los niveles apropiados de usabilidad teniendo en cuenta los perfiles de usuario y sus niveles de experiencia.

Los requerimientos están especificados en el documento “LCH_Usabilidad_Arkheia”. (**Ver Anexo VI**)

2.4 Patrones de caso de uso a utilizar.

Utilizando los patrones de casos de uso, arquitectos, analistas e ingenieros, pueden lograr mejores resultados de forma eficiente, en la realización de un sistema. Los patrones de CU utilizados son los siguientes:

CRUD Completo: consiste en un caso de uso para gestionar la información, permitiendo modelar las diferentes operaciones para administrar una entidad de información, tales como insertar, obtener, modificar y eliminar. Un ejemplo de su uso se evidencia cuando el actor administrador inicializa el caso de uso Gestionar Usuario:

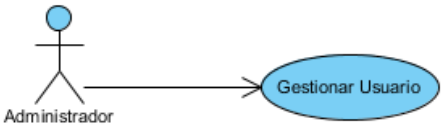

Imagen 3 Patrón de Caso de Uso: CRUD Completo.

Extensión: consiste en dos casos de usos y una relación extendida entre ellos. Hay situaciones en que el caso de uso de extensión no es indispensable que ocurra, y cuando lo hace ofrece un valor extra (extiende) al objetivo original del caso de uso base. Un ejemplo de su uso se evidencia en la siguiente figura:

Imagen 4 Patrón de Caso de Uso: Extensión.

Inclusión: se incluye una relación del caso de uso base al caso de uso de inclusión, es decir, el segundo es parte esencial del primero. Sin el segundo, el primero no podría funcionar correctamente; pues no podría cumplir su objetivo. Como ejemplo de su aplicación se puede ver la figura que a continuación se muestra:

Imagen 5 Patrón de Caso de Uso: Inclusión.

2.5 Modelo del sistema. Definición de actores y casos de uso del sistema.

En este epígrafe se describen los actores y los casos de uso que rigen el desarrollo del Subsistema de Administración Informática y Configuración Inicial del Sistema Xabal Arkheia.

2.5.1 Definición de los actores del sistema.

Un actor es una entidad externa que va a interactuar con el sistema, ya sea un ser humano o un sistema de software. Los actores sirven para capturar a los usuarios del sistema. Es decir, que los actores toman el lugar de cualquier entidad de interés con la que el sistema interactúa. (29)

Imagen 6 Actor del sistema.

Actor	Justificación
Usuario	Persona que tiene acceso para trabajar con el sistema.
Administrador	Usuario con permisos de administración encargado de realizar la gestión de roles, usuarios, nomencladores y la configuración del sistema.

Tabla 2 Descripción de los actores del sistema.

2.5.2 Definición de los casos de uso del sistema.

Un caso de uso especifica una secuencia de acciones, incluyendo variantes, que el sistema puede llevar a cabo, y que producen un resultado observable de valor para un actor concreto. (29)

Imagen 7 Caso de uso del sistema.

A continuación se muestran los diagramas de casos de uso del Subsistema de Administración Informática y Configuración Inicial. Para un mejor entendimiento de estos diagramas, se definen las siguientes normas:

- ✓ Color amarillo: cuando los casos de uso pertenecen a otros módulos del Sistema Xabal Arkheia.
- ✓ Color azul: cuando los casos de uso pertenecen al Subsistema de Administración Informática y Configuración Inicial.

Imagen 8 Diagrama de CU del módulo Administración Informática.

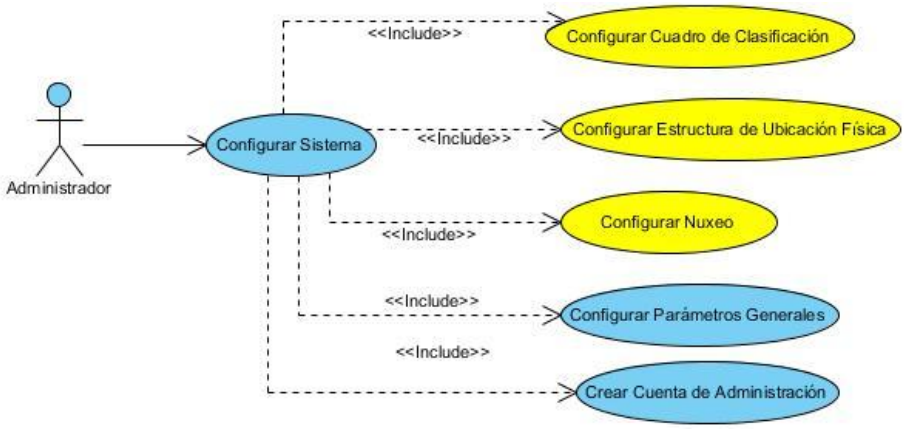

Imagen 9 Diagrama de CU del módulo Configuración Inicial.

La descripción de los CU de los módulos Administración Informática y Configuración Inicial se muestran a continuación:

Objetivo	Registrar, modificar, eliminar, activar, desactivar y ver detalles de un usuario en el sistema.
Actores	Administrador: (Inicia) Registra, modifica, activa, desactiva, ve los detalles y elimina un usuario en el sistema. Usuario: Ve los detalles del usuario autenticado en el sistema.
Resumen	El caso de uso inicia cuando el Actor decide realizar las acciones permitidas sobre un usuario. Las acciones son: registrar, modificar, activar, desactivar y ver detalles de usuario. El sistema permite realizar la acción solicitada por el Actor y termina el caso de uso.
Complejidad	Media.
Prioridad	Alta.
Precondiciones	• Debe estar mostrado el listado de usuarios. • Debe existir un usuario seleccionado.
Postcondiciones	• Datos del usuario registrados en la BD. • Datos del usuario actualizados en la BD. • Datos del usuario eliminados de la BD.

Tabla 3 Definición del CU Gestionar Usuario del módulo Administración Informática.

Objetivo	Mostrar auditorías en el sistema.
Actores	Administrador: (Inicia) Busca auditorías en el sistema.
Resumen	El Administrador selecciona la opción "Mostrar Auditoría". Se muestra un formulario con los criterios de búsqueda. El administrador introduce los criterios de búsqueda. Se obtiene de la BD los datos que cumplen con los criterios de búsqueda especificados.
Complejidad	Baja.
Prioridad	Media.
Precondiciones	No Aplica.
Postcondiciones	No Aplica.

Tabla 4 Definición del CU Mostrar Auditoría del módulo Administración Informática.

Objetivo	Registrar, modificar y eliminar los roles de los usuarios.
Actores	Administrador: (Inicia) Registra, modifica y elimina los roles de los usuarios en el sistema.
Resumen	El Administrador solicita realizar una acción sobre un rol, el sistema muestra el formulario que permite realizar la acción solicitada. Las acciones son: registrar, modificar y eliminar roles de un usuario. El Administrador introduce los datos especificados. Se almacenan los datos en la BD.
Complejidad	Media.
Prioridad	Alta.
Precondiciones	• Debe estar mostrado el listado de roles. • Debe existir un rol seleccionado.
Postcondiciones	• Datos del rol registrados en la BD. • Datos del rol actualizados en la BD. • Datos del rol eliminados de la BD.

Tabla 5 Definición del CU Gestionar Rol del módulo Administración Informática.

Objetivo	Adicionar, modificar y eliminar datos de nomencladores.
Actores	Administrador: (Inicia) Adiciona, modifica y elimina datos de nomencladores en el sistema.
Resumen	El Administrador solicita realizar una acción sobre un nomenclador, el sistema muestra el formulario que permite realizar la acción solicitada. Las acciones son: adicionar, modificar y eliminar datos de un nomenclador. El Administrador selecciona el nomenclador e introduce los datos especificados. Se almacenan los datos en la BD.
Complejidad	Media.
Prioridad	Alta.
Precondiciones	• Deben existir elementos del nomenclador registrados.
Postcondiciones	• Listado de elementos del nomenclador mostrado. • Datos del nomenclador registrados en la BD. • Datos del nomenclador actualizados en la BD. • Datos del nomenclador eliminados en la BD.

Tabla 6 Definición del CU Gestionar Nomencladores del módulo Administración Informática.

Objetivo	Generar reportes en el sistema.
Actores	Administrador.
Resumen	Obtiene de la BD y muestra en un documento PDF los datos que coincidan con los criterios de búsqueda especificados.
Complejidad	Baja.
Prioridad	Media.
Precondiciones	No Aplica.
Postcondiciones	No Aplica.

Tabla 7 Definición del CU Generar Reporte del módulo Administración Informática.

Objetivo	Cargar la configuración del usuario en el sistema.
Actores	Usuario.
Resumen	Obtiene de la BD las opciones a las que tiene acceso el usuario autenticado.
Complejidad	Media.
Prioridad	Media.
Precondiciones	No Aplica.
Postcondiciones	• Cargada la configuración del usuario autenticado en el sistema.

Tabla 8 Definición del CU Cargar Configuración del módulo Administración Informática.

Objetivo	Actualizar la configuración de los parámetros generales del sistema.
Actores	Administrador: (Inicia) Actualiza la configuración de los parámetros generales del sistema.
Resumen	El caso de uso inicia cuando el Administrador decide modificar la configuración de los parámetros generales en el sistema. El sistema permite actualizar los valores modificados por el Administrador y termina el caso de uso.
Complejidad	Baja.
Prioridad	Media.
Precondiciones	• Debe existir parámetros de configuración en la BD.
Postcondiciones	• Valores actualizados en la BD.

Tabla 9 Definición del CU Actualizar Configuración del módulo Administración Informática.

Objetivo	Establecer y cambiar la contraseña de un usuario.
Actores	Administrador: (Inicia) Establece contraseñas en el sistema. Usuario: Cambia la contraseña en el sistema.
Resumen	El actor solicita realizar una acción sobre la contraseña de un usuario seleccionado, el sistema muestra el formulario que permite realizar la acción solicitada. Las acciones son: establecer y cambiar la contraseña de un usuario. El actor introduce los datos especificados. Se almacenan los datos en la BD.
Complejidad	Media.
Prioridad	Alta.
Precondiciones	• Debe estar mostrado el listado de los usuarios. • Debe existir un usuario seleccionado. • El usuario debe estar activado.
Postcondiciones	• Se actualiza el estado de la contraseña a "*Cambiada por el Administrador*". • Se actualiza el estado de la contraseña a "*Cambiada por el Usuario*". • Contraseña actualizada.

Tabla 10 Definición del CU Gestionar Contraseña del módulo Administración Informática.

Objetivo	Cerrar la sesión del usuario.
Actores	Usuario: (Inicia) Cierra la sesión.
Resumen	El usuario selecciona en las opciones de usuario la opción "Cerrar Sesión" y el sistema cierra la sesión del usuario autenticado.
Complejidad	Media.
Prioridad	Media.
Precondiciones	No Aplica.
Postcondiciones	• Sesión cerrada.

Tabla 11 Definición del CU Cerrar Sesión del módulo Administración Informática.

Objetivo	Ver el reporte de usuarios en el sistema.
Actores	Administrador: (Inicia) Busca usuarios en el sistema y ve el reporte de los mismos.
Resumen	El Administrador selecciona la opción "Reporte de Usuarios". Se obtiene de la BD los datos de todos los usuarios registrados en el sistema. Se muestra un documento PDF con los datos obtenidos.
Complejidad	Baja.
Prioridad	Media.
Precondiciones	No Aplica.
Postcondiciones	No Aplica.

Tabla 12 Definición del CU Ver Reporte de Usuarios del módulo Administración Informática.

Objetivo	Autenticar usuario en el sistema.
Actores	Usuario: (Inicia) Se autentica en el sistema.
Resumen	El usuario introduce los datos para autenticarse, se verifica que los datos introducidos son correctos y que el usuario cuenta con los permisos necesarios para autenticarse en el sistema. En caso contrario no se permite la autenticación del usuario.
Complejidad	Media.
Prioridad	Alta.
Precondiciones	No Aplica.
Postcondiciones	• Usuario autenticado en el sistema.

Tabla 13 Definición del CU Autenticar Usuario del módulo Administración Informática.

Objetivo	Mostrar los usuarios que estén conectados.
Actores	Administrador: (Inicia) Busca los usuarios que estén conectados en el sistema.
Resumen	El Administrador selecciona la opción "Usuarios Conectados". Se obtiene de la BD el listado de usuarios conectados en el sistema. Se muestra el listado de usuarios conectados obtenido.
Complejidad	Baja.
Prioridad	Media.
Precondiciones	No Aplica.
Postcondiciones	• Listado de usuarios conectados.

Tabla 14 Definición del CU Mostrar Usuarios Conectados del módulo Administración Informática.

Objetivo	Mostrar usuarios en el sistema.
Actores	Administrador: (Inicia) Busca usuarios en el sistema.
Resumen	El Administrador selecciona la opción "Mostrar Usuario". Se muestra un formulario con los criterios de búsqueda. El administrador introduce los criterios de búsqueda. Se obtiene de la BD y se muestra el listado de usuarios que cumplan con los criterios de búsqueda especificados.
Complejidad	Baja.
Prioridad	Media.
Precondiciones	No Aplica.
Postcondiciones	No Aplica.

Tabla 15 Definición del CU Mostrar Usuarios del módulo Administración Informática.

Objetivo¤	Configurar·los·parámetros·generales·del·sistema.·¤
Actores¤	Administrador:·(Inicia)·Configura·los·parámetros·generales·del·sistema.¤
Resumen¤	El· caso· de· uso· inicia· cuando· el· Administrador· decide· configurar· los· parámetros· generales· en· el· sistema.· El· sistema· almacena· la· información· introducida·por·el·Administrador·y·termina·el·caso·de·uso.¤
Complejidad¤	Baja.¤
Prioridad¤	Media.¤
Precondiciones¤	No·Aplica.¤
Postcondiciones¤	•→ Parámetros·generales·configurados.¤

Tabla 16 Definición del CU Configurar Parámetros Generales del módulo Configuración Inicial.

Objetivo	Crear la cuenta de administración en el sistema.
Actores	Administrador: (Inicia) Crea la cuenta de administración en el sistema.
Resumen	El caso de uso inicia cuando el Administrador decide crear la cuenta de administración del sistema. El sistema permite guardar la información introducida por el Administrador y termina el caso de uso.
Complejidad	Baja.
Prioridad	Media.
Precondiciones	No Aplica.
Postcondiciones	• Cuenta de Administración creada.

Tabla 17 Definición del CU Crear Cuenta de Administración del módulo Configuración Inicial.

Objetivo	Instalar y configurar el sistema.
Actores	Administrador: (Inicia) Instala y configura el sistema.
Resumen	El caso de uso inicia cuando el Administrador decide instalar y configurar inicialmente el sistema. El sistema permite realizar la acción solicitada por el Administrador y termina el caso de uso.
Complejidad	Alta.
Prioridad	Media.
Precondiciones	No Aplica.
Postcondiciones	• Sistema instalado. • Sistema configurado.

Tabla 18 Definición del CU Instalar y Configurar del módulo Configuración Inicial.

Para ver la descripción completa de los CU **(VerAnexo II)**.

2.6 Modelo de análisis.

El modelo de análisis es un modelo conceptual, ya que en él se identifican una serie de clases y relaciones que conectadas entre sí, permiten una mejor comprensión de los requisitos de la aplicación que se está modelando.

2.6.1 Diagramas de clases del análisis.

Las clases del análisis se centran en los requisitos funcionales y son evidentes en el dominio del problema porque representan conceptos y relaciones del dominio. Tienen atributos y entre ellas se establecen relaciones de asociación, agregación / composición, generalización / especialización y tipos asociativos. RUP propone clasificar las clases en: clase de interfaz, clase controladora y clase entidad. A continuación se muestra el diagrama de clases del análisis del CU Gestionar Usuario:

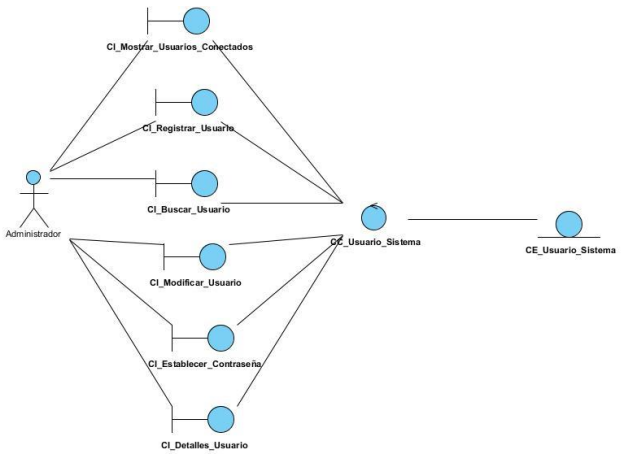

Imagen 10 Diagrama de clases del análisis CU Gestionar Usuario.

Para ver los restantes diagramas de clases del análisis, remitirse al **(Anexo III)**.

2.6.2 Diagramas de Secuencia.

Otro de los diagramas del flujo de trabajo Análisis y Diseño, es el diagrama de secuencia que muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso. Contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario y mensajes intercambiados entre los objetos. A continuación se muestra el diagrama de secuencia del CU Modificar Usuario:

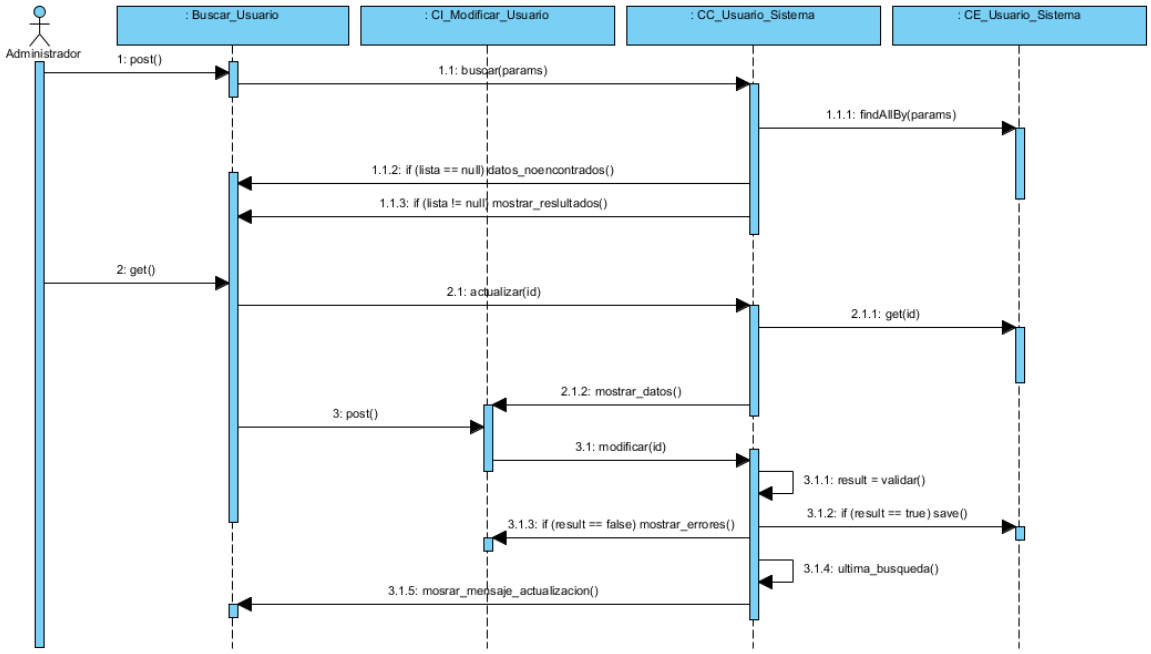

Imagen 11 Diagrama de Secuencia del CU Modificar Usuario.

Para ver los restantes diagramas de secuencia, remitirse al **(Anexo III)**.

2.7 Modelo de diseño.

El Diseño es el centro de atención al final de la fase de elaboración y el comienzo de las iteraciones de construcción. Esto contribuye a una arquitectura estable y sólida, creando un plano del modelo de implementación.

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en los requisitos funcionales como en los no funcionales. Las abstracciones del modelo de diseño tienen una correspondencia directa con los elementos físicos del ambiente de implementación. (29)

2.7.1 Diagrama de clase del diseño.

El diagrama de clases se modeló en paquetes para dividir el modelo en partes manejables mediante la agrupación de las diferentes clases. Teniéndose un paquete general, el cual en su interior tiene dos paquetes que son los dos módulos por los que está compuesto el subsistema.

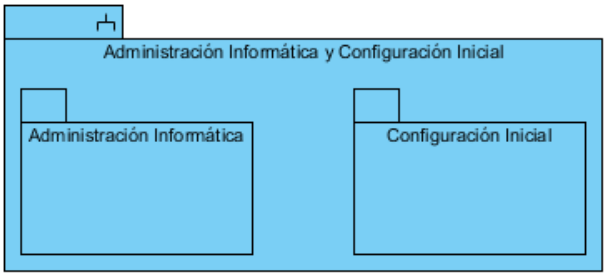

Imagen 12 Diagrama de clases agrupado por paquetes.

2.7.2 Diagrama de clase del diseño web.

El diagrama de clase del diseño web es un artefacto que representa un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos, aplicándole los estereotipos web a las clases, formularios, vistas y páginas servidoras.

Para un mejor entendimiento del origen de las clases representadas en los diagramas de clases del diseño y en el diagrama de clases persistentes, se definen las siguientes normas:

- ✓ Color amarillo: cuando las clases pertenecen a otros módulos o a la arquitectura del Sistema Xabal Arkheia.
- ✓ Color anaranjado: cuando las clases pertenecen al framework de desarrollo o al lenguaje de programación Groovy.
- ✓ Color azul: cuando las clases pertenecen al Subsistema de Administración Informática y Configuración Inicial.

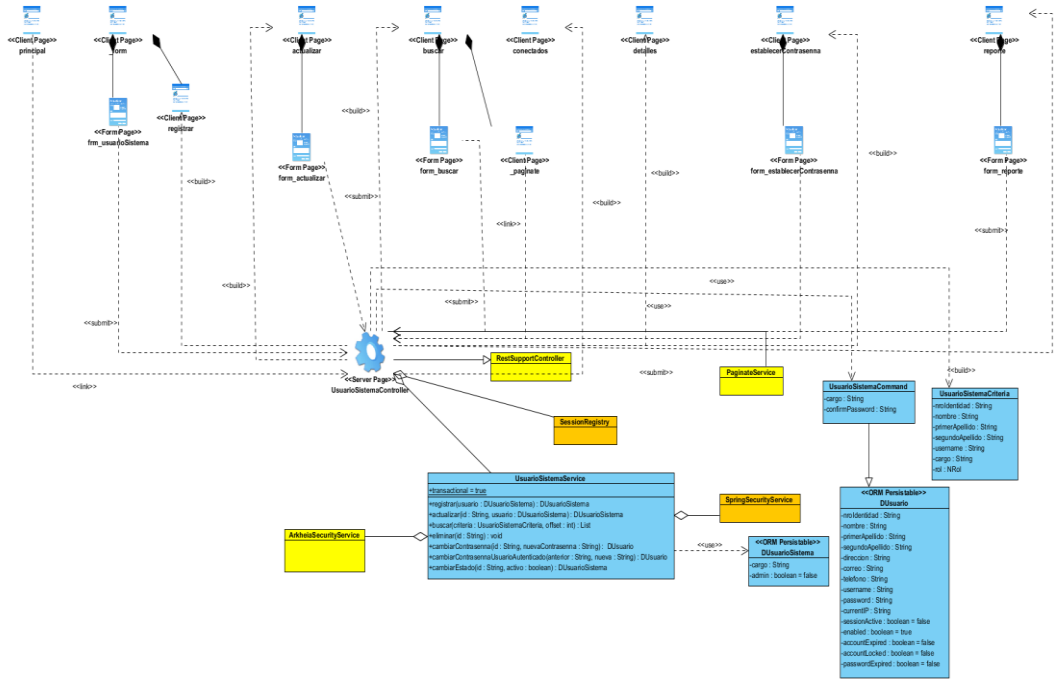

Imagen 13 Diagrama de clase del diseño web usuarioSistema.

Para ver los restantes diagramas de clases del diseño web, remitirse al **(Anexo IV)**.

2.8 Visión general de la arquitectura de software.

La arquitectura de software es de vital importancia, ya que la manera en que se estructura un sistema tiene un impacto directo sobre la capacidad de este para satisfacer los requerimientos de software. A continuación se describe el patrón de arquitectura que se utilizará en el desarrollo del subsistema.

Modelo-Vista-Controlador (MVC): es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón fue descrito por primera vez en 1979 por Trygve Reenskaug, realizando la primera implementación para Smalltalk-80[17].

El modelo representa los datos y las reglas de negocio que rigen su acceso y actualización; puede verse como una modelación de los procesos del mundo real.

Las vistas se encargan de presentar los datos obtenidos del modelo. Es responsabilidad de las vistas mantener la información actualizada, esto se puede lograr a través de peticiones de actualización al modelo o a través de notificaciones de cambio que el modelo emite (eventos).

El controlador actúa como un traductor de las acciones que se realizan en las vistas en las operaciones que ocurren en el modelo. Las acciones realizadas por el modelo desencadenan la activación de procesos de negocio o cambian el estado del modelo. Sobre la base de las

[17] Lenguaje de programación orientado a objeto con tipos dinámicos.

acciones del usuario y los resultados del modelo, el controlador responde mediante la selección de la vista apropiada. (13)

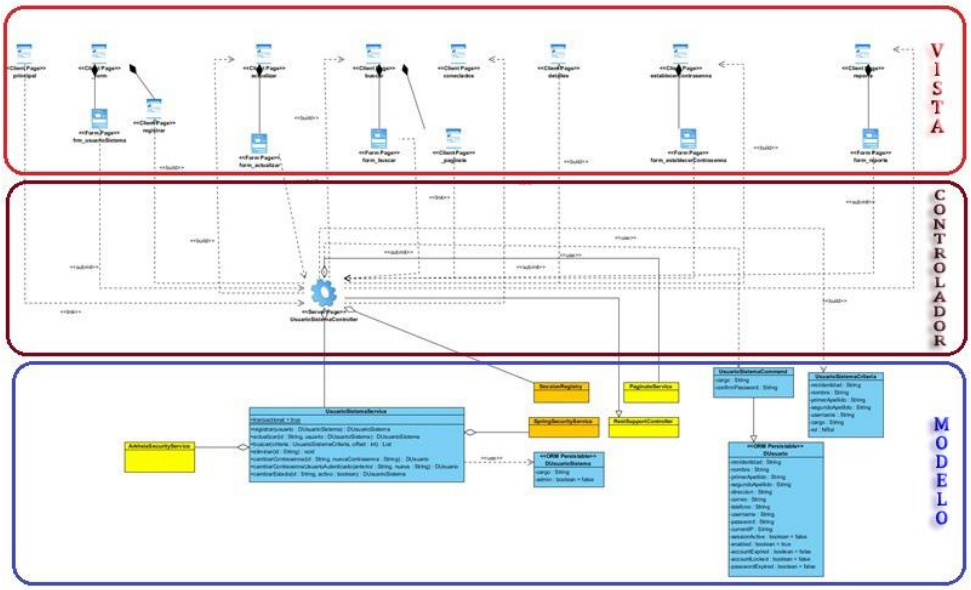

Imagen 14 Diagrama de la Arquitectura Modelo_Vista_Controlador.

2.9 Patrones de diseño.

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes encontrados en el desarrollo de software. En el desarrollo del subsistema se utilizarán los siguientes patrones:

Controlador: es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases del negocio según el método llamado. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación para aumentar la reutilización de código y a la vez tener un mayor control. (30)

El patrón *Controlador* se utilizará como intermediario entre las peticiones de los usuarios y la capa de negocio, se recomienda dividir los eventos del sistema en el mayor número de controladores para poder aumentar la cohesión y disminuir el acoplamiento.

Fachada: es un patrón que provee una interfaz unificada para un conjunto de interfaces de un subsistema. Define una interfaz de alto nivel que hace el subsistema fácil de usar. (31) (32)

El patrón *Fachada* se utilizará para proveer un único punto de acceso de los controladores a las funcionalidades del negocio, permitiendo que sean más reusables y fáciles de adaptar, logrando el desacople entre la capa de presentación y la capa de negocio.

Inversión del Control (IoC): es un patrón de diseño pensado para permitir un menor acoplamiento entre componentes de una aplicación y fomentar así nuevamente su uso. El

patrón *IoC* aplica un principio de diseño denominado principio de Hollywood (No nos llames, nosotros te llamaremos). (33)

El patrón *IoC* se utilizará para definir las dependencias de los controladores con los servicios de la capa de negocio, así desacoplar las clases de sus dependencias de manera de que las mismas puedan ser remplazadas o actualizadas con muy pocos o casi ningún cambio en el código fuente de sus clases.

Inyección de Dependencia: es un patrón que propone no instanciar las dependencias explícitamente en su clase, sino declarativamente expresarlas en la definición de la clase. La esencia de la inyección de las dependencias es contar con un componente capaz de obtener instancias válidas de las dependencias del objeto y pasárselas durante la creación o inicialización del objeto. (33)

Con la ayuda del mecanismo que brinda Grails de inyección de dependencias por convención de nombre, se definirá las dependencias de los controladores con los servicios de la clase de negocio, a partir del identificador de las clases.

2.10 Diseño de la Base de Datos.

Una base de datos correctamente diseñada permite obtener acceso a información exacta y actualizada. Puesto que un diseño correcto es esencial para lograr los objetivos fijados, es importante emplear el tiempo que sea necesario en aprender los principios de un buen diseño, ya que, es mucho más probable que la base de datos termine adaptándose a sus necesidades y pueda modificarse fácilmente. (34)

Un buen diseño de base de datos es aquel que:

- ✓ Divide la información en tablas para reducir los datos redundantes.
- ✓ Ayuda a garantizar la exactitud e integridad de la información.
- ✓ Satisface las necesidades de procesamiento de los datos y de generación de informes.

2.10.1 Diagrama de clases persistentes.

Las clases persistentes son aquellas que tienen durabilidad en el tiempo. Son el punto de partida para la creación del modelo físico de datos.

A continuación el diagrama de clases persistentes correspondiente al subsistema:

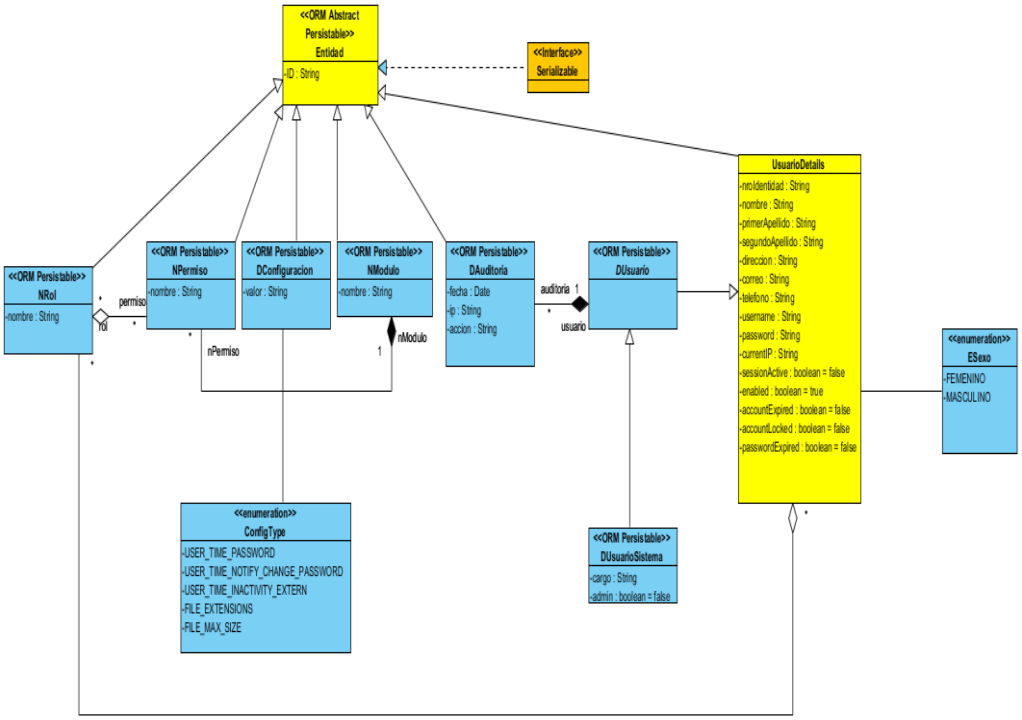

Imagen 15 Diagrama de clases persistentes.

DUsuario: entidad que almacena los datos del usuario del sistema.

DUsuarioSistema: entidad que almacena los datos de los usuarios que trabajan con el sistema.

DUsuarioDetails: clase base de los datos del usuario.

DAuditoría: entidad que almacena la traza de las acciones de los usuarios en el sistema.

NMódulo: entidad que almacena los datos del módulo del sistema.

NRol: entidad que agrupa los permisos para crear roles de usuarios del sistema.

NPermiso: entidad que almacena los permisos para el acceso al sistema.

DConfiguración: entidad que almacena los valores de las configuraciones generales.

Entidad: clase base de las entidades del subsistema que define el identificador como una cadena de caracteres.

2.10.2 Modelo de datos de clases persistentes.

Luego de realizar el diagrama de clases persistentes se representa el modelo de datos de clases persistentes representando las tablas, atributos y relaciones de la base de datos.

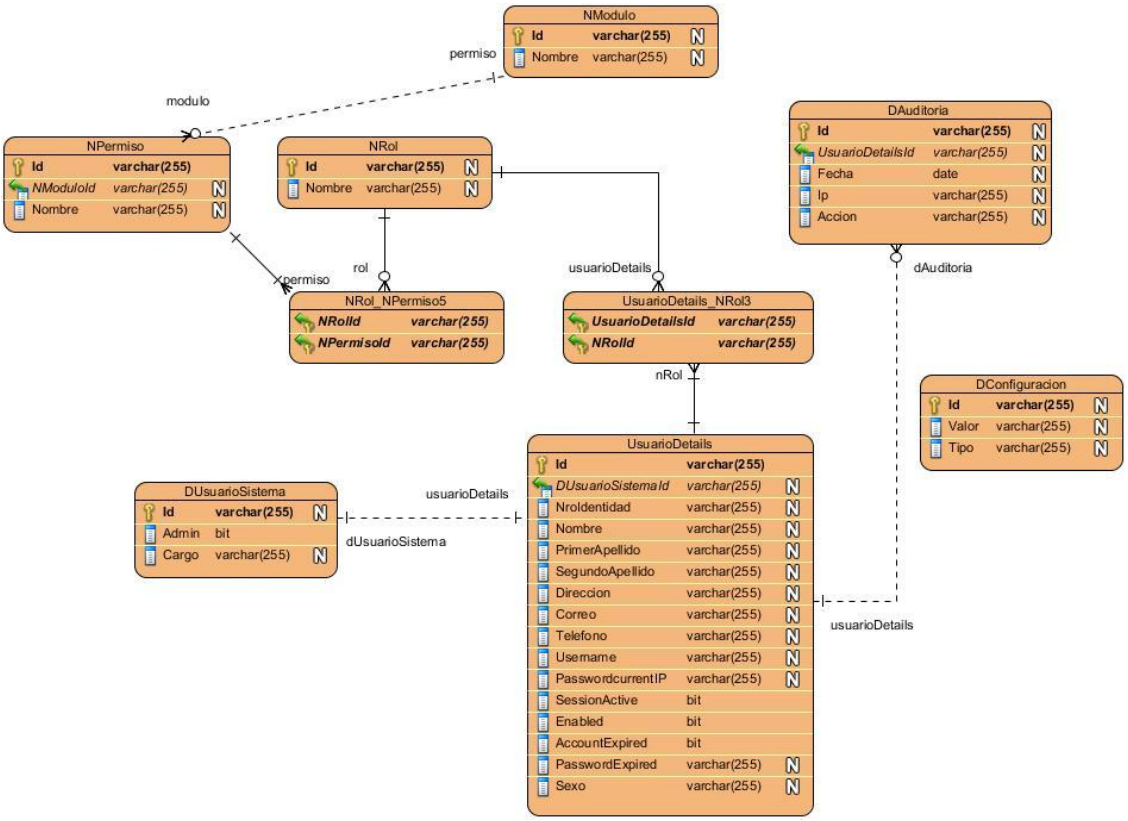

Imagen 16 Modelo de datos de clases persistentes.

2.11 Descripción del subsistema propuesto.

Luego de concluir el Análisis y Diseño se divide la implementación del subsistema en dos módulos. Cada uno de los módulos se regirán por el patrón de arquitectura Modelo-Vista-Controlador, definiendo los componentes principales de la misma en vista, controlador, servicios y dominio.

A continuación se describen las responsabilidades de cada uno de los componentes:

- ✓ Vista: es la encargada de mostrar los datos a los usuarios del sistema
- ✓ Controlador: es el responsable de atender las peticiones de las vistas, debe tener acceso al negocio mediante un único servicio.
- ✓ Servicio: es donde se implementa la lógica de negocios y brinda una interfaz única de acceso al negocio a los controladores. Este componente, de ser necesario, utiliza las funcionalidades de otros servicios para realizar una tarea.
- ✓ Dominio: define las clases que persiste la información en la base de datos.

El módulo de Configuración Inicial utiliza la capa Modelo de los subsistemas *Organización e Incorporación Documental* y *Almacenamiento y Conservación*, además del módulo *Administración Informática*, para implementar sus funcionalidades.

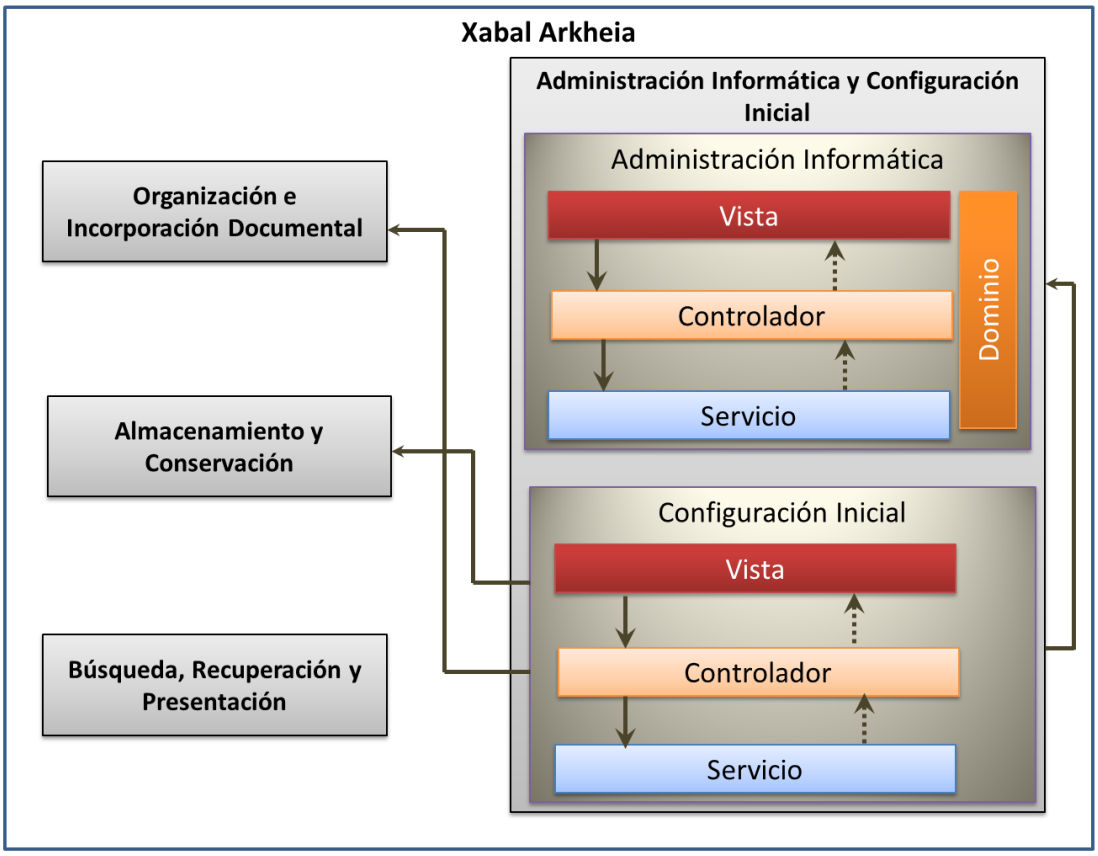

Imagen 17 Sistema propuesto.

2.12 Conclusiones.

En este capítulo se han identificado los requerimientos de software que tendrá el subsistema que se describe en el presente trabajo de diploma, además de las reglas del negocio que se deben tener en cuenta en su desarrollo.

La arquitectura Modelo-Vista-Controlador definida permitirá un desacople entre las distintas partes del sistema, permitiendo que en un futuro se puedan realizar cambios en el código fuente, según las necesidades de los clientes, sin que se afecten las capas no involucradas.

El modelado del diagrama de casos de uso del sistema se realizó utilizando los patrones correspondientes, la descripción detallada de cada uno en un lenguaje sencillo y su clasificación según la prioridad requerida. Además se elaboraron los diagramas de clases del análisis y del diseño, logrando una guía para la implementación de la solución propuesta.

Capítulo 3

3 Implementación y Pruebas

En este capítulo se describe la implementación del subsistema. Se realizará el modelo de implementación para una mejor descripción de la solución propuesta, describiéndose las pautas de codificación y la organización de los ficheros. También se validarán las funcionalidades a partir de las pruebas de caja blanca y caja negra al subsistema desarrollado, además de las pruebas de usabilidad de los requerimientos no funcionales.

3.1 Diagrama de Componentes.

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo código fuente, binario y ejecutable, pueden ser utilizados para modelar sistema de software de cualquier tamaño y complejidad.

A continuación se muestran los diagramas de componentes del módulo Administración Informática agrupados por conceptos:

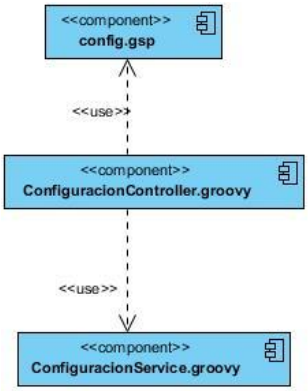

Imagen 18 Diagrama de Componentes Actualizar Configuración.

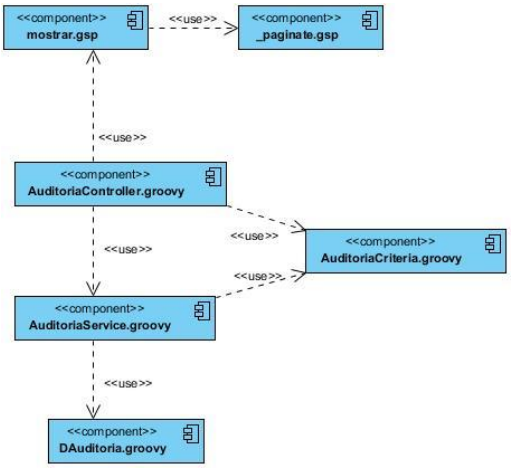

Imagen 19 Diagrama de Componentes Auditoría.

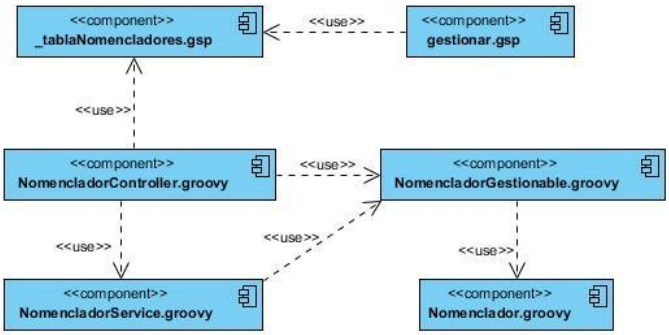

Imagen 20 Diagrama de Componentes Nomenclador.

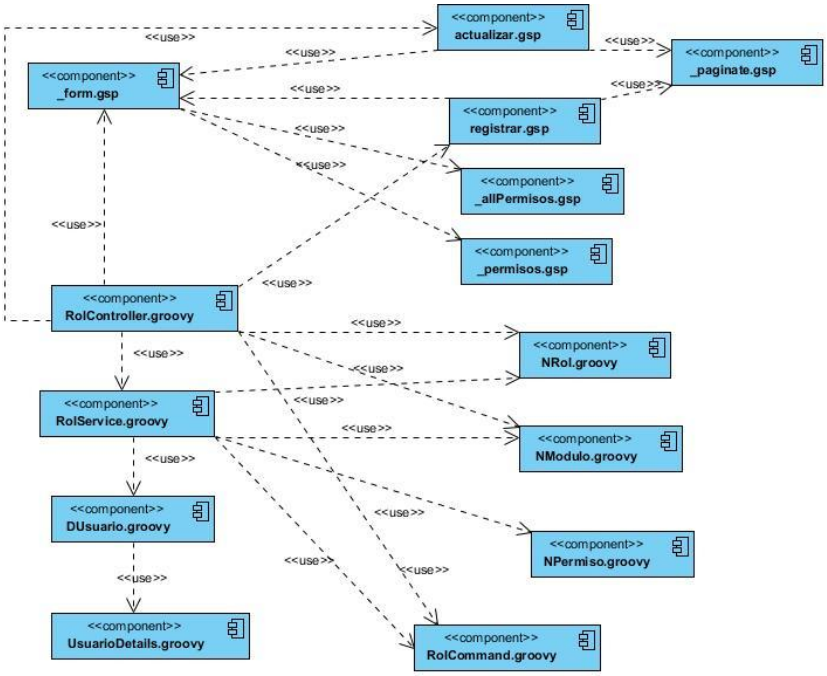

Imagen 21 Diagrama de Componentes Rol.

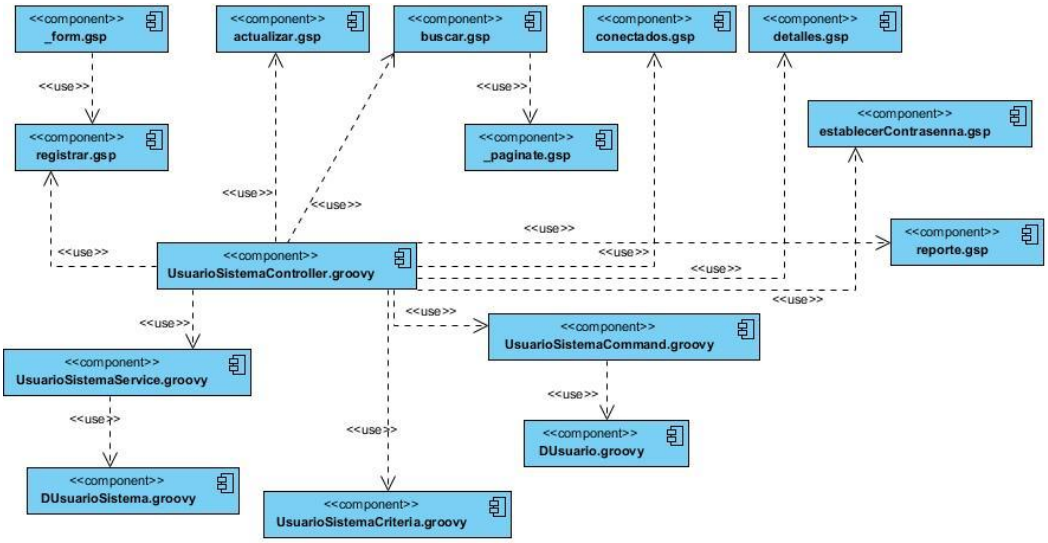

Imagen 22 Diagrama de Componentes usuarioSistema.

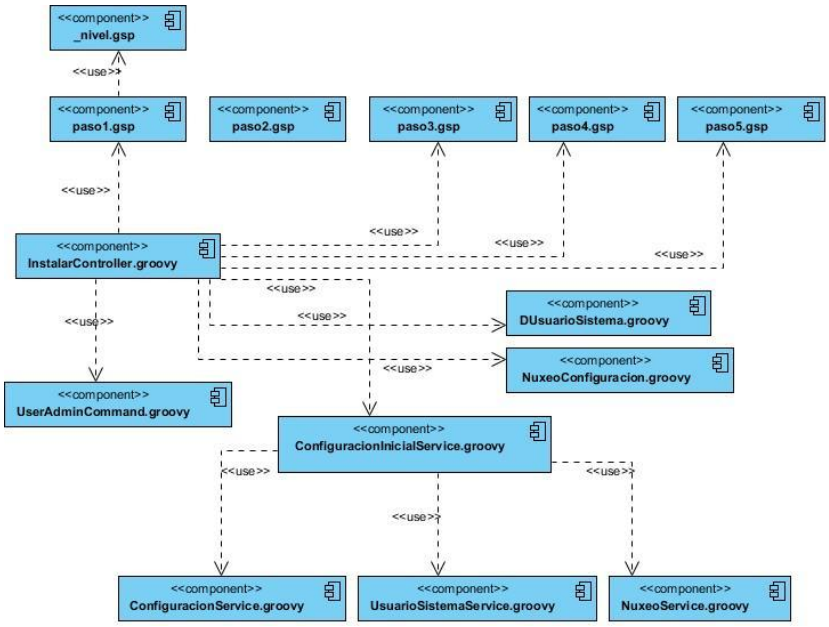

Imagen 2 1 Diagrama de Componentes Configuración Inicial.

3.2 Modelo de Despliegue.

El diagrama de despliegue muestra las relaciones físicas de los distintos nodos que componen un sistema. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. (35)

El diagrama de despliegue del sistema está conformado por un servidor web donde se encontrará desplegado el Sistema Xabal Arkheia, un servidor con el gestor de documentación Nuxeo y un servidor con el gestor de base de datos PostgreSQL. Las PC clientes mediante un navegador web se comunican a través del protocolo http con el Sistema Xabal Arkheia y presentan una impresora para imprimir los reportes generados por el subsistema, si el usuario lo desea.

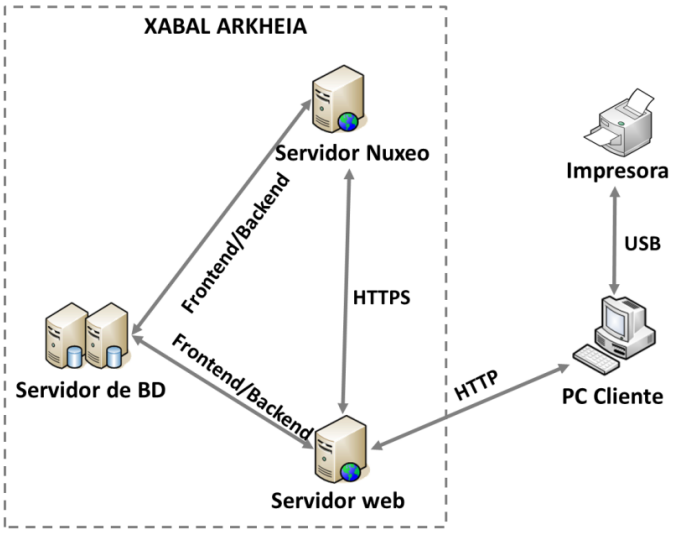

Imagen 23 Diagrama de Despliegue.

A continuación se realizará una descripción de los nodos y artefactos que componen el diagrama de despliegue:

Servidor Nuxeo: nodo que representa el servidor de gestión documental Nuxeo, donde se almacenan los archivos asociados a un documento.

Servidor Web: este nodo contiene el servidor web Apache Tomcat 7.23, en el que estará desplegado el Sistema Xabal Arkheia.

Servidor de BD: representa el servidor de Base de Datos PostgreSQL 9.0, en el cual se persistirá la información sensible manejada por el sistema.

PC Cliente: representa un conjunto de pc a través de las cuales los usuarios pueden actualizar y consultar la información que se encuentra en el servidor web.

Impresora: representa un artefacto el cual será utilizado para imprimir los reportes o estadísticas generadas por el sistema.

Los conectores están definidos por los protocolos de comunicación entre los artefactos y los nodos:

HTTP (Protocolo de Transferencia de Hipertexto): es un protocolo de red para publicar páginas web o HTML, siendo la base sobre la cual está fundamentado Internet, o la WWW[18]. (36)

HTTPS (Protocolo de Transferencia de Hipertexto Seguro): es una versión segura del protocolo HTTP que implementa un canal de comunicación seguro y basado en SSL (Secure Socket Layers) entre el navegador del cliente y el servidor HTTP. El cliente es el navegador web que realiza las peticiones a las que el servidor se encarga de dar respuesta. (37)

USB (Universal Serial Bus): consiste en una norma desarrollada tanto por industrias de computación como de telecomunicaciones. USB permite adjuntar dispositivos periféricos al ordenador rápidamente, sin necesidad de reiniciarlo ni de volver a configurar el sistema. (38)

Frontend/Backend: protocolo de comunicación para el intercambio de datos entre el cliente y el servidor de Base de Datos PostgreSQL.

3.3 Pautas de Codificación.

Las pautas de codificación son de gran importancia para el desarrollo de la aplicación. Estas rigen estándares a seguir por todo el equipo de desarrollo en la escritura del código fuente.

Para el desarrollo del Sistema Xabal Arkheia se definió las siguientes reglas:

- ✓ Nombre de variables: se utiliza el estilo camello.

Ejemplo: nombreCompleto, primerApellido, segundoApellido.

[18] World Wide Web.

- ✓ Clases del dominio: las clases que son nomencladoras van a tener el prefijo “N” y las clases de datos “D”.

Ejemplo: NRol, NPermiso, DusuarioSistema, DAuditoria.

- ✓ Clases controladoras: estas clases terminarán con el sufijo Controller.

Ejemplo: UsuarioSistemaController, AuditoriaController, RolController.

- ✓ Clases servicios: estas clases terminarán con el sufijo Service.

Ejemplo: UsuarioSistemaService, AuditoriaService, RolService.

- ✓ Fichero de la capa vista: se escribirán con minúscula los nombres de los ficheros.

Ejemplo: registrar.gsp.

3.4 Organización de los ficheros.

Tener una buena organización del código fuente permite una mejor colaboración dentro del equipo de desarrollo. Además de establecer el acceso a los desarrolladores según su rol, a las carpetas del módulo donde se encuentran trabajando.

Para el desarrollo del sistema se adoptó la estructura de ficheros propuesta por el framework Grails, por lo que el código fuente estará organizado de la siguiente forma:

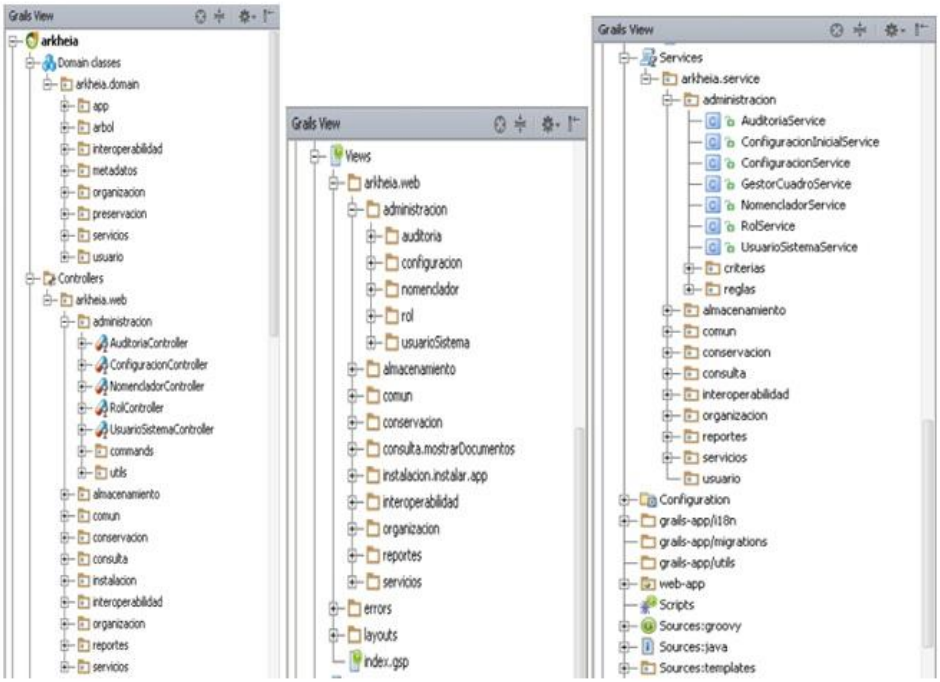

Imagen 24 Organización de los ficheros.

Las clases del dominio estarán dentro de la carpeta domain, esta carpeta está compuesta por un paquete llamado *arkheia.domain*, en el que se agrupará el dominio por concepto.

Las clases controladoras se encontrarán dentro de la carpeta controllers y las clases del subsistema estarán dentro del paquete *arkheia.web.administracion*.

Las clases del servicio se definirán dentro de la carpeta services y las clases del subsistema estarán dentro del paquete *arkheia.service.administracion*.

Los ficheros de las vistas: estarán dentro de la carpeta *web-app/arkheia/web/administracion*. Dentro de esta, estarán agrupadas las vistas de acuerdo al nombre de la clase controladora a la que pertenece.

3.5 Módulos implementados.

En el subsistema por la cantidad de funcionalidades que presenta se decide crear dos módulos que agrupan funcionalidades comunes, teniendo en cuenta el momento en que se ejecuta cada una de ellas. Los dos módulos identificados son:

Administración Informática: este módulo se encarga de la gestión de los roles, estos al ser definidos totalmente por el administrador hace que el módulo sea adaptable a los cambios que se realicen en la institución con respecto al acceso de la información. Otra de las características que presenta es la gestión de los usuarios del sistema, permite tener un control sobre las personas que trabajan dentro de la entidad conociendo en todo momento las acciones realizadas por ellos. También la gestión de los nomencladores permite establecer, modificar y eliminar valores significativos del sistema. La configuración general lleva la gestión de valores constantes que influyen en algunos procesos del negocio.

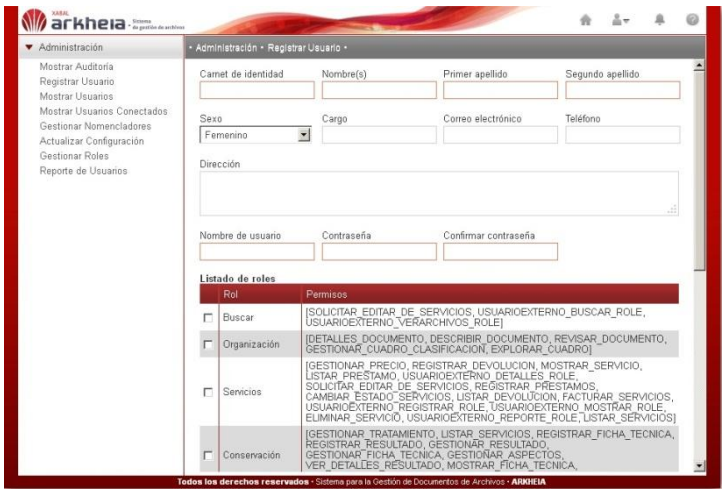

Imagen 25 Funcionalidad Registrar Usuario del módulo Administración Informática.

De forma significativa es importante destacar que en este módulo para realizar la auditoría de las acciones de los usuarios, se usa el mecanismo de filtros que brinda Grails. En este caso, se interceptan todas las acciones realizadas por el usuario y se guarda en la base de datos la

fecha, hora y dirección IP desde donde se realizó la acción. Este mecanismo permite llevar un control de quién, dónde y cuándo se accede a la información.

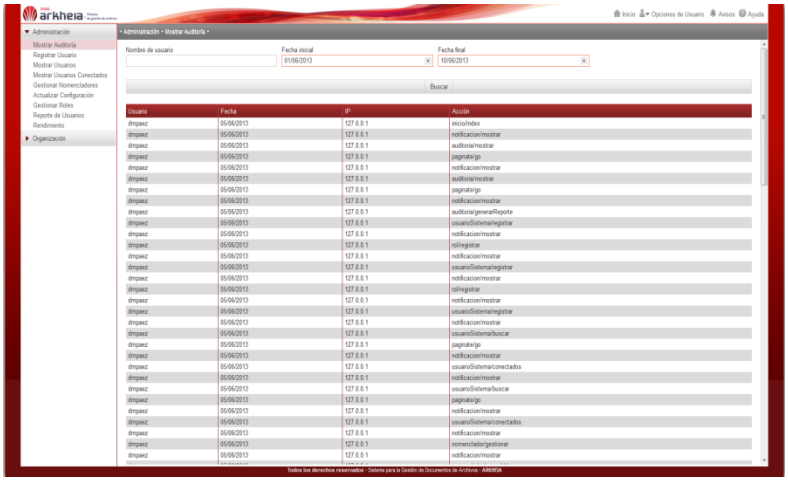

Imagen 26 Funcionalidad Mostrar Usuarios del módulo Administración Informática.

Configuración Inicial: en este módulo se configura el cuadro de clasificación y estructura de ubicación física, lo que posibilita que el sistema sea más adaptable a nuevos clientes con especificaciones propias en el trabajo con la organización física y lógica de sus documentos. También configura el acceso al servidor Nuxeo, permitiendo la eliminación de ficheros de configuración donde el usuario pudiese cometer errores en el proceso de editarlos. Establece los valores de configuración general y crea la cuenta con los permisos de administración, permitiendo tener un usuario con acceso a todas las funcionalidades de administración del sistema.

El módulo se implementó haciendo uso del mecanismo de WebFlows que brinda Grails, para guiar al usuario de forma intuitiva por las acciones que debe hacer. Lleva un flujo de páginas desde el inicio hasta el final de la configuración del sistema, teniendo la funcionalidad en todo momento de regresar a las acciones anteriores que previamente ha configurado.

Imagen 27 Configurar el Cuadro de Clasificación del módulo Configuración Inicial.

Imagen 28 Datos para configurar el Cuadro de Clasificación del módulo Configuración Inicial.

Imagen 29 Configurar la Estructura de Ubicación Física del módulo Configuración Inicial.

Imagen 30 Configurar acceso a Nuxeo del módulo Configuración Inicial.

Imagen 31 Configurar Parámetros Generales del módulo Configuración Inicial.

Imagen 32 Configurar la Cuenta de Administración del módulo Configuración Inicial.

3.6 Pruebas.

Las pruebas son una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema.

La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación.

3.6.1 Pruebas de caja negra.

Las pruebas de caja negra son pruebas funcionales sin acceso al código fuente de las aplicaciones, se trabaja con entradas y salidas.

Los casos de prueba que se le realizaron al sistema corresponden a los casos de uso que necesitan la entrada de datos, cada uno está dividido en tantas secciones como escenarios tenga el caso de uso, incluyendo los flujos de datos alternos. Con estos casos de prueba se pretende mostrar que todas las funciones del software son operativas, que las entradas realizadas se aceptan de forma adecuada y que se produce una salida correcta. (39)

En las pruebas de liberación realizadas por CALISOFT[19] hubo cuatro iteraciones, la siguiente gráfica arroja los resultados obtenidos:

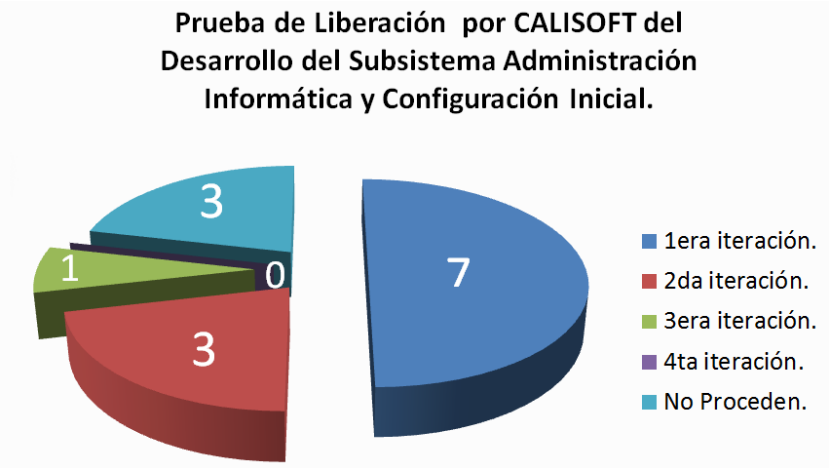

Imagen 33 Gráfica del análisis de las pruebas de liberación realizadas por CALISOFT al Subsistema de Administración Informática y Configuración Inicial.

Casos de Prueba del Caso de Uso Gestionar Usuario del módulo Administración Informática.

Entrada	Salida	Resultados
El administrador en la sección eliminar usuario presiona el botón “Eliminar”.	El sistema muestra un mensaje preguntándole al administrador si está seguro de eliminar el usuario.	Satisfactorio.
El administrador en la sección eliminar usuario luego de presionar el botón “Eliminar” presiona el botón “Si”.	El sistema elimina el usuario seleccionado y actualiza el listado de usuarios mostrado.	Satisfactorio.
El administrador en la sección eliminar usuario luego de presionar el botón “Eliminar” presiona el botón “No”.	El sistema cancela la operación en proceso y retorna a la página.	Satisfactorio.

Tabla 19 Sección Eliminar Usuario.

[19] Centro Nacional de Calidad de Software.

Entrada	Salida	Resultados
El administrador en la sección "Registrar Usuario" deja datos obligatorios vacíos.	El sistema muestra un mensaje indicando que faltan datos por introducir y señala los datos que faltan.	Satisfactorio.
El administrador en la sección "Registrar Usuario" introduce datos incorrectos.	El sistema muestra un mensaje indicando que existen datos incorrectos y señala los datos que están incorrectos.	Satisfactorio.
El administrador en la sección "Registrar Usuario" inserta un usuario registrado anteriormente con el mismo Carnet de Identidad.	El sistema muestra un mensaje indicando que el Carnet de Identidad del usuario ya se encuentra almacenado en la Base de Datos.	Satisfactorio.
El administrador en la sección "Registrar Usuario" inserta un usuario registrado anteriormente con el mismo Nombre de Usuario.	El sistema muestra un mensaje indicando que el Nombre de Usuario ya se encuentra almacenado en la Base de Datos.	Satisfactorio.
El administrador en la sección "Registrar Usuario" selecciona la opción "Cancelar".	El sistema cancela la operación en proceso y retorna a la misma página con los datos vacíos.	Satisfactorio.

Tabla 20 Sección Registrar Usuario.

Entrada	Salida	Resultados
El administrador en la sección "Ver Detalles de Usuario" selecciona la opción "Atrás".	El sistema retorna a la página que le dio origen.	Satisfactorio.

Tabla 21 Sección Ver Detalles de Usuario.

Entrada	Salida	Resultados
El administrador selecciona la opción "Activar Usuario".	El sistema muestra el usuario como activado.	Satisfactorio.

Tabla 22 Sección Activar Usuario.

Entrada	Salida	Resultados
El administrador selecciona la opción "Desactivar Usuario".	El sistema muestra el usuario como desactivado.	Satisfactorio.

Tabla 23 Sección Desactivar Usuario.

Entrada	Salida	Resultados
El administrador en la sección	El sistema muestra un mensaje	Satisfactorio.

"Modificar Usuario" deja datos obligatorios vacíos	indicando que faltan datos por introducir y señala los datos que faltan.	
El administrador en la sección "Modificar Usuario" introduce datos incorrectos.	El sistema muestra un mensaje indicando que existen datos incorrectos y señala los datos que están incorrectos.	Satisfactorio.
El administrador en la sección "Modificar Usuario" inserta un usuario registrado anteriormente con el mismo Carnet de Identidad.	El sistema muestra un mensaje indicando que el Carnet de Identidad del usuario ya se encuentra almacenado en la Base de Datos.	Satisfactorio.
El administrador en la sección "Modificar Usuario" inserta un usuario registrado anteriormente con el mismo Nombre de Usuario.	El sistema muestra un mensaje indicando que el Nombre de Usuario ya se encuentra almacenado en la Base de Datos.	Satisfactorio.
El administrador en la sección "Modificar Usuario" selecciona la opción "Cancelar".	El sistema cancela la operación en proceso y retorna a la misma página con los datos vacíos.	Satisfactorio.

Tabla 24 Sección Modificar Usuario.

Para ver los restantes Casos de Pruebas remitirse al **(Anexo V)**.

3.6.2 Pruebas de caja blanca.

Las pruebas de caja blanca son pruebas con acceso al código. Con este tipo de prueba se comprueban los caminos lógicos del software. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coincide con el esperado.

Se aplicará el método de prueba de camino básico, el cual permite obtener una medida de la complejidad lógica de un diseño y usar esta medida como guía para la definición de un conjunto básico. La idea es derivar casos de prueba a partir de un conjunto dado de caminos independientes por los cuales puede circular el flujo de control. Para obtener dicho conjunto de caminos independientes se construye el grafo de flujo asociado y se calcula su complejidad ciclomática. (40)

Los pasos que se siguen para aplicar esta técnica son:

- ✓ A partir del código fuente se dibuja el grafo de flujo asociado.
- ✓ Se calcula la complejidad ciclomática del grafo.
- ✓ Se determina un conjunto básico de caminos independientes.

- ✓ Se preparan los casos de prueba que obliguen a la ejecución de cada camino del conjunto básico.

Un Grafo de Flujo está compuesto por tres componentes fundamentales:

- ✓ Nodo (Cada círculo representado se denomina nodo del Grafo de Flujo).
- ✓ Aristas (Son las flechas del grafo y representan el flujo de control).
- ✓ Regiones (Área delimitada por las aristas y nodos, también se incluye el área exterior del grafo).

La complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y permite un límite superior para el número de pruebas que se deben realizar asegurando que se ejecute cada sentencia al menos una vez. Un camino independiente es cualquier camino del programa que introduce por lo menos una nueva condición y se debe mover por una arista que no haya sido recorrida anteriormente.

Existen tres formas fundamentales de calcular la complejidad:

- ✓ El número de regiones del grafo de flujo coincide con la complejidad ciclomática (V (G)).
- ✓ V (G) = A-N+2

A: Número de aristas del grafo.

N: Número de nodos.

- ✓ V (G) = P+1

P: Número de nodos predicado contenidos en el grafo G.

```
DUsuarioSistema actualizar(String id, DUsuarioSistema usuario) {
    def actual = DUsuarioSistema.get(id)     1
    if (!actual)  2
        return null  3

    def query = DUsuarioSistema.where { id != id} 4
    if (query.findByUsername(usuario.username)){  5
        throw new ReglaNegocioViolada("administracion.domain.usuario.username.duplicate.error") 6
    }

    if (query.findByNroIdentidad(usuario.nroIdentidad)){7
        throw new ReglaNegocioViolada("administracion.domain.usuario.nroIdentidad.duplicate.error"8
    }

    arkheiaSecurityService.validarContrasenna(usuario.password)

    actual.nombre = usuario.nombre
    actual.primerApellido = usuario.primerApellido
    actual.segundoApellido = usuario.segundoApellido
    actual.direccion = usuario.direccion                    9
    actual.cargo = usuario.cargo
    actual.sexo = usuario.sexo
    actual.correo = usuario.correo
    actual.username = usuario.username
    if (usuario.password)10
        actual.password = springSecurityService.encodePassword(usuario.password)11
    actual.enabled = usuario.enabled  12

    new ArrayList<NRol>(actual.roles).each { 13
        actual.removeFromRoles(it)   14
    }

    usuario.roles.each { 15
        actual.addToRoles(it) 16
    }

    actual.save(flush: true)17
}18
```

Imagen 34 Código fuente de la funcionalidad Actualizar Usuario.

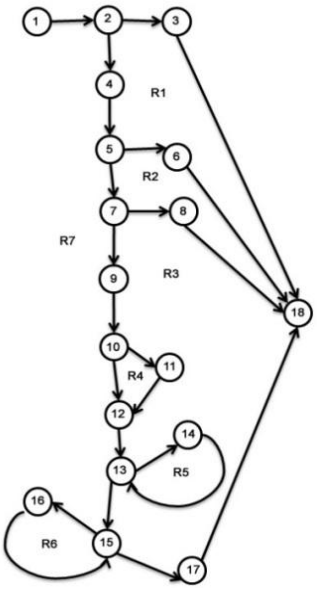

Imagen 35 Representación del Grafo de Flujo del código fuente de la funcionalidad Actualizar Usuario.

Complejidad ciclomática del Grafo de Flujo.

V (G) = A-N+2

V (G) = 23-18+2

V (G) = 7.

La complejidad ciclomática del Grafo de Flujo es de 7.

Los caminos independientes son:

Camino 1: 1-2-3-18.

Camino 2: 1-2-4-5-6-18.

Camino 3: 1-2-4-5-7-8-18.

Camino 4: 1-2-4-5-7-9-10-12-13-14-13-15-16-15-17-18.

Los casos de pruebas para cada camino se muestran a continuación:

Camino 1: 1-2-3-18.

Escoger algún id que no exista en la tabla DUsuarioSistema.

id = "4".

Camino 2: 1-2-4-5-6-18.

Seleccionar un id y crear una instancia de usuarioSistema que contenga un username que ya exista en la base de datos.

id = "1" y usuario = new DUsuarioSistema (username: "admin").

Camino 3: 1-2-4-5-7-8-18.

Seleccionar un id y crear una instancia de usuarioSistema que contenga un nroIdentidad que ya exista en la base de datos.

id = "1" y usuario = new DusuarioSistema (nroIdentidad: "85022501231").

Camino 4: 1-2-4-5-7-9-10-12-13-14-13-15-16-15-17-18.

Seleccionar un id que exista en la base de datos, crear un usuario del sistema que contenga un username y nroIdentidad que no existan en la base de datos, el password que sea distinto de null y que tenga una lista de roles.

id = "1" y usuario = new DUsuarioSistema (nroIdentidad: "85022501231", username: "admin01", password:"secret", roles: [administrador]).

A continuación usando la herramienta de pruebas de integración que brinda Grails se comprobó que se ejecutan correctamente cada uno de los caminos especificados anteriormente, validando así que se ejecute cada parte del software:

```
/*
* Camino 1: 1-2-3-18
* Se selecciona un id que no exista en la base de datos
* */
void testCamino1() {
    String id = "4"
    def result = usuarioSistemaService.actualizar(id, null)

    assertNull(result)
}

/*
* Camino 2: 1-2-4-5-6-18
* Seleccionar un id y crear una instancia de usuarioSistema que contenga un username que ya exista en la base de datos
* */
void testCamino2() {
    String id = "8abb85843df0afb9013df0b53204003e"
    DUsuarioSistema usuario = new DUsuarioSistema(username: "test")
    ReglaNegocioViolada excepcion
    try {
        usuarioSistemaService.actualizar(id, usuario)
    } catch (ReglaNegocioViolada e) {
        excepcion = e
    }

    assertNotNull(excepcion)

    assertEquals("administracion.domain.usuario.username.duplicate.error", excepcion.message)
}
```

Imagen 36 Validación al Camino 1 y Camino 2.

```
/*
* Camino 3: 1-2-4-5-7-8-18
* Seleccionar un id y crear una instancia de usuarioSistema que contenga un nroIdentidad que ya exista en la base de datos.
* */
void testCamino3() {
    String id = "8abb85843df0afb9013df0b53204003e"
    DUsuarioSistema usuario = new DUsuarioSistema(username: "admin", nroIdentidad: "87022420948")
    ReglaNegocioViolada excepcion
    try {
        usuarioSistemaService.actualizar(id, usuario)
    } catch (ReglaNegocioViolada e) {
        excepcion = e
    }

    assertNotNull(excepcion)

    assertEquals("administracion.domain.usuario.nroIdentidad.duplicate.error", excepcion.message)
}
```

Imagen 37 Validación al Camino 3.

```
/*
* Camino 4: 1-2-4-5-7-9-10-12-13-14-13-15-16-15-17-18
* Seleccionar un id que exista en la base de datos, crear un usuario del sistema que contenga un username y
* nroIdentidad que no existan en la base de datos, el password que sea distinto de null y que tenga una lista de roles
* */
void testCamino4() {
    String id = "8abb85843df0afb9013df0b53204003e"
    DUsuarioSistema usuario = DUsuarioSistema.get(id)
    usuario.password = "1234567"

    def result = usuarioSistemaService.actualizar(id, usuario)

    assertNotNull(result)
}
```

Imagen 38 Validación al Camino 4.

Imagen 39 Reporte generado por Grails al ejecutar el código para la validación de los caminos independientes.

3.6.3 Pruebas de Usabilidad

La facilidad de uso es un punto de vital importancia porque un sistema puede ser rechazado por los usuarios si no encuentran sencillo su uso.

Para la validación de los requerimientos no funcionales de usabilidad se utilizó la Lista de Chequeo Usabilidad de Sitios Web, elaborada por el centro de calidad CALISOFT. Esta cuenta con varios puntos, los cuales serán clasificados antes de ser aplicados, teniendo en cuenta su pertinencia, además de tener un punto donde se referencia las observaciones en cuanto al aspecto a evaluar **(Ver Anexo VI).**

A continuación se muestra una gráfica con los resultados de la primera iteración de las pruebas de usabilidad:

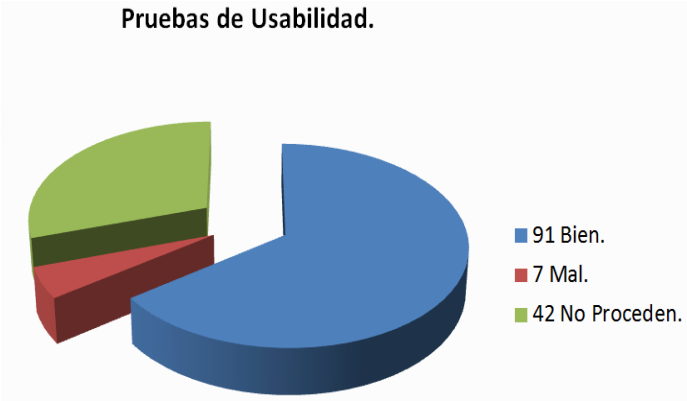

Imagen 40 Resultado de las Pruebas de Usabilidad de los Requisitos No Funcionales en la primera iteración.

En la segunda iteración de las pruebas de usabilidad se erradicaron las 7 no conformidades detectadas anteriormente en la primera iteración.

3.7 Conclusiones.

En el presente capítulo se elaboraron diagramas del flujo de trabajo de Implementación que permiten un mejor entendimiento de la construcción de la aplicación. El administrador y el usuario son los que tienen acceso sobre la aplicación y a los cuáles se les asignaron determinados privilegios para garantizar la seguridad del sistema. Además se realizaron pruebas que permitieron verificar su buen funcionamiento.

Conclusiones

En el desarrollo de la presente investigación se recoge lo referente al proceso de desarrollo del Subsistema de Administración Informática y Configuración Inicial, logrando dar respuesta al problema planteado. El análisis de algunos sistemas informáticos para la gestión de archivos permitió conocer sus tendencias actuales, incorporando nuevas funcionalidades al desarrollo del subsistema en el área de gestión de usuarios, auditoría, configuración de parámetros generales y administración de recursos. Luego de una caracterización de los modelos de control de acceso, se seleccionó el modelo de Control de Acceso Basado en Roles (RBAC) por brindar una flexibilidad en la asignación de permisos a los usuarios. Los lenguajes, tecnologías y herramientas, permitieron el desarrollo del subsistema construido sobre bases sólidas y en un entorno bien definido. El uso de la metodología de desarrollo RUP, guió el proceso de desarrollo, asegurando la calidad del producto.

Además se realizó el análisis y diseño del Subsistema de Administración Informática y Configuración Inicial, obteniendo como resultado los artefactos: diagramas de casos de uso, modelo de análisis, modelo de diseño, diagramas de componentes, diagrama de clases persistentes y el modelo de datos, definiendo los patrones de arquitectura y diseño utilizados en el desarrollo del subsistema.

Se desarrolló la implementación de las funcionalidades del subsistema, validándose mediante pruebas de caja negra y pruebas de caja blanca, permitiendo comprobar el adecuado funcionamiento de los módulos de Administración Informática y Configuración Inicial y la liberación del subsistema por CALISOFT. También se realizó la validación de los requerimientos no funcionales de usabilidad, utilizándose la Lista de Chequeos Usabilidad de Sitios Web, logrando calidad en el producto final.

Se puede plantear que en el transcurso de la investigación se cumplió con el objetivo general: *desarrollar el Subsistema de Administración Informática y Configuración Inicial para administrar los parámetros generales, la gestión de usuarios y la configuración de los subsistemas "Incorporación y Organización Documental" y "Almacenamiento y Conservación" del Sistema Xabal Arkheia*.

Recomendaciones

- ✓ Investigar cómo empaquetar en un plugin de grails la solución desarrollada en la investigación para que sea reutilizable en sistemas similares y para versiones futuras del sistema arkheia.
- ✓ Diseñar e implementar una capa vista que se adapte a los dispositivos móviles, posibilitando que los administradores del sistema tengan un mayor dinamismo en el desarrollo de su trabajo.
- ✓ Integrar los usuarios con un directorio LDAP.
- ✓ Gestionar el almacenamiento en disco duro de los documentos.

Bibliografía

1. **Demetrio.** La Archivología. *SURGIMIENTO DE LA ADMINISTRACIÓN DE ARCHIVOS.* [En línea] 19 de Marzo de 2009. [Citado el: 5 de Noviembre de 2012.] http://demetrio-demetrio.blogspot.com/.

2. **Arcángel Eduardo Sánchez Gómez, Martha Cristina Rondón de Rincón.** *Legislación archivística venezolana: una contribución para la consolidación de la gestión de archivos en Venezuela.* Maracaibo : versión impresa ISSN 1960-7515, mayo 2008. Enlace v.5 n.2.

3. **Hernández León, Rolando Alfredo y Coello González, Sayda.** *El proceso de investigación científica.* Ciudad de La Habana : Universitaria del Ministerio de Educación Superior, 2011. ISBN 978-959-16-1307-3.

4. **Barchini, Graciela Elisa.** *Métodos "I + D" de la Informática.* Santiago del Estero, Argentina. : s.n., 1912.

5. **Zayas, Dr. Carlos Alvarez de.** *METODOLOGIA DE LA INVESTIGACION CIENTIFICA .* Santiago de Cuba. : s.n., 1995.

6. **Falgueras, Benet Campderrich.** *Ingeniería del software.* s.l. : Editorial UOC.

7. **SOMMERVILLE, IAN.** *Ingeniería del Software.* [ed.] Miguel Martín Romo. Séptima. Madrid(España) : PEARSON EDUCACIÓN.SA, 2005. pág. 712.

8. **Grajales, Nubia Fernández.** Importancia de la Auditoría Informática en las organizaciones. [En línea] Octubre de 2005. [Citado el: 10 de Diciembre de 2012.] http://www.enterate.unam.mx/Articulos/2005/octubre/auditoria.htm.

9. **Jiménez, Ricardo Hernández.** *Administración de la función informática, una nueva profesión.* Primera Edición. México : Limusa, 2003. ISBN 968-18-6395- X.

10. **Grupo Archicentro.** SCAV. [En línea] 1986-2011. [Citado el: 2 de Diciembre de 2012.] http://www.archicentro.com/Productos.

11. **Sandra González Valdés, René Suarez Font.** *Diseño e Implementación de los módulos de Preparación de Documentos, Digitalización de Documentos y Asociación de Metadatos del Centro de Digitalización para la División de Antecedentes Penales.* La Habana, Cuba. : s.n., 2011.

12. **nuxeo.** nuxeo. [En línea] [Citado el: 3 de Diciembre de 2012.] http://www.nuxeo.com/en.

13. **Rodríguez, Frank Ernesto Verdecia.** *Sistema de Gestión Policial. Especificación de la Arquitectura.* Habana : s.n., 2009.

14. **Yerbabuena Software.** athento iDM. *Descubre Athento y las razones para confiarle tus documentos.* [En línea] [Citado el: 3 de Diciembre de 2012.] http://www.athento.com/why/#seguridad.

15. greenstone digital library software. *About Greenstone.* [En línea] Mayo de 2012. [Citado el: 10 de Diciembre de 2012.] http://www.greenstone.org/.

16. **Domínguez Martín, Gabriel A. y Cassolini, Raul A.** *Control de Acceso Basado en Roles.* 1999.

17. **U.S. Department of Commerce.** National Institute of Standards and Technology. [En línea] 11 de June de 2012. [Citado el: 09 de Junio de 2013.] http://csrc.nist.gov/groups/SNS/rbac/.

18. **Oracle.** Guía de administración del sistema: servicios de seguridad. [En línea] Agosto de 2011. [Citado el: 09 de Junio de 2013.] http://docs.oracle.com/cd/E24842_01/html/E23286/rbac-1.html. E23286.

19. **Refsnes Data.** w3schools.com. *HTML.* [En línea] 1999. [Citado el: 6 de Noviembre de 2012.] http://www.w3schools.com/html/default.asp.

20. **w3schools.** w3schools.com. [En línea] [Citado el: 8 de Febrero de 2013.] http://www.w3schools.com/css/default.asp.

21. —. w3schools.com. [En línea] [Citado el: 10 de Febrero de 2013.] http://www.w3schools.com/js/default.asp.

22. **Borrillo, Ricardo.** GENBETA:dev. *Bootstrap.* [En línea] 23 de Febrero de 2012. [Citado el: 15 de Diciembre de 2012.] http://www.genbetadev.com/desarrollo-web/disenando-tu-nuevo-proyecto-web-con-bootstrap-2-0.

23. **Angel, López.** *Java la progracmación del futuro.* Buenos Aires, Argentina : MP Ediciones S.A, 1997. ISBN 987-9131-38-X.

24. **Dierk, König, y otros, y otros.** *Groovy in Action. Second Edition.* 2009. ISBN: 9781935182443.

25. **UNIFÉ.** UNIFÉ-Universidad Femenina del Sagrado Corazón. *Hibernate.* [En línea] 2012. [Citado el: 14 de Diciembre de 2012.] www.unife.edu.pe/ing/desarrollo.doc.

26. JetBrains. *IntelliJ IDEA.* [En línea] 200-2013. [Citado el: 15 de Noviembre de 2012.] http://www.jetbrains.com/idea/.

27. **Ivar Jacobson, Grady Booch,James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Educación,S.A, 2000. pág. 464. ISBN:84-7829-036-2.

28. **Antonio Aliaga Ibarra, Marcos Agustin Miani Flores.** I.E.S.San Vicente. *PostgreSQL.* [En línea] 21 de Enero de 2008. [Citado el: 10 de Diciembre de 2012.] https://iessanvicente.com/colaboraciones/postgreSQL.pdf.

29. **Botta, Adrián.** Scribd. *Resumen de Tería de:Diseño de Sistemas.* [En línea] 2010. [Citado el: 20 de Enero de 2012.] http://es.scribd.com/doc/85235594/9/FLUJO-DE-TRABAJO-DE-CAPTURA-DE-REQUISITOS.

30. **Marcello Visconti, Hernán Astudillo.** *Fundamentos de Ingeniería de Software.* Departamento de Informática, Universidad Técnica Federico Santa María. Documento.

31. **Benneth Christiansson (Ed.), Mattias Forss,Ivar Hagen,Kent Hansson,Johan Jonasson,Mattias Jonasson,Fredrik Lott,Sara Olsson,Thomas Rosevall.** *GoF Design Patterns - with examples using Java and UML2.* 2008.

32. **Gamma, Erich, y otros, y otros.** *Design Patterns: Elements of Reusable Object-Oriented Software.* s.l. : Addison-Wesley, 1995.

33. **Guillerón, Gastón.** GLN Blog ingeniería & consultoría. [En línea] 12 de Julio de 2009. [Citado el: 11 de Marzo de 2013.] http://glnconsultora.com/blog/?p=3.

34. **Microsoft.** Conceptos básicos del diseño de una base de datos. [En línea] [Citado el: 12 de Marzo de 2013.] http://office.microsoft.com/es-es/access-help/conceptos-basicos-del-diseno-de-una-base-de-datos-HA001224247.aspx#BMgood.

35. **Quisbert Limachi Nancy Susana, Marca Huallpara Hugo Michael.** *ANALISIS Y DISEÑO DE SISTEMAS II. "Diagrama de Despliegue".*

36. **Castro, Luis.** About.com . *Guía básica sobre qué es HTTP y HTTPS.* [En línea] [Citado el: 1 de Abril de 2013.] http://aprenderinternet.about.com/od/ConceptosBasico/a/Que-Es-Http.htm.

37. **Martín, Eloi de San.** ProgramacionWeb.net . *El protocolo HTTPS.* [En línea] 23 de Junio de 2006. [Citado el: 1 de Abril de 2013.] http://www.programacionweb.net/articulos/articulo/?num=411.

38. **USB Implementers Forum, Inc.** Universal Serial Bus. [En línea] . [Citado el: 2 de Abril de 2013.] http://www.usb.org/home.

39. Entorno Virtual de Aprendizaje. *IS2R Material_de_caja_b_y_caja_n.* [En línea] [Citado el: 30 de Abril de 2013.] http://eva.uci.cu/file.php/158/Documentos/Recursos_bibliograficos/Libros_y_articulos_UD_2/Comun/Material_de_caja_b_y_caja_n.pdf.

40. Entorno Virtual de Aprendizaje. *IS2R. camino_básico.* [En línea] [Citado el: 30 de Abril de 2013.]

http://eva.uci.cu/file.php/158/Documentos/Recursos_didacticos/Presentaciones_digitales_UD_2/camino_basico.pdf.

41. MOZILLA DEVELOPER NETWORK. *JavaScript.* [En línea] 2005-2013. [Citado el: 10 de 11 de 2012.] https://developer.mozilla.org/en-US/docs/JavaScript.

42. Zentyal. *Máquinas Virtuales.* [En línea] 2004-2012. [Citado el: 14 de Noviembre de 2012.] http://doc.zentyal.org/es/virt.html.

43. W3C. *Hojas de Estilo.* [En línea] [Citado el: 9 de Noviembre de 2012.] http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo.

44. W3C. *CSS.* [En línea] [Citado el: 8 de Noviembre de 2012.] http://www.w3.org/Style/CSS/.

45. **Martínez, Rafael.** PostgreSQL-es. *Sobre PostgreSQL.* [En línea] 2 de Octubre de 2010. [Citado el: 20 de Noviembre de 2012.] http://www.postgresql.org.es/sobre_postgresql.

46. MASTERMAGAZINE. *Definición de Usuario.* [En línea] 2007-2012. [Citado el: 5 de Noviembre de 2012.] http://www.mastermagazine.info/termino/7056.php.

47. HTML.net. [En línea] [Citado el: 7 de Diciembre de 2012.] http://es.html.net/.

48. Grails. [En línea] 2009-2012. [Citado el: 20 de Noviembre de 2012.] www.grails.org.

49. **Sola, María José Aldaz.** Archivistica.net. *SOFTWARE LIBRE PARA SISTEMAS DE INFORMACIÓN.* [En línea] [Citado el: 2 de Diciembre de 2012.] http://www.archivistica.net/softwareopen.htm.

50. **Ibáñez, Blasco.** Universitat de Valéncia. *Guía para escribir documentos HTML*. [En línea] Septiembre de 2005. [Citado el: 6 de Noviembre de 2012.] http://www.uv.es/jac/guia/.

51. **Mtra. María de Lourdes Santiago Zaragoza, Lic. Mónica Flores López.** Universidad Tecnológica del Valle del Mezquital. *Desarrollando aplicaciones informáticas con el Proceso de Desarrollo Unificado (RUP).* [En línea] [Citado el: 19 de Noviembre de 2012.] http://www.utvm.edu.mx/OrganoInformativo/orgJul07/RUP.htm.

52. Ingeniería de Software ll. *EVA.* [En línea] [Citado el: 1 de Diciembre de 2012.] http://eva.uci.cu/file.php/158/Documentos/Bibliografia_general/Textos_Complementarios/El_Lenguaje_Unificado_de_Modelado/06_Parte_2_Conceptos_de_UML.pdf.

53. **Larman, Craig.** *UML y Patrones.Introducción al ánalisis y diseño orientado a objetos.* México : Prentice Hall, 1999. ISBN:970-17-0261-1.

54. software, selección. *Nuxeo DM.* [En línea] 2009. [Citado el: 16 de Diciembre de 2012.] http://www.softwareseleccion.com/nuxeo+dm-p-2238.

55. **Baratz.** Alfresco. [En línea] [Citado el: 15 de Diciembre de 2012.] http://www.alfresco.com/es/noticias/comunicados-de-prensa/alfresco-inicia-su-ciclo-taller-experto-el-23-de-septiembre-en-madrid.

56. **Miranda, Lic. Jorgelina Jiménez.** Revista Cubana de Informática Médica. *APROXIMACIÓN A LA PERSONALIZACIÓN DE LA PÁGINA PRINCIPAL DE GREENSTONE.* [En línea] 2002. [Citado el: 14 de Diciembre de 2012.]

57. **Serrano, Carlos J.Quintero.** JavaScript. [En línea] 16 de Mayo de 2011. [Citado el: 14 de Diciembre de 2012.] http://carlosquinterojavascript.blogspot.com/2011/05/ventajas-y-desventajas.html.

58. **MundoArchivistico.com.** mundoarchivistico. *Archivología.* [En línea] 2008. [Citado el: 28 de Noviembre de 2012.] http://mundoarchivistico.com.ar/?menu=diccionario&accion=ver&id=165.

59. **Walter.** Ingeniería Informática-UNPRG. *Definición de Administración.* [En línea] 2008. [Citado el: 29 de Diciembre de 2012.] http://inginformatica-unprg.blogspot.com/2008/07/definicion-de-administracion.html.

60. **Sun Microsystems, Inc.** Java BluePrints: Model-View-Controller. [En línea] http://java.sun.com/blueprints/patterns/MVC-detailed.html.

61. **Walls, Craig y Breidenbach, Ryan.** *Spring in Action.* s.l. : Manning Publications Co., 2005. 1-932394-35-4.

62. **Fowler, Martin.** Inversion of Control Containers and the Dependency Injection pattern. [En línea] Enero de 2004. http://martinfowler.com/articles/injection.html.

63. **Sun Microsystems, Inc.** Simplified Guide to the Java™ 2 Platform, Enterprise Edition. *Sun Developer Network (SDN).* [En línea] Septiembre de 1999. http://java.sun.com/j2ee/reference/whitepapers/j2ee_guide.pdf.

64. —. Java 2 Platform, Enterprise Edition (J2EE) Overview. *Sun Developer Network (SDN).* [En línea] http://java.sun.com/j2ee/overview.html.

65. **Fowler, Martin, y otros, y otros.** *Patterns of Enterprise Application Architecture.* 2002. ISBN : 0-321-12742-0 .

66. **García, Mauricio.** *Patrones de Diseño.* [En línea] 24 de Mayo de 2009. [Citado el: 11 de Marzo de 2013.] http://modelosprogramacion.blogspot.com/2009/05/nombre-facade.html.

67. **Erick Salazar, Anaís Aponte.** Sistemas de Programas (CI-3711). *Tópico: Patrones de Diseño Grasp.* [En línea] 25 de Marzo de 1999. [Citado el: 11 de Marzo de 2013.]

68. **Torres, Gisela.** Geeks.ms. *Inyección de dependecias e Inversión de control.* [En línea] 7 de Febrero de 2010. [Citado el: 10 de Marzo de 2013.] http://geeks.ms/blogs/gtorres/archive/2010/02/07/inyeccion-de-dependecias-e-inversion-de-control-con-structuremap.aspx.

69. **Tedeschi, Nicolás.** MSDN. *¿Qué es un Patrón de Diseño? .* [En línea] [Citado el: 10 de Marzo de 2013.] http://msdn.microsoft.com/es-es/library/bb972240.aspx#XSLTsection125121120120.

70. **Ing. Violena Hernández Aguilar, Ing. Michael González Jorrín.** Entorno Virtual de Aprendizaje. *Proceso de pruebas de caja negra basado en la descripción de casos de uso.* [En línea] [Citado el: 30 de Abril de 2013.] http://eva.uci.cu/file.php/158/Documentos/Recursos_didacticos/Presentaciones_digitales_UD_2/Proceso_de_pruebas_de_caja_negra_basado_en_la_descripcion_de_casos_de_uso.pdf.

71. **Acuña, Mirían Balestrini.** *Como se elabora el proyecto de investigación: (Para Estudios Formulativos o Exploratorios, Descriptivos, Diagnósticos, Evaluativos, Formulación de Hipótesis Causales, Experimentales y los Proyectos Factibles).* [ed.] 2001 Bl. 5. Caracas, Venezuela : Servicio Editorial, 2002. pág. 248.

72. **Sommerville, Ian.** *Ingeniería del software 7/e.* s.l. : Pearson Education.

Anexos

3.8 Anexo I Documento Entrevista con el cliente.

3.9 Anexo II. Documento Especificación de los Casos de Uso.

3.10 Anexo III. Documento Modelo de Análisis.

3.11 Anexo IV Documento Modelo de Diseño.

3.12 Anexo V Documento de Pruebas de Caja Negra.

3.13 Anexo VI Documento LCH_Usabilidad_Arkheia.