UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

MÓDULO INCORPORACIÓN DE DOCUMENTOS DIGITALES DE XABAL REPXOS 3.0 EN LA TECNOLOGÍA DEL SHARE DE ALFRESCO 5.2

Autor: Ibrahim Caballero Pacheco

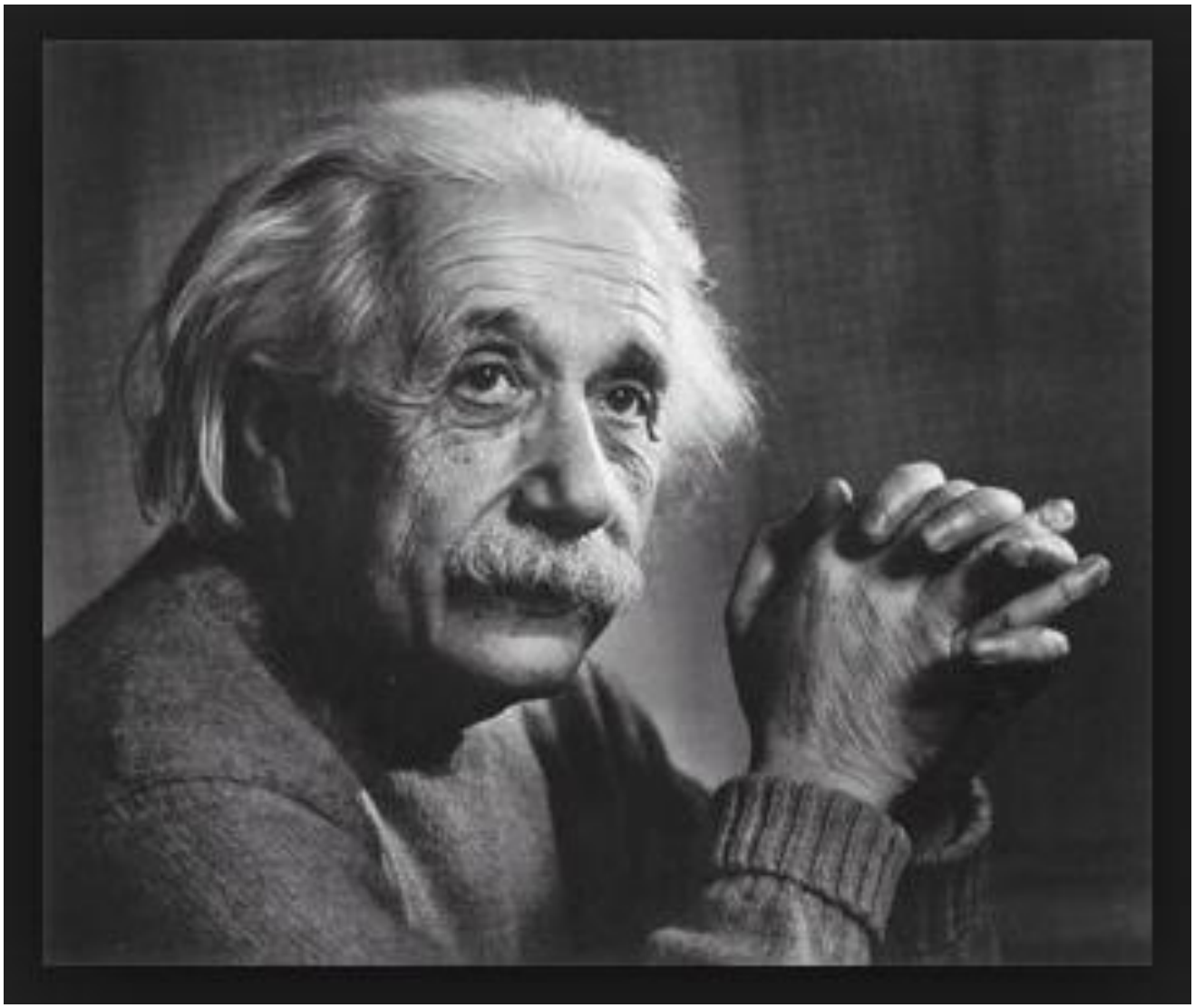
Tutores: MSc. Madelis Pérez Gil

Ing. José Javier Hernández Benítez

Ing. Dariel Núñez Acosta

"Año 61 de la Revolución"

La Habana, junio 2019



"Hay una fuerza motriz más poderosa que el vapor, la electricidad, y la energía atómica, la voluntad"

Albert Einstein

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis que tiene por título: Módulo Incorporación de Documentos Digitales de Xabal Repxos 3.0 en la Tecnología del Share de Alfresco 5.2 y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Ibrahim Caballero Pacheco

_____

Firma del Autor

MSc. Madelis Pérez Gil	Ing. Dariel Núñez Acosta	Ing. José Javier Hernández
_____	_____	_____
Firma del Tutor	Firma del Tutor	Firma del Tutor

Dedicatoria

Dedico la misma a:

A mis padres y hermanos, a mis abuelos, a mi prima Karina, a mi novia, a mis amigos, que siempre esperaron lo mejor de mí, y deseaban tanto como yo que llegara este grandioso día.

Agradecimientos

A mis padres y hermanos:

Por apoyarme siempre, por confiar en mi, por ser ejemplos a seguir, y por ayudarme a alcanzar esta meta.

A mis tutores:

Por ser más que mis tutores, mi familia, por las noches largas dedicadas, por los regaños, por sus exigencias y por ayudarme a formarme como un profesional.

A mi novia:

Por estar a mi lado siempre, por comprenderme y por ser una de mis inspiraciones.

A mi prima Karina:

Por ser como mi madre en estos duros 5 años de mi carrera.

A mis amigos:

Por ser durante todo este tiempo como mis hermanos, y por estar en las buenas y en las malas.

Resumen

El Centro de Informatización de la Gestión Documental, se especializa en el desarrollo de sistemas y servicios informáticos integrales de alta calidad y competitividad en la informatización de los procesos de gestión documental. En dicho centro se encuentra el producto Repxos el cual tiene implementado el Módulo Depósito de Documentos. Debido a la necesidad de creación de una suit de Gestión Documental para CIGED, se hace necesario el desarrollo de este módulo con las tecnologías que sean compatibles con otros productos desarrollados en el centro como es el caso de Xabal eXcriba. Se propone como objetivo general desarrollar el Módulo Incorporación de Documentos Digitales de Xabal Repxos 3.0 utilizando Alfresco Community 5.2. Para su desarrollo fue necesario un estudio del Módulo Depósito de Documentos de Xabal Repxos 3.0, las tecnologías, lenguajes, características y funciones del mismo. Durante el desarrollo del módulo se utilizaron como lenguajes de desarrollo JavaScript 1.6, XML 1.0, UML 2.1 y HTML5, para diagramar el Visual Paradigm 8.0, y los editores de textos Sublime Text 3.0 y Visual Studio Code. Fue definida una estrategia de pruebas que permitió corroborar el éxito del desarrollo de la solución, en el escenario de validación.

Palabras clave: gestión documental, share, suit.

Abstract

The Center for Computerization of Documentary Management (CIGED), is an organization specialized in the development of high quality computer systems and services of high quality and competitiveness in the computerization of document management processes. In this center is the Repxos system which has implemented the Document Deposit Module, due to the need to create a document management suite for CIGED, it is necessary to develop this module with the technologies that are compatible with other products. developed in the center as is the case of Excriba. The research was carried out in an approximate time of 6 months and proposes as a general objective to develop the Module of Incorporation of Digital Documents of Repxos using the technology of the Share of Alfresco 5.2. For its development it was necessary a study of the Deposit Module of Documents of Repxos, the technologies, languages, characteristics and functions of the same. During the development of the module, JavaScript 1.6, XML 1.0, UML 2.1 and HTML5 were used as development languages, as were Visual Paradigm 8.0, and Sublime Text 3.0 and Visual Code Studio text editors. A test strategy was defined to corroborate the success of the development of the solution, in the validation scenario.

Keywords: document management, share, suite

Índice

Índice de tablas

Índice de figuras

Introducción

Todas las empresas u organizaciones, en el transcurso de su actividad, generan documentos donde quedan reflejadas las acciones realizadas en sus procesos de trabajo. Muchas empresas no tienen conciencia o percepción de que entre sus activos, al igual que el patrimonio arquitectónico, sus infraestructuras o sus bienes de capital, están también los documentos donde se registra la información de las actividades de negocio. Muy frecuentemente, solo cuando desaparecen esos documentos por cualquier catástrofe advierten del valor de los documentos, a veces vitales para que la empresa pueda continuar. Pocas empresas hacen un análisis de riesgos y evalúan qué podría ocurrir si perdiesen los documentos, en el formato o formatos en los que se hayan generado (Giménez Chornet, 2015).

El desarrollo de la Sociedad y de las Tecnologías de la Información y la Comunicación, ha generado en el tejido empresarial la necesidad de contar con sistemas que permitan una gestión eficaz de los recursos de información y documentación de las empresas. Este hecho ha favorecido la reciente aparición y fuerte crecimiento de un sector de empresas, denominadas servicios documentales, que ofrecen productos y servicios relacionados con la gestión de información, documentación y conocimiento (E. V. López, 2017).

La gestión documental surge como área de gestión responsable de un control eficaz y sistemático de la creación, la recepción, el mantenimiento, el uso y la disposición de documentos, incluidos los procesos para incorporar y mantener en forma de documentos la información y prueba de las actividades y operaciones de la organización (Martínez & de Mingo, 2018).

La nueva realidad introducida por la incorporación de las TIC a las organizaciones y el documento electrónico, engendró una mayor accesibilidad a dicha documentación. De esta forma se inicia un sensible crecimiento en el número de documentos, añádase los más variados formatos y en múltiples soportes, lo que tributa a un incremento del número de usuarios que utilizan dichos servicios, lo cual propicia, más allá de la accesibilidad, una protección de los documentos más vulnerables (Diez, Domínguez, Martínez, & Sáenz, 2012).

Con el gradual desarrollo de las TIC la sociedad se ha visto envuelta en un proceso evolutivo que ha beneficiado a varias de sus esferas, trayendo consigo facilidades en la gestión de la documentación a nivel empresarial. Evidenciándose en la creación de sistemas electrónicos capaces de gestionar de forma automatizada el control documental.

En la Universidad de las Ciencias Informáticas (UCI) se encuentra el Centro de Informatización de la Gestión Documental (CIGED). El mismo está dedicado al desarrollo de sistemas y servicios informáticos

integrales de alta calidad y competitividad en la informatización de los procesos de gestión documental. Cuenta con el producto Gestor de Documentos Administrativos Xabal eXcriba, este es un software diseñado para la gestión documental, que tramita los documentos administrativos que se generan o reciben dentro de las organizaciones a partir de sus funciones, por lo tanto, involucra todas las áreas de una organización, permitiendo gestionar de forma correcta la documentación como prueba, testimonio y evidencia de las actividades organizacionales.

Otros de los productos existentes en el centro es el sistema Xabal Repxos 3.0, desarrollado por el proyecto Repositorio Institucional donde existe el Módulo Depósito de Documentos Digitales. Este módulo permite describir un documento utilizando el estándar de metadatos Dublin Core y posteriormente realizar su envío al repositorio. Independiente de esto, el envío puede contener o no un flujo de revisión y no se hace visible en el repositorio dicho documento hasta que este flujo no sea completado.

Actualmente en el centro CIGED se están llevando a cabo la evolución de otros softwares en tecnologías diferentes y como meta del centro, la UCI y el país, se quiere la integración de los mismos en una suit de gestión documental. Debido a que Xabal Repxos presenta dificultades a la hora de integrarse con otras tecnologías se hace de vital importancia la migración del Módulo Depósito de Documentos de Xabal Repxos 3.0, hacia otra tecnología, escogiéndose para ello Alfresco Community en su versión 5.2. Partiendo de esta situación se obtiene como **problema de investigación**:

¿Cómo lograr que el Módulo Depósito de Documentos de Xabal Repxos 3?0 se integre con Alfresco Community 5.2? Se tiene como **objeto de estudio**: La incorporación de documentos digitales en los repositorios. Se plantea como **objetivo general**: Desarrollar el Módulo Incorporación de Documentos Digitales de Repxos utilizando la tecnología del Share de Alfresco 5.2 teniendo como **campo de acción** La incorporación de documentos digitales en Xabal Repxos 3.0, definiendo las siguientes **preguntas científicas:**

1. ¿Cuáles son los referentes teóricos a tener en cuenta para abordar la solución del problema planteado relacionado con el Módulo Depósito de Documentos Digitales de Xabal REPXOS 3?0?

2. ¿Qué propuesta de solución se define para migrar el Módulo Depósito de Documentos Digitales de Xabal Repxos 3?0 a la tecnología del Share de Alfresco 5.2?

3. ¿Cómo desarrollar un Módulo Incorporación de Documentos Digitales en la tecnología del Share de Alfresco 5?2 que permita integrar las tecnologías del centro?

4. ¿Cómo se valida el correcto funcionamiento del Módulo Incorporación de Documentos Digitales de

Xabal Repxos 3?0 en la tecnología del Share de Alfresco 5.2?

Se proponen las siguientes **tareas de investigación**:

1. Revisar la bibliografía para elaborar el marco teórico conceptual en lo referente al desarrollo del Módulo Incorporación de Documentos Digitales de Xabal Repxos 3.0 en la tecnología del Share de Alfresco 5.2.
2. Analizar las herramientas, tecnologías y lenguajes informáticos, además de la metodología de desarrollo de software asumidas para realizar la implementación del Módulo Incorporación de Documentos Digitales en Alfresco Community 5.2.
3. Identificar las principales funcionalidades del sistema para la posterior implementación del mismo.
4. Identificar los diferentes tipos de pruebas de software, y aplicarlas sobre el módulo desarrollado.

Para el desarrollo de la investigación se hizo uso de métodos científicos de investigación siendo estos la forma de abordar la realidad, de estudiar la naturaleza, la sociedad y el pensamiento, con el propósito de descubrir su esencia y sus relaciones. Se clasifican en teóricos y empíricos, los cuales están dialécticamente relacionados(Rolando Alfredo, 2012). En la investigación se utilizaron los siguientes métodos teóricos:

Analítico-Sintético: Se utilizó con el objetivo de analizar los documentos, teorías y otros materiales que permitieron la comprensión de los elementos relacionados con la incorporación de documentos.

Histórico-lógico: Se empleó en el estudio de cómo ha evolucionado el tema que se está investigando, de las diferentes herramientas existentes, sus características, ventajas y desventajas.

Modelado: Se utiliza con la intención de ayudar a comprender las teorías relacionadas con la incorporación de documentos de forma simplificada mediante modelos teóricos. También se ha utilizado para modelar todos los diagramas correspondientes a la etapa de análisis, diseño e implementación del módulo a desarrollar en la presente investigación.

También se utiliza para la investigación los siguientes métodos empíricos:

Observación: Se utilizó para el diagnóstico del problema que se investiga, a fin de obtener los elementos necesarios para desarrollar el Módulo Incorporación de Documentos Digitales de Repxos en la tecnología del Share de Alfresco 5.2.

Documental: Se utilizó para consultar bibliografía en fuentes de carácter documental tales como libros, artículos y ensayos.

Para lograr una mejor organización y lectura el trabajo de diploma se estructuró de la siguiente manera: Resumen, Introducción, Tres Capítulos, Conclusiones, Recomendaciones, Referencias Bibliográficas, Bibliografía y Anexos, donde se abarca todo lo relacionado con la investigación realizada.

Capítulo 1: Fundamentación teórica: Se establecen los principales aspectos teóricos para el objeto de estudio de la investigación. Se realiza el estudio de los módulos de incorporación existentes. Estudio de los sistemas como Alfresco 5.2. Se seleccionan herramientas y tecnologías que se emplean en el desarrollo del módulo incorporación para Alfresco Share, y se caracteriza la metodología para seleccionar aquella que se adecue mejor a las características del producto.

Capítulo 2: Análisis y diseño de la propuesta de solución: Se describe la propuesta de solución. Se diseña el modelo de domino. Se definen los requisitos funcionales y no funcionales. Se diseñan y elaboran diagramas y especificaciones de casos de uso. Se explican y ejemplifican los patrones arquitectónicos y de diseños utilizados.

Capítulo 3: Implementación y pruebas: Se describe el proceso de implementación del módulo, además se elabora el diagrama de despliegue y se define la estrategia de pruebas de software.

Capítulo 1 Fundamentación teórica

En el presente capítulo se abordan los conceptos fundamentales y sistemas existentes relacionados con la incorporación de documentos digitales en el ámbito nacional e internacional, para ello se tienen en cuenta características que permiten comprender su funcionamiento y cuánto aportan a la investigación en curso. Se hace referencia a las tendencias y tecnologías actuales sobre metodologías de desarrollo y herramientas para el modelado de datos. Además, se realiza un estudio de algunas tecnologías libres de desarrollo, lenguajes de programación y sistemas gestores de base de datos, con el objetivo de seleccionar los más indicados para el desarrollo de una solución de software con las características deseadas.

1.1 Conceptos fundamentales

Metadatos

Los metadatos son, en efecto, datos más o menos estructurados sobre datos, los cuales no se colocan en una ficha como los datos catalográficos, sino que se almacenan en una base de datos con una referencia al texto completo o se incluyen mediante metaetiquetas HTML, o los llamados encabezamientos SGML en la fuente de los documentos, o en algún fichero relacionado, de manera que puedan recuperarse por un buscador; un proceso transparente para el usuario (Vargas-Arcila, Baldassarri, & Arciniegas, 2016).

Los metadatos tienen como objetivo describir un documento en una colección, proveer de múltiples vías para obtenerlo y poner su contenido en contexto con otra información, tanto en el entorno bibliotecológico, donde se almacenan en catálogos y describen documentos físicos para ayudar a su búsqueda, como en el contexto digital (Morgado, Ortuño, & Fernández, 2018).

Una particularidad en este entorno es que los metadatos buscan facilitar el funcionamiento y la capacidad de compartir los datos entre los buscadores, a elevar su precisión, para hacer más efectivo el proceso de búsqueda y recuperación en el Web y, de esta manera, permitir a los usuarios, conocer los datos sobre los recursos que necesitan.

Documento

Un documento es cualquier soporte material en el que se incorpora algún tipo de información. El lenguaje escrito , la música o la matemática necesitan de medios materiales a traves de los cuales se transmite algo, lo cual se realiza a través de una modalidad de documento (una notificación, una estadística, una grabación o una redacción (Torres, Morales, Hernández, Matos, & Palomino, 2016)

En una empresa es vital el resguardo y cuidado de los documentos que la constituyen, ya que es importante mantener un registro de todo lo que se procesa y calcula en ella por medio de estos documentos. Un documento debe seguir una serie de reglas y condiciones que deben ser respetadas por el que lo redacta, entre ellas destacan el uso correcto de una ortografía aceptable, además debe estar correctamente identificado para saber de qué lugar viene y que destino lleva. Los documentos pueden ser manuscritos (escritos a mano), pero en la actualidad con el creciente auge de la tecnología, es común hacer los documentos en computadora, parar esto existen completos y sencillos editores que hacen la experiencia de redactar un documento más fácil (Gallo, 2011).

Gestión documental

La gestión documental es un proceso administrativo que permite analizar y controlar sistemáticamente, a lo largo de su ciclo de vida, la información registrada que crea, recibe, mantiene o utiliza la organización en correspondencia con su misión, objetivos y operaciones. También la considera como un proceso para mantener la información en un formato que permita su acceso oportuno, y por ello se requiere de la realización de tareas y procedimientos particulares para cada fase de su ciclo de vida y su explotación. La información registrada es evidencia de las actividades y transacciones de las organizaciones, y su uso oportuno permite a la organización una mayor eficacia en su labor (de Dios Arias, Cano Inclán, García García, & Raposo Villavicencio, 2015).

La gestión documental es un proceso vital para la organización, debido a la magnitud que alcanzan los documentos, como resultado del (Ruiz & Piere, 2017):

- Amplio alcance y complejidad de las actividades gubernamentales y empresariales.
- Incremento del sector de servicios donde se realizan múltiples actividades relativas al manejo y transferencia de datos e informaciones.
- Aumento de las regulaciones y las normativas oficiales que reglamentan el empleo de los documentos.
- Incorporación y extensión de medios técnicos (fotocopiadoras, impresoras, otros.) que generan grandes volúmenes de información registrada en diversos formatos.

Sistemas de gestión documental

Un sistema de gestión documental, o *document management system* (DMS), por sus siglas en inglés, está diseñado para almacenar, administrar y controlar el flujo de documentos dentro de una organización. Se trata de una forma de organizar los documentos e imágenes digitales en una localización centralizada a la

que los empleados puedan acceder de forma fácil y sencilla (del Prado Martínez & Esteban Navarro, 2016).

Ventajas de un sistema de gestión documental:

- Digitalización de documentos: Comenzar a trabajar con un sistema de gestión documental significa llevar a cabo la digitalización de documentos en papel. Con la colaboración de un escáner, los documentos físicos se convertirán en versiones digitales que se almacenarán en una localización central. Este procedimiento puede ser tedioso y bastante costoso, sin embargo, una digitalización organizada y planificada es primordial para el correcto aprovechamiento del sistema.

- Localización central: La cantidad de canales a través de los que la información llega a las empresas es amplia. A consecuencia de esto, grandes volúmenes de información quedan desestructurados y repartidos entre los distintos sistemas de una compañía. Un sistema de gestión documental almacena y organiza toda la información proveniente del trabajo diario de la empresa en una ubicación central. La empresa decidira que empleados pueden tener acceso a los documentos alojados en dicha localización gracias al mecanismo de concesión de permisos. Esta centralización de la información supone terminar con la búsqueda infinita de documentos por las redes de carpetas de la organización agilizando, de esta forma, el ritmo de trabajo.

- Mejorar el flujo de trabajo: Un DMS puede convertir los flujos de trabajo en procesos más eficientes y productivos. Gracias a la automatización de las funciones, el sistema proporcionará una imagen global de los procesos de trabajo dentro de la compañía. Este control de procesos permitirá seguir las tareas incompletas, conocer aquellas que ya han finalizado o automatizar tareas repetitivas que terminarán ahorrando tiempo a la organización.

- Seguridad de la información: Aún son muchas las empresas que mantienen sus documentos almacenados en ficheros físicos y, de este modo, la posibilidad de que un archivo termine perdido o deteriorado es alta. Los DMS solucionan este problema. Estableciendo copias digitales de documentos en papel, el riesgo de pérdida disminuye de forma considerable. De esta forma, la organización trabaja con la certeza de que toda su información está segura ubicada en un mismo sistema, sistema que puede incluir entre otras características, la recuperación de datos en caso de desastre.

- Compartir documentos: Muchos documentos son creados para ser compartidos. Los sistemas de gestión documental facilitan esta tarea. A través de la creación de grupos o con accesos ilimitados a la localización central, los documentos pueden ser distribuidos tanto de forma interna como

externa. Los gestores documentales permiten a diferentes grupos externos a la empresa (proveedores, clientes…) el acceso a documentos necesarios para la relación que mantienen. Con esta característica ya no son necesarias las pequeñas memorias USB, o los emails con diferentes versiones de un documento.

Repositorio digital

Entendemos a un repositorio como una plataforma digital "que recoge, preserva y difunde la producción académica de una institución y/o de una disciplina científica, permitiendo el acceso a los objetos digitales que contiene y a sus metadatos". Pueden contener distintos tipos de documentos y/o información (F.-A. López, 2013).

Deben tener cuatro características fundamentales:

- Auto-archivo
- Interoperabilidad
- Acceso libre y gratuito
- Preservación a largo plazo

Xabal Repxos 3.0

Repxos es un sistema web que permite almacenar y difundir la producción intelectual digital generada por una institución. Basado en la herramienta Dspace, tiene como objetivos principales garantizar la visibilidad de los autores, facilitar el contacto entre ellos, favorecer la discusión de los trabajos depositados, contribuir al aumento de las citaciones y al impacto de los trabajos en la comunidad científica de la institución donde se instale.

Toda la información generada en los distintos tipos de eventos que se desarrollan en la institución, es almacenada en XABAL REPXOS 3.0; su integración con los procesos investigativos, permite utilizar la documentación como fuente de información, ofreciendo una vía para la divulgación del conocimiento y también una herramienta en el proceso docente-educativo. Se facilita de esta forma el acceso abierto a la información a los investigadores de la institución ("UCI," 2019).

Xabal eXcriba 3.1

Sistema de gestión documental orientado a la automatización de los procesos documentales de las organizaciones. Contiene todos los instrumentos para cubrir desde la generación de documentos, su revisión, administración, distribución, custodia y disposición, cumpliendo con las normas internacionales para la gestión de documentos, manteniendo la seguridad de los documentos en formato electrónico.

Entre sus principales funcionalidades se destacan el control de versiones, accesos y permisos, automatización de los flujos documentales, gestión de documentos y carpetas, gestión de notas sobre los documentos y listas de datos. Contiene módulos para la gestión de acuerdos, gestión de reportes, notificaciones, gestión de registros de entrada y salida de documentos y gestión del cuadro de clasificación de la entidad ("UCI," 2019).

1.2 Sistemas para repositorios

DSpace

DSpace es un sistema que provee una forma de gestionar materiales y publicaciones resultantes, tanto de la actividad de investigación como de educación, almacenadas en un repositorio que ofrece a los usuarios accesibilidad en todo momento. Creado en colaboración entre la empresa *Hewlett-Packard* y las bibliotecas del *Massachusetts Institute of Technology*, el programa en sus inicios satisfizo las necesidades de estos centros de información pero luego se colocó a disposición de la comunidad internacional, como una herramienta *open spurce*, gratuita y de licencia GPL(Gairin, 2018).

Se instala sobre sistema operativo Unix y Windows, que además necesita de la aplicación Java para su funcionamiento. Esta herramienta también incluye un sistema de bases de datos relacionales; este repositorio de colecciones digitales puede montarse sobre servidores Web PostgreSQL. Apache o Tomcat.

DSpace utiliza el estándar de metadatos Dublin Core para la descripción de los documentos, que van desde texto, hasta imágenes y videos, entre otros, y que luego posibilita su recuperación precisa. La interfaz en ambiente Web varía en dependencia de la persona que la utilice; así, los responsables de la colección tienen una, los administradores del sistema otra y los usuarios finales otra; aunque todas coinciden en que son en idioma inglés. Los usuarios pueden hacer sus búsquedas a partir de los metadatos declarados por los especialistas o simplemente mediante las listas de documentos por campos específicos, como son: autor, título y fecha (Ramirez, 2016).

Fedora

Fedora es el acrónimo de *Flexible Extensible Digital Object Repository Architecture* (Arquitectura digital de repositorio de objetos digitales flexible y extensible). El repositorio de documentos digitales Fedora requiere de la instalación previa del programa Java en el servidor en el que funcionará; además, Fedora incluye en su paquete de instalación una base de datos que en caso de que los especialistas decidan no utilizarla, puede sustituirse por otras como Oracle y MySQL. Este software funciona con los sistemas operativos Windows y Unix y sobre servidores Web Apache y Tomcat (Doria, Prado, & Haustein, 2015)

Fedora, al ser un sistema *open source* gratuito, ofrece a los programadores su código fuente. Se distribuye bajo la Licencia de la Comunidad Educativa, que permite que el programa se propague a todos los que lo requieran y que se hagan modificaciones, siempre que se coloquen en forma visible los términos de esta licencia para que otras personas puedan conocer sobre ella.

El procesamiento de los documentos se realiza según los metadatos asignados por los especialistas en formato Dublin Core. La interfaz de presentación de Fedora es distinta para cada una de sus sesiones, es decir, una para el procesamiento de los documentos y otra para los usuarios finales, que además tendrán la posibilidad de recuperar los contenidos mediante búsquedas en varios índices, previamente declarados por los procesadores o mediante la navegación por las listas de las colecciones. Fedora permite crear colecciones digitales en varios formatos de documentos, como son: texto, imagen, sonido, y otros. El lenguaje del programa, tanto de la interfaz de trabajo como de presentación a los usuarios, es en inglés, aunque es posible configurar, por medio de la agregación de aplicaciones adicionales que ofrece el sistema, el programa en varios idiomas (Sarduy Domínguez & Urra González, 2006).

CONTENTdm

CONTENTdm provee herramientas para la organización, gestión, publicación y recuperación de colecciones digitales de todo tipo de documentos, desde texto (en varios formatos) hasta imágenes, videos y audio. Este programa se utiliza desde servicios Web; así los usuarios pueden ver los resultados desde sus navegadores Web, sin necesidad de instalar herramientas adicionales (Merlino-Santesteban, 2012).

Los servidores en los que se instala CONTENTdm requieren de Windows Server, Linux o Solaris, además de servidores Web dedicados, montados en IIS con Windows o Apache. Las estaciones de trabajo de las personas que se ocupan de desarrollar las colecciones necesitan instalar Windows 2000 o superior para poder trabajar con el programa.

CONTENTdm es una herramienta paga, con licencia privada, que permite a los usuarios desarrollar sus colecciones a partir de los patrones que ellos definen, pero no pueden hacer ningún tipo de cambio o adecuación si lo necesitaran (Sarduy Domínguez & Urra González, 2006).

1.3 Módulo Depósito de Documento de Repxos 3.0

Fue implementado en año 2015 por los autores Adrián Porras Cabreras y Landy Torres Cregor. Este módulo permite realizar de forma correcta la configuración y validación de los metadatos asociados a los documentos que se envíen hacia Xabal Repxos 3.0. Para ello se muestran los metadatos específicos según el tipo de documento que se desea enviar. Se realiza la validación de los pasos necesarios para

lograr el correcto depósito de los documentos, mostrando en todo momento mensajes sugerentes de acuerdo a los posibles errores que pudiera cometer un usuario durante el envío. Previo al envío de documentos digitales se debe crear un formulario, llenando los campos necesarios según el tipo de documento definido. Este formulario puede ser guardado con los elementos básicos y posteriormente retomarlo para realizar el envío. Además, podrá ser actualizado o eliminado por el usuario en el momento que desee. El módulo está desarrollado independiente del sistema Xabal REPXOS 3.0 y el envío se realiza a través del servicio web REST. Esta forma de envío evitaría que, en caso de que se libere una nueva versión del DSpace, no se tengan que realizar cambios en el código fuente.

1.4 Metodología para desarrollo de software

En las dos últimas décadas las notaciones de modelado y posteriormente las herramientas pretendieron ser las "balas de plata" para el éxito en el desarrollo de software, sin embargo, las expectativas no fueron satisfechas. Esto se debe en gran parte a que otro importante elemento, la metodología de desarrollo, había sido postergado. De nada sirven buenas notaciones y herramientas si no se proveen directivas para su aplicación. Así, esta década ha comenzado con un creciente interés en metodologías de desarrollo. Hasta hace poco el proceso de desarrollo llevaba asociada un marcado énfasis en el control del proceso mediante una rigurosa definición de roles, actividades y artefactos, incluyendo modelado y documentación detallada. Este esquema "tradicional" para abordar el desarrollo de software ha demostrado ser efectivo y necesario en proyectos de gran tamaño (respecto a tiempo y recursos), donde por lo general se exige un alto grado de ceremonia en el proceso. Sin embargo, este enfoque no resulta ser el más adecuado para muchos de los proyectos actuales donde el entorno del sistema es muy cambiante, y en donde se exige reducir drásticamente los tiempos de desarrollo, pero manteniendo una alta calidad. Ante las dificultades para utilizar metodologías tradicionales con estas restricciones de tiempo y flexibilidad, muchos equipos de desarrollo se resignan a prescindir del buen hacer de la ingeniería del software, asumiendo el riesgo que ello conlleva. En este escenario, las metodologías ágiles emergen como una posible respuesta para llenar ese vacío metodológico. Por estar especialmente orientadas para proyectos pequeños, las metodologías ágiles constituyen una solución a medida para ese entorno, aportando una elevada simplificación que a pesar de ello no renuncia a las prácticas esenciales para asegurar la calidad del producto (Canós & Letelier, 2012).

En un proyecto de desarrollo de software la metodología define quién debe hacer, qué, cuándo y cómo, de ahí la importancia que tiene realizar una correcta elección de la metodología a emplear. Hoy día no existe una metodología que sea enteramente universal por eso para su selección hay que tener en cuenta las

características de cada proyecto. AUP-UCI en el escenario número dos es la metodología seleccionada para desarrollar el módulo.

Proceso Unificado Ágil para el Desarrollo de la Actividad Productiva en la Universidad de las Ciencias Informáticas

Al no existir una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos) exigiéndose así que el proceso sea configurable. Por lo que se decide por parte de la dirección de producción hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI. Una metodología de desarrollo de software tiene entre sus objetivos aumentar la calidad del software que se produce, de ahí la importancia de aplicar buenas prácticas, para ello se tiene en cuenta el Modelo CMMI-DEV v1.3. El cual constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora. Estas prácticas se centran en el desarrollo de productos y servicios de calidad (UCI,2015).

Esta metodología de tipo ágil divide el ciclo de vida de la producción del software en tres fases:

Inicio: Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto, específicamente para el desarrollo del módulo esta fase se evidencia en el estudio del estado del arte.

Ejecución: En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto, en el desarrollo de este módulo la fase ejecución se pone de manifiesto en la creación de la propuesta de solución y la realización de las pruebas.

Cierre: En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto, esta fase no se ve envuelta durante del desarrollo del módulo.

AUP-UCI propone para el ciclo de vida de los proyectos en la UCI tener siete disciplinas, los flujos de trabajo son los siguientes: Modelado de negocio, Requisitos, Análisis y diseño, Implementación, Pruebas internas, Pruebas de liberación y Pruebas de aceptación. Además, está compuesta por cuatro escenarios:

1. El escenario número uno propone a los proyectos que modelan el negocio con casos de usos del negocio que solo pueden modelar el sistema con casos de usos del sistema.

2. El escenario número dos propone modelar el negocio con un modelo conceptual y el sistema con casos de uso del sistema.
3. El escenario número tres propone modelar el negocio con descripción de proceso de negocio, junto al modelo conceptual y el sistema mediante la descripción de requisitos por proceso.
4. El escenario número cuatro propone no modelar el negocio y describir el sistema mediante historias de usuario.

Se selecciona para esta investigación el escenario número dos porque este aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelan exclusivamente los conceptos fundamentales del negocio. Se recomienda este escenario para proyectos donde el objetivo primario es la gestión y presentación de información.

Con la adaptación de AUP para la actividad productiva de la UCI se logra estandarizar el proceso de desarrollo de software, dando cumplimiento además a las buenas prácticas que define CMMIDEV v1.3. Se logra hablar un lenguaje común en cuanto a fases, disciplinas, roles y productos de trabajos.

1.5 Herramientas, tecnologías y lenguajes informáticos

Visual Paradigm 8.0

Visual Paradigm es una herramienta para desarrollo de aplicaciones utilizando modelado UML, el cual es ideal para ingenieros de software, analistas de sistemas y arquitectos de sistemas que están interesados en el desarrollo de sistemas de gran tamaño y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos (Peña, 2016).

Fue definido utilizar Visual Paradigm 8.0 principalmente porque la UCI tiene licencia para su uso y entre las ventajas que ofrece actualmente cumple con las políticas de migración a software libre en Cuba, ya que es una herramienta multiplataforma que se puede utilizar tanto en Linux como en Windows.

Apache Ant 1.9.6

Apache Ant es una herramienta usada en programación para la realización de tareas mecánicas y repetitivas, normalmente durante la fase de Compilación y construcción. Es, por tanto, un software para procesos de automatización de compilación, desarrollado en lenguaje java y requiere la plataforma java por lo que resulta apropiado para la construcción de proyectos java.

Esta herramienta tiene la ventaja de no depender de las órdenes del shell de cada sistema operativo, sino que se basa en archivos de configuración XML y clases Java para la realización de las distintas tareas,

siendo idónea como solución multi-plataforma. Ant utiliza XML para describir el proceso de generación y sus dependencias. Por defecto, el archivo XML se denomina build.xml (MARTÍN LÓPEZ-ASÚNSOLO, 2017).

Postgres SQL 9.4

Los sistemas de mantenimiento de Bases de Datos relacionales tradicionales (DBMS,s) soportan un modelo de datos que consisten en una colección de relaciones con nombre, que contienen atributos de un tipo específico. En los sistemas comerciales actuales, los tipos posibles incluyen numéricos de punto flotante, enteros, cadenas de caractéres, cantidades monetarias y fechas. Está generalmente reconocido que este modelo será inadecuado para las aplicaciones futuras de procesado de datos. El modelo relacional sustituyó modelos previos en parte por su "simplicidad espartana". Sin embargo, como se ha mencionado, esta simplicidad también hace muy dificil la implementación de ciertas aplicaciones. Postgres ofrece una potencia adicional sustancial al incorporar los siguientes cuatro conceptos adicionales básicos en una vía en la que los usuarios pueden extender fácilmente el sistema, Clases, herencia, tipos y funciones (Ginestà & Mora, 2012).

Sublime Text 3.0

Sublime Text es un editor de texto y editor de código fuente que está escrito en C++ y Python para los plugins. Desarrollado originalmente como una extensión de Vim, el sistema de resaltado de sintaxis de Sublime Text soporta un gran número de lenguajes (C, C++, C#, CSS, D, Erlang, HTML, Groovy, Haskell, HTML, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, Matlab, OCaml, Perl, PHP, Python, R, Ruby, SQL, TCL, Textile y XML); soporta macros, Zippers y auto completar, entre otras funcionalidades. Algunas de sus características son ampliables mediante plugins (Rovitto & Nisim, 2018).

Dspace 6.0

DSpace es el software de elección para organizaciones académicas, sin fines de lucro y comerciales que crean repositorios digitales abiertos. Es gratuito y fácil de instalar "listo para usar" y completamente personalizable para adaptarse a las necesidades de cualquier organización.

DSpace conserva y permite un acceso fácil y abierto a todo tipo de contenido digital, incluido texto, imágenes, imágenes en movimiento, mpeg y conjuntos de datos. Y con una comunidad cada vez mayor de desarrolladores, comprometidos a expandir y mejorar continuamente el software, cada instalación de DSpace se beneficia de la siguiente.

¿Por qué seleccionar Dspace?

Los objetivos principales de DSpace son centralizar, normalizar, almacenar, diseminar y preservar la producción científica y académica de las instituciones. Su estructura permite organizar la información en comunidades que, a su vez, se segmentan en colecciones de documentos.

Dspace almacena casi cualquier tipo de formato y documento, así como la catalogación de los mismos utilizando el estándar Dublin Core. Es posible crear repositorios que integran contenidos de texto plano, documentos con formato, imágenes, bases de datos, programas ejecutables y contenido multimedia.

Al ser una plataforma de software libre, es muy factible la configuración de funcionalidades que respondan a las necesidades específicas de cada organización.

Algunas ventajas en la organización de la información para las instituciones, organizaciones y empresas que usan Dspace son (Gairin, 2018):

- Cada departamento podrá definir el flujo de trabajo y proceso de publicación de sus documentos de forma independiente.
- Cada departamento podrá formar las colecciones de documentos que requiera, según sus necesidades.
- Cada colección de documentos puede contar con un esquema de metadatos personalizado.
- Cada colección de documentos puede contar con restricciones de acceso personalizadas.
- Cada registro permite almacenar uno o varios documentos (archivos que pueden ser de diversos formatos, por ejemplo, texto, imágenes, vídeo o audio), de acuerdo con la configuración del sistema.
- Cuenta con un sistema de búsqueda muy depurado que permite utilizar filtros para delimitar búsquedas por comunidad, colección, autor, título, año, temática, área de interés, clasificación, etc.
- El sistema de búsqueda permite localizar información dentro del texto completo de los documentos (para formatos de texto, PDF y MS Office).
- Permite recibir alertas a través del protocolo RSS.
- Las colecciones pueden contener archivos de texto, imágenes, sonido y vídeo. Los contenidos no textuales se enlazan con los documentos textuales o se acompañan de descripciones textuales (como las leyendas de las figuras) para poder buscar y consultar en modo de texto completo.
- Cada colección puede manejar permisos diferentes para cada usuario, de forma que los objetos que el usuario puede consultar irán variando dependiendo de los permisos asignados; de igual

forma es posible ocultar los objetos que no serán utilizados por los usuarios sin necesidad de borrarlos o darlos de baja de la colección.

- Un mismo objeto publicado puede mostrarse en tantas colecciones como se definan, sin necesidad de duplicarlo dentro de la plataforma.
- Al ser una plataforma de código abierto, DSpace no se adquiere como un software comercial, ni establece límites de concurrencia, almacenamiento o de descarga de objetos y documentos.

Alfresco Community 5.2

Alfresco es un sistema de gestión de contenidos de código abierto que permite a las organizaciones capturar, almacenar, buscar y colaborar en documentos de muchos tipos distintos. Se habla de gestion de contenidos porque no sólo ofrece a las empresas herramientas de gestión documental, también se ocupa de la gestión de contenido web, trabajo colaborativo, flujos de trabajo (mediante Activiti o jBoss jBPM). Esta versatilidad hace que Alfresco sea un producto dirigido a grandes y medianas empresas, que puedan aprovechar todas las ventajas que ofrece. (Revelles, 2014).

Ventajas

- Añade continuamente nuevas funcionalidades: integraciones (SAP, SharePoint, Salesforce), análisis de datos o flujos de trabajo entre otros.
- Alfresco One incluye una instancia de Alfresco en la nube, dando opciones para una gestión de contenidos sólida de forma hibrida.
- Alfresco Community es la edición gratuita para desarrolladores, entornos de testeo o pequeñas instalaciones.
- Gran comunidad de foros de soporte y resolución de dudas.
- El precio de su licencia es de los más bajos respecto a otros gestores de contenido similares.

Desventajas

- Sus licencias de pago se dirigen más bien a medianas y grandes empresas.
- Para lograr un rendimiento aceptable hace falta un servidor de amplia capacidad.
- Tener un entorno de alta disponibilidad puede aumentar el coste total de la propiedad(TCO) al añadir licencias al hardware necesario.
- Encontrar un buen proveedor o experto es imprescindible para una implementación correcta. En el libro blanco de Alfresco se expone cómo elegir al proveedor más adecuado para la implementación de este sistema.

Share de Alfresco Community 5.2

Alfresco Share es un cliente web introducido en la versión 3 de Alfresco que se basa en Spring Surf y YUI. Permite establecer espacios colaborativos, relacionados con la gestión documental y la gestión de contenidos. Ha sido concebido para la creación de portales colaborativos. La unidad fundamental de Alfresco Share es el sitio web (Revelles, 2014).

XML 1.0

XML significa lenguajes de marcas generalizado (Extensible Markup Language) Es un lenguaje usado para estructurar información en un documento o en general en cualquier fichero que contenga texto, como por ejemplo ficheros de configuración de un programa o una tabla de datos. Ha ganado muchísima popularidad en los últimos años debido a ser un estándar abierto y libre, creado por el consorcio World Wide Web, W3C (los creadores de la www) en colaboración con un panel que incluye representantes de las principales compañías productoras de software (Boulanger, 2015).

HTML 5

HTML es un lenguaje de marcado que se utiliza para el desarrollo de páginas de internet. Se trata de la sigla que corresponde a HyperText Markup Language, es decir, Lenguaje de Marcas de Hipertexto, que podría ser traducido como Lenguaje de Formato de Documentos para Hipertexto.

Se trata de un formato abierto que surgió a partir de las etiquetas SGML (Standard Generalized Markup Language). Concepto traducido generalmente como «Estándar de Lenguaje de Marcado Generalizado» y que se entiende como un sistema que permite ordenar y etiquetar diversos documentos dentro de una lista. Este lenguaje es el que se utiliza para especificar los nombres de las etiquetas que se utilizarán al ordenar, no existen reglas para dicha organización, por eso se dice que es un sistema de formato abierto (Diez et al., 2012).

JavaScript 1.6

JavaScript (JS) es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web, pero también usado en muchos entornos sin navegador, tales como node.js, Apache CouchDB y Adobe Acrobat. Es un lenguaje script multi-paradigma, basado en prototipos, dinámico, soporta estilos de programación funcional, orientada a objetos e imperativa.

El estándar de JavaScript es ECMAScript. Desde 2012 todos los navegadores más antiguos soportan por lo menos ECMAScript 3. El 17 de Julio de 2015 ECMA International publicó la sexta versión de

ECMAScript, la cual es oficialmente llamada ECMAScript 2015, y fue inicialmente nombrada como ECMAScript 6 o ES6. Desde entonces, los estándares ECMAScript están en ciclos de lanzamiento anuales (Eguíluz Pérez, 2012).

1.6 Conclusiones parciales

En el presente capítulo se abordaron los conceptos fundamentales de la investigación, se estudiaron algunos de los sistemas de repositorios existentes y al Módulo Depósito de Documentos, se analizaron los principales elementos que se consideran importantes para sustentar la propuesta de solución, así como las características fundamentales de las herramientas, tecnologías, lenguajes y metodologías de desarrollo que se utilizaran en la implementación del Módulo Incorporación de Documentos Digitales de repxos en la tecnología del Share de Alfresco 5.2.

Capítulo 2: Análisis y diseño de la propuesta de solución

En el presente capítulo se describe el Módulo de Incorporación de Documentos y se realiza el modelo de dominio definiendo los elementos manejados en el sistema junto a las relaciones establecidas entre estos. Se especifican los requerimientos funcionales y no funcionales del módulo. Además, se representa el diagrama de casos de uso del sistema unido a las descripciones de los casos de uso del mismo.

2.1 Descripción de la propuesta de solución

Después de analizar la situación problemática y conocer la necesidad de creación de una suit de Gestión Documental para CIGED, además de estudiar el Módulo Depósito de Documentos de Xabal Repxos 3.0 se propone como solución el desarrollo del Módulo Incorporación de Documentos Digitales de Xabal Repxos 3.0 en la tecnología del Share de Alfresco 5.2 utilizando para ello la misma configuración y validación de los metadatos asociados que utiliza el Módulo Depósito de Documentos, las colecciones que se encuentran registradas en Xabal Repxos 3.0, y Dspace como destino final para la incorporación del documento debido a las ventajas que este proporciona, el Módulo Incorporación de Documentos Digitales permite al usuario llenar el formulario describiendo así el documento que desea incorporar, después de llenar los campos del formulario con los metadatos, se envía el mismo para que sea revisado y si es aprobado se incorpora al depósito, si es rechazado se envía de regreso al usuario para que modifique el formulario.

2.2 Modelo de dominio

Un modelo de dominio es una representación de las clases conceptuales del mundo real, no de componentes software. No se trata de un conjunto de diagramas que describen clases software, u objetos software con responsabilidades (Larman, 2003).

La metodología AUP-UCI con el fin de esclarecer el entorno del problema define en su primera fase de desarrollo la realización de un Modelo de Casos de Uso del Negocio. Cuando los procesos del negocio no se encuentran bien identificados, AUP-UCI propone realizar un Modelo de Dominio. El Modelo de Dominio tiene como objetivo comprender y describir solamente las clases más importantes dentro del contexto en el cual se desempeña el software, con el propósito de sentar las bases del entendimiento del desarrollo y no para definirlo completamente.

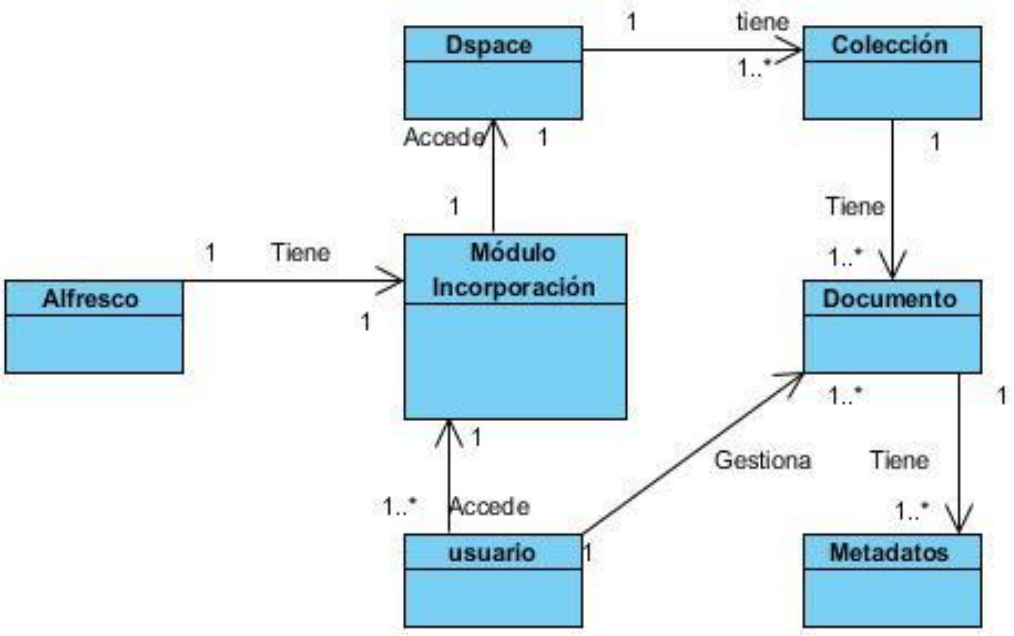


Fig 1:Modelo de dominio

Descripción del modelo de dominio

- Usuario: Persona que podrá interactuar con el módulo, también podrá realizar el rol de revisor.
- Documento: Tipo de archivo que será incorporado a Dspace, puede ser Word, pdf, foto y otros.
- Metadatos: Son los datos que van a describir el documento que el usuario desea incorporar.
- Colección: Es el destino donde se incorporarán los documentos.
- Módulo Incorporación de Documentos: Módulo al cual el usuario podrá realizar todas las acciones.
- Dspace: Sistema donde se encuentran las colecciones que van a recibir a los documentos

2.3 Especificación de Requisitos

El objetivo principal de la Especificación de Requisitos del Sistema (ERS) es servir como *medio de* comunicación entre clientes, usuarios, ingenieros de requisitos y desarrolladores. En la ERS deben recogerse tanto las necesidades de clientes y usuarios (necesidades del negocio, también conocidas como requisitos de usuario, requisitos de cliente, necesidades de usuario) como los requisitos que debe cumplir el sistema software a desarrollar para satisfacer dichas necesidades (requisitos del producto, también conocidos como requisitos de sistema o requisitos software) (Pollo Cattaneo, Britos, Pesado, & García Martínez, 2009).

Los requisitos se clasifican en dos tipos; los requisitos funcionales y los requisitos no funcionales. Los requisitos funcionales describen lo que el sistema debe hacer. Mientras que los requisitos no funcionales son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento.

Requisitos funcionales

Los requerimientos funcionales (RF) son capacidades o condiciones que el sistema debe cumplir, define qué es lo que el sistema debe hacer, así como las funciones que será capaz de realizar. Describen las transformaciones que el sistema realiza sobre las entradas, para producir salidas.

Tabla 1. Requisitos funcionales

No.	Nombre	Descripción	Nivel de acceso
RF1	Mostrar envíos rechazados.	El sistema debe permitir al usuario autenticado listar todos los envíos realizados por él, que tiene estado rechazado. De ellos se visualizarán los siguientes metadatos: Fecha de publicación, Título y Autores.	Público
RF2	Mostrar envíos aceptados.	El sistema debe permitir al usuario autenticado listar todos los envíos realizados por él, que tiene estado aceptado. De ellos se visualizarán los siguientes metadatos: Fecha de publicación, Título y Autores.	Público
RF3	Mostrar envíos no terminados.	El sistema debe permitir al usuario autenticado listar todos los envíos realizados por él, que tiene estado no terminado. De ellos se visualizarán los siguientes metadatos: Enviado por, Título y Enviado a (Colección). formulario.	Público.

RF4	Listar colecciones.	El sistema debe permitir al usuario autenticado visualizar todas las colecciones registradas en el sistema a la cuales tiene permiso.	Público
RF5	Registrar envío.	El sistema debe permitir al usuario autenticado registrar un nuevo envío.	Público
RF6	Modificar envío.	El sistema debe permitir al usuario autenticado modificar los envíos realizados por él, cuyo estado sea: no terminado y rechazado.	Público.
RF7	Eliminar envío.	El sistema debe permitir al usuario autenticado eliminar los envíos realizados por él cuyo estado sea: no terminado y rechazado.	Público
RF8	Guardar envío.	El sistema debe permitir al usuario autenticado guardar un envío sin iniciar el proceso de revisión.	Público
RF9	Borrar adjunto.	El sistema debe permitir al usuario autenticado borrar el o los adjuntos de un envío.	Público
RF10	Añadir adjunto.	El sistema debe permitir al usuario autenticado adjuntar uno o más ficheros a un envío.	Público
RF11	Iniciar flujo de revisión y aprobación	El sistema automáticamente debe iniciar el flujo de trabajo Revisión y aprobación cuando se registra un envío.	Público

RF12	Aprobar envío.	El sistema debe permitir al revisor asignado aprobar el envío.	Administrativo
RF13	Rechazar envío	El sistema debe permitir al revisor asignado rechazar el envío.	Administrativo
RF14	Archivar envío.	El sistema una vez aprobado el envío debe archivar los ficheros asociados al mismo en la colección correspondiente.	Administrativo
RF15	Notificar decisión de revisión.	El sistema debe notificar al creador del envío mediante un correo electrónico la decisión de la revisión.	Administrativo
RF16	Notificar nueva revisión.	El sistema debe notificar al grupo de revisores mediante un correo electrónico la asignación de la nueva revisión.	Público

Requisitos no funcionales

Los requerimientos no funcionales (RNF) son propiedades o cualidades que el producto debe tener, debe pensarse en propiedades que hacen al producto atractivo, usable, rápido o confiable. Son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este, como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento.

A continuación, se muestran los requisitos no funcionales que utiliza el Gestor de Documentos Administrativos Xabal eXcriba 3.1 y son utilizados por el módulo de gestión de registros de entrada y salida:

Usabilidad

1. Utilizar el idioma español para los mensajes y textos de la interfaz.

2. Permitir selección de algunos datos, predefinidos con anterioridad en los formularios.

Fiabilidad

1. La precisión y exactitud de las respuestas del módulo se corresponden con la calidad y exactitud de la información insertadas por los usuarios del sistema y la información contenida en el Gestor de Documentos Administrativos Xabal eXcriba 3.1.

Portabilidad

1. Se puede utilizar el módulo en sistemas operativos Windows y GNU/Linux. Se recomienda utilizar los de GNU/Linux.

Soporte

1. La estación de trabajo cliente debe tener instalado un navegador web. Se recomienda utilizar Mozilla Firefox 30.X o superior.

Legales

1. Las herramientas seleccionadas para el desarrollo del módulo están respaldadas por licencias libres, bajo las condiciones de software libre.

2.4 Especificación y resumen de los casos de uso del sistema.

Actores del sistema.

Un actor es una agrupación uniforme de personas, sistemas o máquinas externas al módulo que se está desarrollando, se relaciona con éste ya que solicita sus funcionalidades (Jacobson,2004). En el Módulo Incorporación interactúan dos actores, los cuales se definen a continuación:

- Usuario: Describirá a través de un formulario el documento que desea incorporar. Puede formar parte del grupo de revisión.
- Revisor: Encargado de revisar el documento y de ahí aprobar o rechazar el mismo.

Diagrama de casos de uso.

Para tener una visión general de los diferentes procesos de negocio de la organización, puede construirse un diagrama de casos de uso en el cual aparece cada proceso del negocio como un caso de uso. Este diagrama permite mostrar los límites y el entorno de la organización bajo estudio. Sólo se mostrarán en este diagrama los actores del negocio correspondientes a los roles externos al sistema, de forma que los

procesos de negocio en los que sólo tomen parte roles internos a la organización no estarán conectados a ningún actor. La figura 2 muestra el diagrama de casos de uso del sistema.

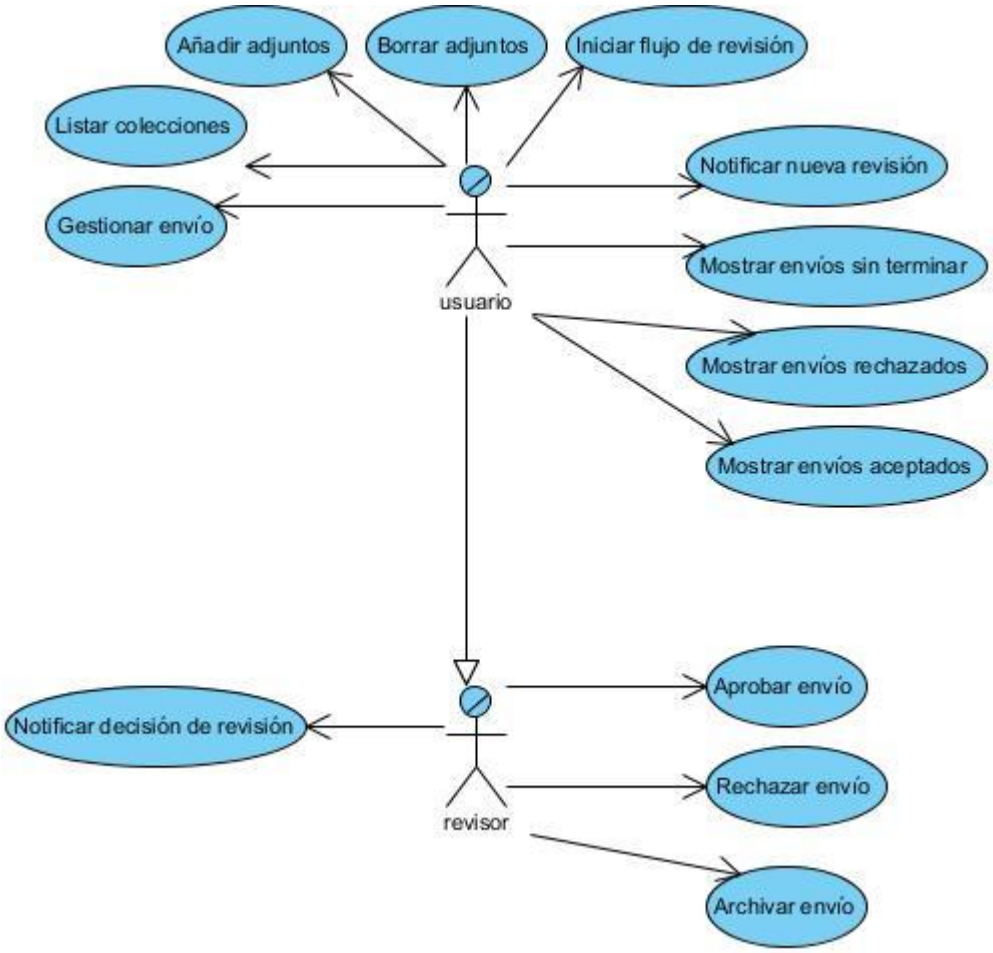
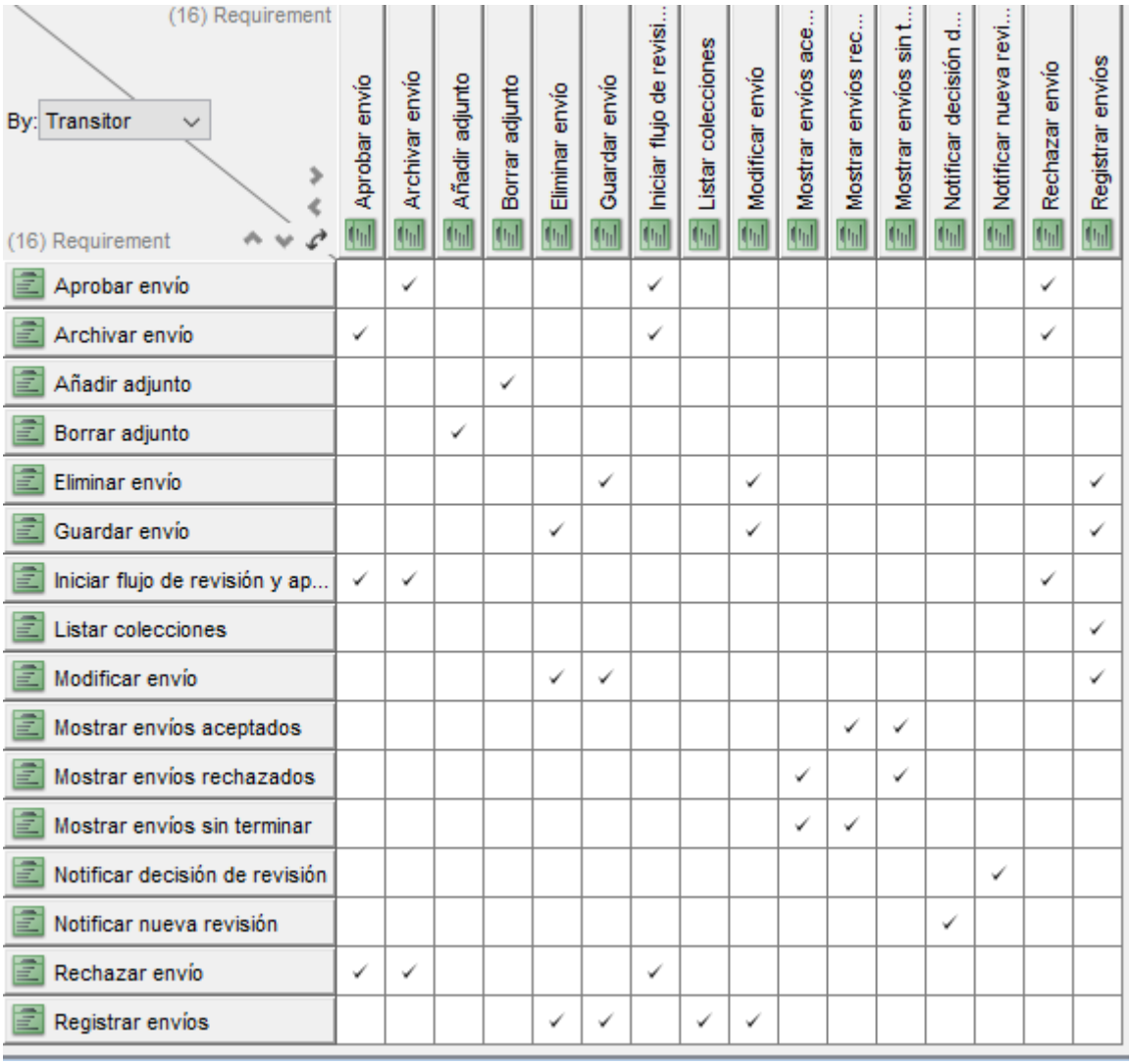


Fig 2: Diagrama de CUS

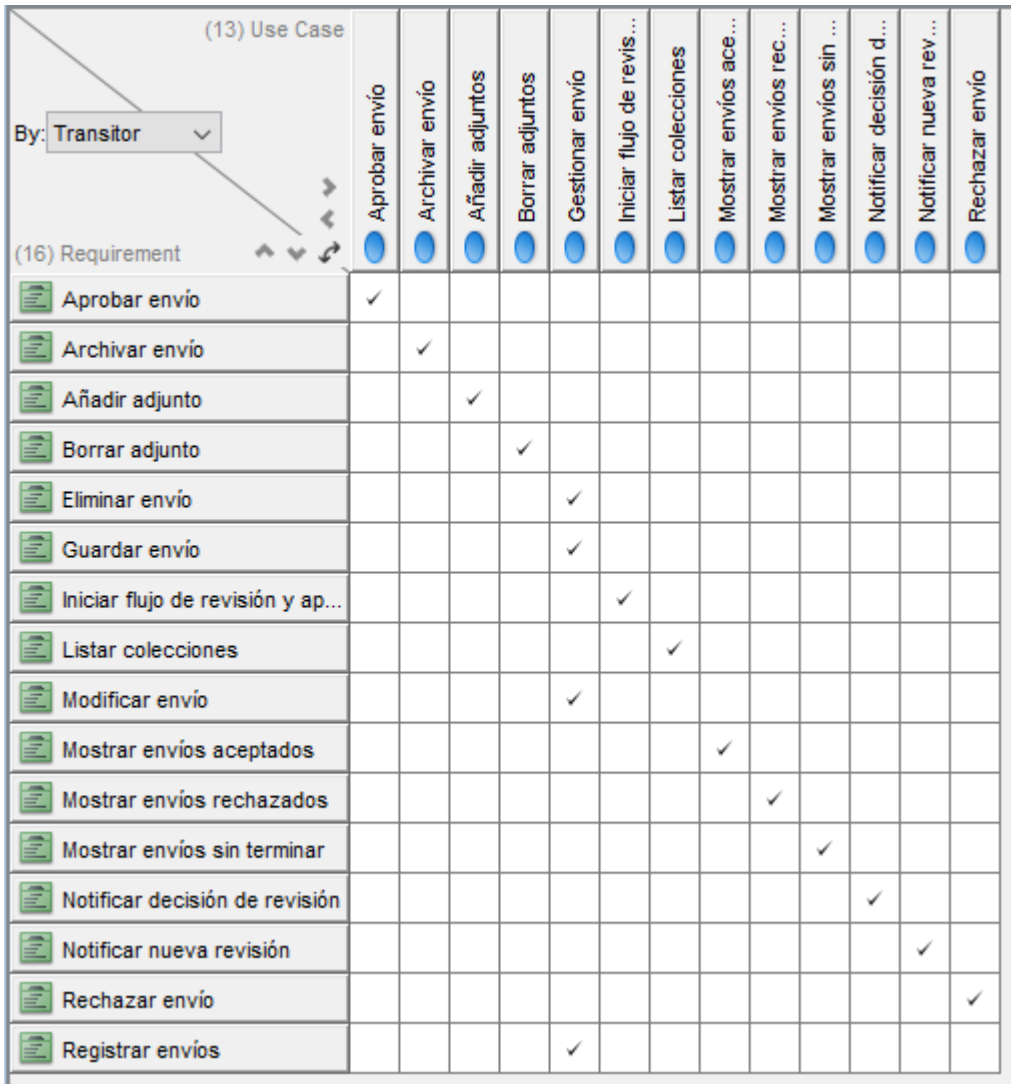
Matriz de trazabilidad

La matriz de trazabilidad de los requisitos del Proyecto tiene como propósito asegurar el cumplimiento de los requisitos validados. Esta actúa como un apoyo para mantener al día el estado de cada uno de los requisitos y especificaciones del proyecto. Este estado puede cambiar con el tiempo y en la matriz de trazabilidad se lleva el registro de estos cambios desde su inicio hasta su consecución. La matriz de trazabilidad mantiene dos versiones generales, un antes y un después, lo que es de gran utilidad (Macias, 2016).



(16) Requirement / By: Transitor / (16) Requirement	Aprobar envío	Archivar envío	Añadir adjunto	Borrar adjunto	Eliminar envío	Guardar envío	Iniciar flujo de revisi...	Listar colecciones	Modificar envío	Mostrar envíos ace...	Mostrar envíos rec...	Mostrar envíos sin t...	Notificar decisión d...	Notificar nueva revi...	Rechazar envío	Registrar envíos
Aprobar envío		✓					✓								✓	
Archivar envío	✓						✓								✓	
Añadir adjunto				✓												
Borrar adjunto			✓													
Eliminar envío						✓			✓							✓
Guardar envío					✓				✓							✓
Iniciar flujo de revisión y ap...	✓	✓													✓	
Listar colecciones																✓
Modificar envío					✓	✓										✓
Mostrar envíos aceptados											✓	✓				
Mostrar envíos rechazados										✓		✓				
Mostrar envíos sin terminar										✓	✓					
Notificar decisión de revisión														✓		
Notificar nueva revisión													✓			
Rechazar envío	✓	✓					✓									
Registrar envíos					✓	✓		✓	✓							

Fig 3: Matriz de requisitos-requisitos



(13) Use Case / By: Transitor / (16) Requirement	Aprobar envío	Archivar envío	Añadir adjuntos	Borrar adjuntos	Gestionar envío	Iniciar flujo de revis...	Listar colecciones	Mostrar envíos ace...	Mostrar envíos rec...	Mostrar envíos sin ...	Notificar decisión d...	Notificar nueva rev...	Rechazar envío
Aprobar envío	✓												
Archivar envío		✓											
Añadir adjunto			✓										
Borrar adjunto				✓									
Eliminar envío					✓								
Guardar envío					✓								
Iniciar flujo de revisión y ap...						✓							
Listar colecciones							✓						
Modificar envío					✓								
Mostrar envíos aceptados								✓					
Mostrar envíos rechazados									✓				
Mostrar envíos sin terminar										✓			
Notificar decisión de revisión											✓		
Notificar nueva revisión												✓	
Rechazar envío													✓
Registrar envíos					✓								

Fig 4: Matriz de requisitos-CUS

Fig 5: Matriz de requisitos-clases

Descripción de casos de uso del sistema

Tabla 2. CU1 Gestionar envío

Objetivo	Registrar, modificar, eliminar y guardar un envío.
Actores	Usuario: Registra, modifica, guarda y elimina un envío.
Resumen	El usuario selecciona realizar alguna acción sobre el envío.

Complejidad	Alta.	
Prioridad	Alta.	
Precondiciones	Para modificar o eliminar debe haber al menos un envío registrado en el sistema.	
Postcondiciones	-Envío registrado. -Envío modificado. -Envío guardado. -Envío eliminado.	
Flujo de eventos		
Flujo básico Registrar envío		
	Actor	Sistema
	Accede al módulo.	
		Permite realizar las siguientes acciones sobre un envío: -Registrar un nuevo envío. -Modificar envío. -Guardar envío. -Eliminar envío.
	Selecciona la opción “Comenzar un nuevo envío”, para registrar un nuevo envío.	
		Muestra las colecciones registradas en el sistema.
	Selecciona una colección.	
		Muestra un formulario en el cual se va a describir el documento que se va enviar.

	Completa el formulario.	
	Selecciona la opción “Siguiente”.	
		Comprueba que se hayan introducido los datos obligatorios. Estos campos se resaltan en la interfaz.
		Comprueba que los datos introducidos sean correctos.
		Muestra una tabla para adjuntar documentos.
	Selecciona la opción “Adjuntar Documento”	
		Añade el o los documentos a la tabla.
	Selecciona la opción “Enviar”	
		Envía el o los documentos hacia un flujo de revisión.
		Termina el caso de uso.
Flujos alternos		
5* Opción “Cancelar”		
	Actor	Sistema
	Selecciona la opción “Cancelar”.	
		Retorna a la página que le dio origen.
8* No se han introducido todos los datos obligatorio.		
	Actor	Sistema
		Comprueba que no se han introducido campos obligatorios.

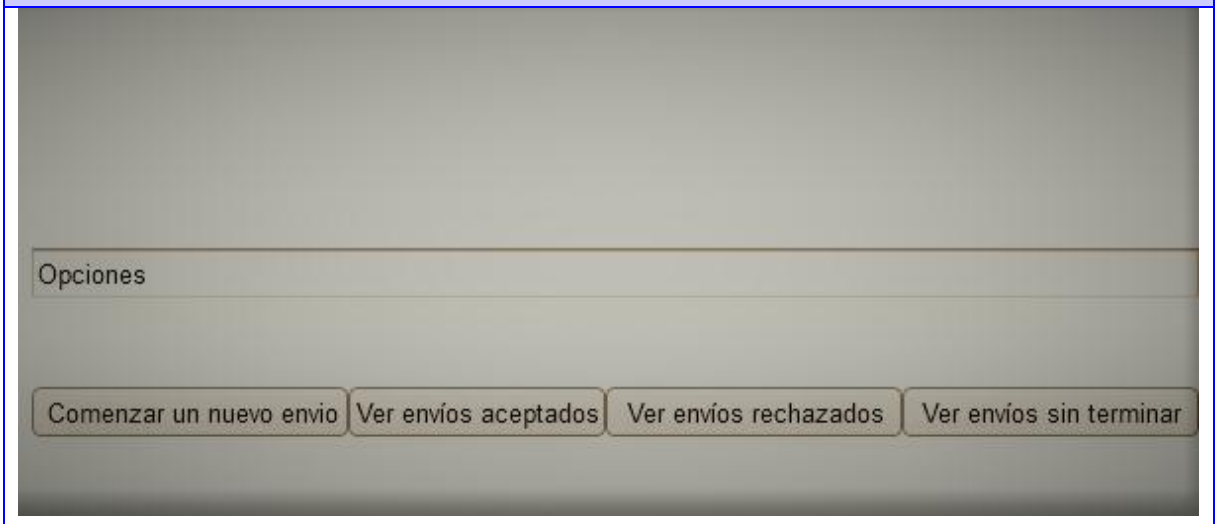
		Señala los campos que deben ser llenados y muestra un mensaje indicando que existen campos incompletos.
12* Opción cancelar.		
	Actor	Sistema
1.	Selecciona la opción “Cancelar”.	
2.		Retorna a la página que le dio origen.
13* Botón Eliminar		
	Actor	Sistema
	Selecciona la opción “Eliminar”	.
		Elimina un documento de la tabla.
Sección 1: “Modificar envío”		
Flujo básico Modificar envío		
	Actor	Sistema
	Selecciona la opción “Mostrar envíos rechazados” ó “Mostrar envíos no terminados”	
		Muestra una tabla con los envíos rechazados ó los no terminados respectivamente.
	Selecciona la opción “Editar”	
	Modifica los datos del envío.	
	Selecciona la opción “Aceptar”.	
		Comprueba que se han introducido los datos

		obligatorios. Estos campos se resaltan en la interfaz.
		Comprueba que los datos introducidos son correctos.
		Se modifica el envío.
		Termina el caso de uso.
Flujos alternos		
3* Opción “Cancelar”		
	Actor	Sistema
1.	Selecciona la opción cancelar.	
2.		Retorna a la página que le dio origen.
5* No se han introducido todos los datos obligatorios		
	Actor	Sistema
1.		Comprueba que no se han introducido campos obligatorios.
2.		Señala los campos que deben ser llenados y muestra un mensaje indicando que existen errores en el formulario.
Sección 2: “Guardar envío”		
Flujo Básico: Guardar envío		
	Actor	Sistema
1.	Selecciona la opción “Guardar”	
2.		Guarda el envío con los datos recibidos hasta ese momento.

3.		Termina el caso de uso.
Flujos alternos		
4a Opción “Cancelar”		
	Actor	Sistema
1.	Selecciona la opción “Cancelar”.	
2.		Regresa a la página que le dio origen.
Sección 3: “Eliminar envío”		
Flujo Básico: Eliminar envío.		
	Actor	Sistema
1.	Selecciona la opción “Mostrar envíos no terminados ” ó ”Mostrar envíos rechazados”	
2.		Muestra una tabla con los envíos rechazados ó los no terminados respectivamente
3.	Selecciona la opción “Eliminar”.	
4.		Se elimina un envío de la tabla
5.		Muestra un mensaje indicando el éxito de la operación y actualiza el listado de los envíos.
6.		Termina el caso de uso.
Flujos alternos		
3* Opción “Cancelar”		
	Actor	Sistema
1.	Selecciona la opción “Cancelar”.	

2.		Regresa a la página que le dio origen.

Prototipo elemental de interfaz gráfica de usuario



Gestionar envíos.

2.5 Arquitectura de software

El concepto de arquitectura de software se refiere a la estructuración del sistema que, idealmente, se crea en etapas tempranas del desarrollo. Esta estructuración representa un diseño de alto nivel del sistema que tiene dos propósitos primarios: satisfacer los atributos de calidad (desempeño, seguridad, modificabilidad), y servir como guía en el desarrollo. Al igual que en la ingeniería civil, las decisiones críticas relativas al diseño general de un sistema de software complejo deben de hacerse desde un principio. El no crear este diseño desde etapas tempranas del desarrollo puede limitar severamente que el producto final satisfaga las necesidades de los clientes. Además, el costo de las correcciones relacionadas con problemas en la arquitectura es muy elevado. Es así que la arquitectura de software juega un papel fundamental dentro del desarrollo (Escalante, 2013).

La arquitectura de software es de especial importancia ya que, la manera en que se estructura un sistema tiene un impacto directo sobre la capacidad de este para satisfacer lo que se conoce como los atributos de calidad del sistema. Ejemplos de atributos de calidad son el desempeño, que tiene que ver con el tiempo de respuesta del sistema a las peticiones que se le hacen, la usabilidad, que tiene que ver con qué tan sencillo les resulta a los usuarios realizar operaciones con el sistema, o bien la modificabilidad, que tiene que ver con qué tan simple resulta introducir cambios en el sistema. Los atributos de calidad son parte de los requerimientos (no funcionales) del sistema y son características que deben expresarse de forma

cuantitativa. No tiene sentido, por ejemplo, decir que el sistema debe devolver una petición “de manera rápida”, o presentar una página “ligera”, ya que no es posible evaluar objetivamente si el sistema cubre o no esos requerimientos (Cervantes, 2018).

Arquitectura N-capas

Esta arquitectura es una extensión de la arquitectura Capas tradicional. En el nivel más alto y abstracto, la vista de arquitectura lógica de un sistema puede considerarse como un conjunto de servicios relacionados agrupados en diversas capas. En Arquitecturas “Orientadas al Dominio” es crucial la clara delimitación y separación de la capa del Dominio del resto de capas. Es realmente un pre-requisito para DDD (Domain Driven Design) (Enriquez, 2017).

En las aplicaciones complejas, el comportamiento de las reglas de negocio (lógica del Dominio) está sujeto a muchos cambios y es muy importante poder modificar, construir y realizar pruebas sobre dichas capas de lógica del dominio de una forma fácil e independiente. Debido a esto, un objetivo importante es tener el mínimo acoplamiento entre el Modelo del Dominio (lógica y reglas de negocio) y el resto de capas del sistema (capas de presentación, capas de infraestructura, persistencia de datos). Así pues, las razones por las que es importante hacer uso de una “Arquitectura N-Capas Orientada al Dominio” es especialmente en los casos donde el comportamiento del negocio a automatizar (lógica del dominio) está sujeto a muchos cambios y evoluciones (Escalante, 2013).

Capa de presentacion:

Consiste en una interfaz gráfica que reúne los aspectos de software enfocados a la interacción con los diferentes tipos de usuarios. Es decir, incluye el manejo y aspecto de los formularios, la autentificación, el formato de los reportes, menús, gráficos y demás elementos multimedia. Durante el desarrollo del módulo esta se pone de manifiesto en los webscripts de presentación.

Capa intermedia

Reúne los aspectos de software que automatizan los procesos de negocio. Conocida también como capa de la Lógica de la Aplicación. Recibe la entrada de la capa anterior, interactúa con los servicios de datos para ejecutar las operaciones y envía el resultado procesado a la capa de presentación. Esta capa se pone de manifiesto en los archivos (css, js).

Capa de datos

Contiene los datos necesarios para la aplicación. Es la encargada de almacenarlos, recuperarlos y mantener su integridad. Estos datos consisten en cualquier fuente de información, incluido una base de datos de empresa como Oracle o MySQL, un conjunto de documentos XML o incluso un servicio de

directorio como LDAP. Además del tradicional mecanismo de almacenamiento relacional de base de datos, existen muchas fuentes diferentes de datos de empresa a las que pueden acceder las aplicaciones. Esta capa se pone de manifiesto en los webscripts de datos.

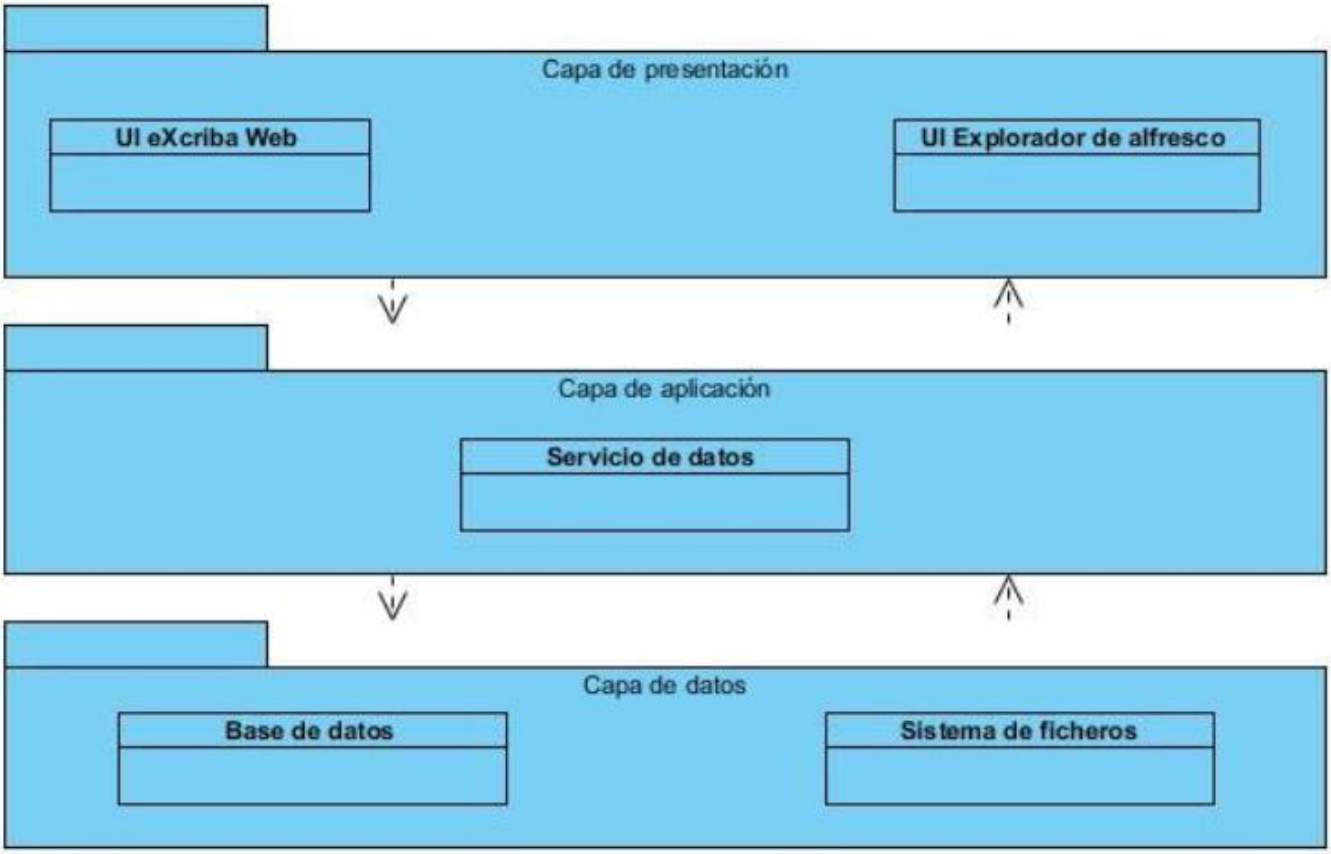


Fig 6. Diagrama de la arquitectura

Patrones de diseños

Es una solución general reusable que puede ser aplicada a problemas que ocurren comúnmente en el desarrollo de software, es la descripción o plantilla de cómo resolver un problema que puede ser usada en diferentes situaciones. Los patrones de diseño son soluciones probadas, expresivas y fáciles de mantener. Muchos desarrolladores están familiarizados con los patrones de diseño, así que podemos decir que es un tipo de estándar de desarrollo (Gros, Escofet, & Marimón, 2016).

Patrones Grasp

Se denominan patrones GRASP por sus siglas en inglés: *General Responsibility Assignment Software Patterns* o Patrones Generales de Software para Asignación de Responsabilidades. Como su nombre lo

indica, estos patrones nos indican cual es la manera de asignar responsabilidades a objetos software (Larman, 2003).

Experto

Es el encargado de asignar una responsabilidad al experto en información: la clase que tiene la información necesaria para realizar la responsabilidad. Nos indica, por ejemplo, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo.

El patrón experto posibilita que se mantenga el encapsulamiento de la información, puesto que los objetos utilizan su propia información para llevar a cabo las tareas. Normalmente, esto conlleva un bajo acoplamiento, lo que da lugar a sistemas más robustos y más fáciles de mantener, además se distribuye el comportamiento entre las clases que contienen la información requerida, por tanto, se estimula las definiciones de clases más cohesivas y ligeras que son más fáciles de comprender y manejar. Se soporta normalmente una alta cohesión. Este patrón se pone de manifiesto en los webscripts de datos.

Bajo Acoplamiento

La función de este patrón es asignar una responsabilidad de manera que el acoplamiento permanezca bajo. Un elemento con bajo acoplamiento no depende de demasiados elementos, estos elementos pueden ser clases, subsistemas, sistemas, etcétera.

Este patrón permite que en caso de modificarse algún elemento sus cambios no afecten a otro componente, además son fáciles de entender de manera aislada y conveniente para la reutilización. La utilización de este patrón se evidencia en la funcionalidad Mostrar envíos sin terminar.

Alta Cohesión

Su función es asignar una responsabilidad de manera que la cohesión permanezca alta. Un elemento con responsabilidades altamente relacionadas, y que no hace una gran cantidad de trabajo, tiene alta cohesión. Estos elementos pueden ser clases y subsistemas.

Posibilitando que se incremente la claridad y facilita la comprensión del diseño, simplifica el mantenimiento y las mejoras, soportando a menudo bajo acoplamiento y el grado bajo de funcionalidades altamente relacionadas incrementa la reutilización porque una clase cohesiva se puede utilizar para un propósito muy específico. Este patrón se evidencia en la funcionalidad Añadir adjunto.

Controlador

El patrón 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 según el método llamado.

Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto para aumentar la reutilización de código y a la vez tener un mayor control.

Se recomienda dividir los eventos del sistema en el mayor número de controladores para poder aumentar la cohesión y disminuir el acoplamiento. Se manifiesta en el fichero incorporación.js quien es el encargado de controlar casi todas las acciones en el módulo.

2.6 Conclusiones parciales

En este capítulo se realizó el análisis y diseño de la propuesta de solución, para ello se elaboró un modelo de dominio, se presentaron los requisitos funcionales y no funcionales, se definieron los casos de usos, así como su diagrama y descripción. Se definió la arquitectura y los patrones de diseño a utilizar. Lo anteriormente planteado permitió un mejor entendimiento del funcionamiento del negocio y trazó una guía para la implementación del módulo.

Capítulo 3: Implementación y pruebas

La implementación en el proceso de desarrollo de un software adquiere gran importancia debido a que le da funcionalidad al producto que se desarrolla. Además, son importantes las pruebas que se le realizan al mismo para validar su correcto funcionamiento. El capítulo abarca las fases de implementación y prueba del módulo. Se describen las pruebas realizadas a la aplicación con el objetivo de verificar si se cumplieron los requerimientos de la misma.

3.1 Implementación

Después de realizado un análisis del Módulo de Depósito de Documentos, de las tecnologías, lenguajes de programación y el modelado del negocio se implementó el Módulo Incorporación de Documentos ya que, a partir de los resultados del Análisis y Diseño se construye el sistema.

En la implementación del módulo:

- Se crearon 23 webscripts de presentación las cuales serán las interfaces que interactuaran con los usuarios. Para la implementación de las mismas se utilizó HTML 5, Bootstrap 2.3.2 para el estilo de las vistas y JavaScript para validar.
- Se implementaron 10 webscripts de datos a traves de las cuales se logró guardar, buscar, actualizar y eliminar nodos de la base de datos, para ello se utilizó JavaScript 1.6.

Se programó utilizando el estandar de codificación “Identación y líneas” usar una identación sin tabulaciones, con un equivalente a cuatro espacios. El uso de las llaves es seguido del nombre del método. La longitud de las líneas de código es aproximadamente de 75-80 caracteres para mantener la legibilidad del código, se utilizó tambien “Escritura en camello” la cual consiste en que el nombre de las funciones es con minúsculas, y en caso de nombres compuestos la primera palabra de cada letra comienza con mayúscula (del Carmen Ramírez & Fuentes, 2014).

3.2 Diagrama de despliegue

Un diagrama de despliegue es la forma de mostrar la configuración de nodos de procesamientos en tiempo de ejecución y los componentes que en ellos residen. Estos nodos forman la topología de hardware sobre el que se ejecuta el sistema. Este diagrama se preocupa principalmente de la distribución, entrega e instalación de las partes que constituye el sistema físico (Taboada, 2009).

En UML, los aspectos estáticos se capturan en los diagramas de despliegue; los aspectos dinámicos se capturan en los diagramas de iteración, diagrama de estados y diagrama de actividades.

Un diagrama de despliegue consta de la interconexión de nodos a partir de relaciones de asociación. En este contexto, las relaciones representan enlaces físicos (normalmente bidireccionales), como es el caso de una conexión directa mediante cables o indirecta por vía satélite (Taboada, 2009).

A continuación, se muestra el diagrama de despliegue del Módulo Incorporación de Documentos Digitales de Repxos en la tecnología del Share de Alfresco 5.2.

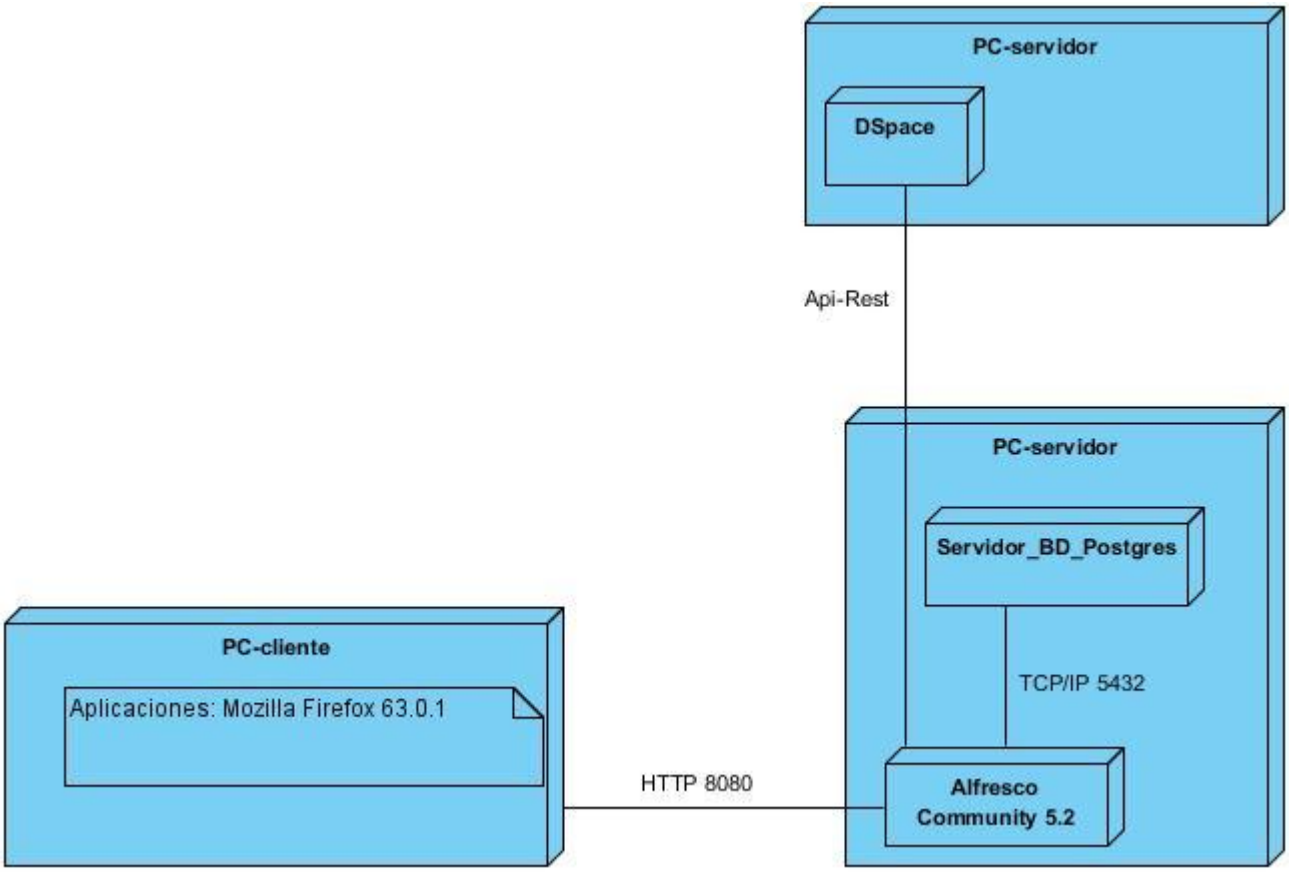


Fig 7: Diagrama de despliegue

Este diagrama muestra la estructura física del sistema, incluyendo las relaciones mediante protocolos entre el hardware y el software. Como se muestra en la figura, la distribución física del sistema en tiempo de ejecución consta de 3 nodos. En uno se encuentra la PC cliente, que debe tener instalado un navegador web, el cual hace peticiones a la PC servidor donde se encuentra instalado Alfresco Community 5.2 usando el protocolo de comunicación HTTP por el puerto 8080. Los otros dos nodos son servidores, los cuales se comunican e intercambian datos usando Api-Rest.

3.3 Pruebas de software

Las pruebas de software comprenden el conjunto de actividades que se realizan para identificar posibles fallos de funcionamiento, configuración o usabilidad de un programa o aplicación, por medio de pruebas sobre el comportamiento del mismo. Los sistemas informáticos, programas y aplicaciones han crecido a niveles inimaginables en complejidad e interoperabilidad, con lo cual también se han incrementado las posibilidades de defectos, a simple vista insignificantes, pero que pudieran adquirir proporciones catastróficas.

Además factores como el uso de software de terceros desde aplicaciones móviles han añadido niveles adicionales de complejidad y por ende incrementado los posibles puntos de fallas. Frente a esto, el reto de los profesionales de es modernizar sus metodologías, tecnologías y herramientas que les permitan automatizar tareas, ejecutar ciclos de pruebas más rápidos y así reducir al mínimo las posibilidades de errores en el Software (PMOinformatica, 2018).

La metodología AUP-UCI propone tres disciplinas de prueba: pruebas internas, pruebas de liberación y pruebas de aceptación (UCI, 2015).

Pruebas internas: En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas. Se deben desarrollar artefactos de prueba como: diseños de casos de prueba, listas de chequeo y de ser posible componentes de prueba ejecutables para automatizar las pruebas.

Pruebas de liberación: Pruebas diseñadas y ejecutadas por una entidad certificadora de la calidad externa, a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación.

Pruebas de aceptación: Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido.

Métodos de prueba

Los métodos de prueba definen la estrategia a seguir en función de la verificación y validación del sistema diseñado para descubrir fallos. Los métodos estudiados fueron las pruebas de caja blanca y las pruebas de caja negra (Sommerville, 2005).

Caja negra

Las pruebas de caja negra también denominadas pruebas de comportamiento, se centran en los requisitos funcionales del software. O sea, la prueba de caja negra permite obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa (ÁLVAREZ, 2009). Esta prueba intenta encontrar errores de las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz, errores en estructura de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y terminación

A continuación se expondrán las técnicas de pruebas más utilizadas dentro de la categoría de caja negra. Estas incluyen pruebas de partición equivalente, análisis del valor limite, tablas de decisión, diagramas de causa-efecto y arreglos ortogonales.

Partición equivalente

La técnica de partición equivalente, muy común entre los probadores y programadores, consiste básicamente en identificar y categorizar entradas que sean tratadas de manera similar por un sistema, y que produzcan el mismo resultado. Siendo más formales, la partición equivalente consta de clases de equivalencia que representan un conjunto de estados validos o inválidos para condiciones de entradas determinadas.

Tabla 3. CP1 Gestionar envío

SC1: Gestionar envío.							
Escenario	**Descripción**	**Nombre**	**Apellidos**	**Titulo**	**Fecha**	**Respuesta del sistema**	**Flujo central**
EC1.1 Modificar envío	Permite al usuario modificar un envío seleccionado.	V	V	V	V	El sistema actualiza los nuevos datos.	-Selecciona “Editar”. -Modifica los datos. -Selecciona “Aceptar”.
		Isidro	Padrón	Visión empresarial.	12/04/2019		

EC1.2 Campos obligatorios vacíos.	El escenario de prueba permite al usuario validar los campos vacíos.	I	I	I	I	El sistema señala los campos obligatorios vacíos, y muestra el mensaje ”Campos incompletos”	Selecciona “Editar”. -Modifica los datos. -Selecciona “Aceptar”.
EC1.3 Validar campos incorrectos.	El escenario de prueba permite al usuario validar los campos del formulario.	I	I	I	I	El sistema muestra el mensaje: "Solo acepta los valores a-z, A-Z, 0-9, [ ].".	Selecciona “Editar”. -Modifica los datos. -Selecciona “Aceptar”.
EC1.4 Cancelar acción	El escenario de prueba permite al usuario autenticado cancelar la acción.	NA	NA	NA	NA	El sistema muestra la página de origen.	Selecciona “Editar”. -Modifica los datos. -Selecciona “Cancelar”

Caja blanca

Las pruebas de caja blanca, también llamadas pruebas de caja de cristal, permiten examinar la estructura interna de uno o varios componentes que están siendo puestos a pruebas con el fin de crear casos de prueba basados en la implementación de dichos componentes. Estos casos de pruebas generados por los métodos de caja blanca tienen la finalidad de (ÁLVAREZ, 2009):

- Garantizar que todos los caminos independientes dentro de un módulo se han ejercitado al menos una vez.
- Ejercitar todas las decisiones lógicas bien sean falsas y/o verdaderas.
- Ejecutar todos los bucles en sus respectivos límites y dentro de sus límites operacionales.
- Ejercitar las estructuras internas de datos con el fin de asegurar su integridad.

Pruebas de ruta básica.

La prueba de ruta básica, propuesta por Tom McCabe, permite al analista de pruebas derivar una medida de complejidad lógica a partir de un diseño procedimental, y usar esta medida como guía para definir un conjunto base de rutas de ejecución. Los casos de prueba resultantes del conjunto base de rutas, ejecutarán todas las sentencias del programa al menos una vez durante las pruebas.

El método de ruta básica puede ser aplicado tanto a un diseño procedimental como a código fuente, y presenta una serie de pasos mencionados a continuación.

- Usando el diseño o el código como base, dibujar el correspondiente diagrama de flujo.
- Determinar la complejidad ciclomática del diagrama de flujo resultante.
- Determinar un conjunto base de caminos linealmente independientes.
- Preparar los casos de pruebas que obliguen la ejecución de cada camino perteneciente al conjunto base.

Fig 8: Código función guardar documentos

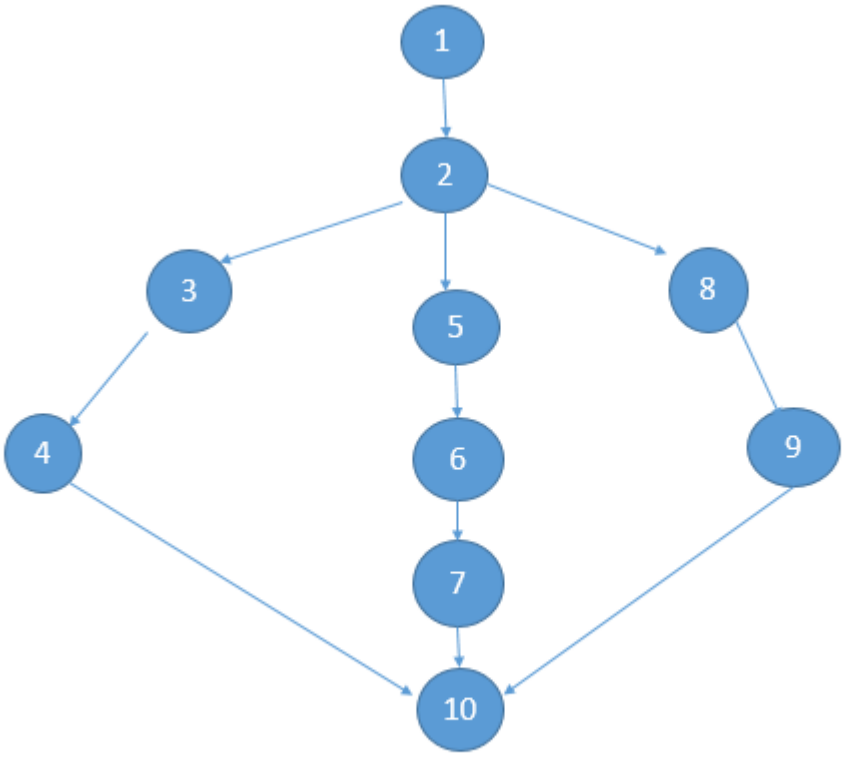


Fig 9: Grafo resultante

Complejidad ciclomática

V (G)= # de regiones =3

V (G)=A-N+2 =11-10+2=3

V (G)= P+1 =2+1=3

Caminos lógicos:

Camino1:1,2,3,4,10

Camino2:1,2,5,6,7,10

Camino3:1,2,8,9,10

Cada camino independiente es un caso de prueba a realizar, de forma que se garantiza que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa. En el caso anterior se calcularon 3 caminos básicos, por tanto surge la necesidad de hacer igual número de casos de prueba. A continuación se muestra uno de los casos de pruebas realizados.

Tabla 4 Caso de prueba de caja blanca para los caminos básicos

Caso de prueba 1 para el camino básico	
Descripción Si no existe la carpeta incorporación, se crea una con ese nombre, y adentro otra de tipo userName, luego otra con el nombre de documento y se envía la dirección al servicio adjuntar documentos.	
Condición de ejecución Se crea la carpeta	
Datos de entrada	
Tipo de dato esperado	
Evaluación del caso de prueba: Satisfactoria	
Caso de prueba 2 para el camino básico	
Descripción Si existe la carpeta incorporación, y no existe dentro de ella una de tipo userName, se crea una carpeta de tipo userName, luego otra con el nombre de documentos y se envía la dirección al servicio adjuntar documentos.	
Condición de ejecución Se crea la carpeta	
Datos de entrada	
Tipo de dato esperado	
Evaluación del caso de prueba: Satisfactoria	

Caso de prueba 3 para el camino básico	
Descripción Si existe la carpeta incorporación, y existe dentro de ella una de tipo userName, y no existe una con el nombre de documentos, se crea una con el nombre de documentos y se envía la dirección al servicio adjuntar documentos.	
Condición de ejecución Se crea la carpeta	
Datos de entrada	
Tipo de dato esperado	
Evaluación del caso de prueba: Satisfactoria	

Resultados de las pruebas

La realización de pruebas al módulo permitió detectar varias no conformidades en las primeras iteraciones siendo estas resueltas. Con la conclusión de esta fase de pruebas, fue posible comprobar que el módulo cumple con las especificaciones que se trazaron en los requisitos definidos. A continuación, se muestra la relación de no conformidades (detectadas y resueltas) por iteración:

Tabla 5. Resultados de las pruebas

Iteración	NC detectadas	Asociadas a:	NC resueltas
1ra	15	Errores ortográficos Campos obligatorios incompletos	15
2da	3	Errores ortográficos	3
3ra	-	-	-

3.4 Conclusiones parciales

En este capítulo se definió el estilo de codificación utilizado en la implementación del sistema, lo que permitió mantener una uniformidad en la codificación y mayor claridad a la hora de leer o agregar líneas de código. El modelo de despliegue definido mostró con mayor claridad la distribución física del sistema sobre una arquitectura de hardware. Luego de la realización de pruebas de caja negra y caja blanca, se comprobó el correcto funcionamiento del módulo. A partir del empleo de estos tipos de pruebas se encontraron un conjunto de no conformidades, las cuales fueron resueltas, las pruebas realizadas brindaron un resultado satisfactorio que le aporta al sistema mayor nivel de robustez y validez.

Conclusiones generales

Durante el desarrollo de la investigación se planteó la necesidad de diseñar e implementar el Módulo Incorporación de Documentos Digitales de Repxos en la tecnología del Share de Alfresco 5.2, dándole así cumplimiento al objetivo planteado de la siguiente forma:

- Con el estudio del Módulo Depósito de Documentos se evidenciaron las características de los sistemas de repositorios, tecnologías y lenguajes factibles para darle solución a la problemática planteada.
- Se diseñó el módulo basándose en los artefactos propuestos por la metodología AUP-UCI.
- Se realizaron pruebas a las funcionalidades del módulo, para verificar la calidad de la solución propuesta. Se comprobó que el software cumple con todas las funcionalidades definidas.

Se implementó el Módulo Incorporación de Documentos Digitales, el cual permite a los usuarios de Alfresco incorporar documentos en Dspace.

Recomendaciones

Se recomienda integrar al módulo la opción de además de incorporar el documento, imprimir el mismo, se recomienda además integrar un chat que posibilite la comunicación entre los usuarios de Alfresco y otros usuarios del centro.

Referencias bibliográficas

1. ÁLVAREZ, A. C. (2009). METODOLOGÍAS DE TESTING DE SOFTWARE Y SU APLICACIÓN EN EL CENTRO DE INFORMÁTICA DE LA UNIVERSIDAD EAFIT. https://repository.eafit.edu.co/xmlui/bitstream/handle/10784/2744/RuizCalle_JuanDavid_2009.pdf;jsessionid=D1293D1288305AB13CA000316D9CDE65?sequence=1

2. Boulanger, T. (2015). *XML práctico: Bases esenciales, conceptos y casos prácticos (2ª edición)*: Ediciones ENI.

3. Canós, J. H., & Letelier, M. C. P. P. (2012). Metodologías ágiles en el desarrollo de software.

4. Cervantes, H. (2018). SG. https://sg.com.mx/revista/27/arquitectura-software

5. de Dios Arias, R. A., Cano Inclán, A., García García, O., & Raposo Villavicencio, R. M. (2015). Diseño de un sistema de gestión documental para organizaciones cubanas. *Revista Cubana de Información en Ciencias de la Salud, 26*(3), 260-272.

6. del Carmen Ramírez, L., & Fuentes, A. S. F. (2014). Buenas prácticas, una solución para un mejor desarrollo de software. *Mundo FESC, 2*(8), 37-45.

7. del Prado Martínez, M. Á., & Esteban Navarro, M. Á. (2016). Propuesta de un modelo de Sistema Integrado de Gestión de la Información Documental para las organizaciones.

8. Diez, T., Domínguez, M. J., Martínez, J. J., & Sáenz, J. (2012). Creación de páginas Web accesibles con HTML5. *Consultado el, 26*.

9. Doria, M. V., Prado, A. M. d., & Haustein, M. C. (2015). Repositorios digitales y software open source. *TE & ET*.

10. Eguíluz Pérez, J. (2012). Introducción a JAVASCRIPT.

11. Enriquez, J. l. (2017). Metología de desarrollo de software. https://www.uladech.edu.pe/images/stories/universidad/documentos/2018/metodologia-desarrollo-software-v001.pdf

12. Escalante, L. C. (2013). *El patrón de arquitectura n-capas con orientación al dominio como solución en el diseño de aplicaciones empresariales*.

13. Gairin, J. M. R. (2018). *DSpace: Un manual especifico para gestores de la información y la documentación*. 6.

14. Gallo, P. R. (2011). *Gestión documental en las organizaciones*: Editorial UOC.

15. Giménez Chornet, V. (2015). Normas ISO para la gestión de los documentos electrónicos: buenas prácticas para la gestión documental en las empresas. *Fuentes, Revista de la Biblioteca y Archivo Histórico de la Asamblea Legislativa Plurinacional, 9*, 7.

16. Ginestà, M. G., & Mora, O. P. (2012). Bases de datos en PostgreSQL. *Sl]:[sn]*.

17. Gros, B., Escofet, A., & Marimón, M. (2016). Los patrones de diseño como herramientas para guiar la práctica del profesorado/The design patterns as tools to guide the practice of teachers. *Revista Latinoamericana de Tecnología Educativa-RELATEC, 15*(3), 11-25.

18. Larman, C. (2003). *UML y Patrones*: Pearson Educación ^ eMadrid Madrid.

19. López, E. V. (2017). Las herramientas tecnológicas al servicio de gestión empresarial y administrativa. *Visionario Digital, 1*(4), 45-61.

20. López, F.-A. (2013). Visibilidad e impacto de los repositorios digitales en acceso abierto. *De bibliotecas y bibliotecarios... Boletín electrónico ABGRA*(5).

21. Macias, J. (2016). MDAP. https://uv-mdap.com/blog/matriz-trazabilidad-requisitos-del-proyecto/

22. MARTÍN LÓPEZ-ASÚNSOLO, A. (2017). Cobertura de código e Informes de Error en Eclipse: más información para evaluar TESTAR.

23. Martínez, A. C., & de Mingo, A. C. (2018). El impacto de la gestión documental en la transparencia de las Administraciones públicas: la transparencia por diseño. *Gestión y Análisis de Políticas Públicas*(19).

24. Merlino-Santesteban, C. (2012). *Repositorios digitales institucionales*.

25. Morgado, E. M. M., Ortuño, R. A. C., & Fernández, T. F. (2018). *Ampliación de metadatos educativos en el repositorio Gredos: Proyecto DIRED.* Paper presented at the Ecosistemas del Acceso Abierto.

26. Peña, D. M. (2016). *Extensión de la herramienta Visual Paradigm for UML para la evaluación y corrección de Diagramas de Casos de Uso.* Universidad de las Ciencias Informáticas.

27. Pollo Cattaneo, M. F., Britos, P., Pesado, P. M., & García Martínez, R. (2009). *Metodología para especificación de requisitos en proyectos de explotación de información.* Paper presented at the XI Workshop de Investigadores en Ciencias de la Computación (San Juan, Argentina).

28. Ramirez, L. (2016). Manual de Dspace.

29. Revelles, J. S. (2014). *Alfresco. La gestión de contenidos empresarial*.

30. Rolando Alfredo, S. C. (2012). *El proceso de investigación cientifica*: Editorial Universitaria.

31. Rovitto, F., & Nisim, A. O. (2018). *Tecnología sustentable IoT-Mobile: sistema de monitoreo y diagnóstico de acuarios.* Paper presented at the XXI Concurso de Trabajos Estudiantiles (EST)-JAIIO 47 (CABA, 2018).

32. Ruiz, G., & Piere, E. (2017). Implementación de un sistema de información bajo plataforma web para la gestión y control documental de la empresa corporación Jujedu EIRL–Talara; 2017.

33. Sarduy Domínguez, Y., & Urra González, P. (2006). Herramientas para la creación de colecciones digitales. *Acimed, 14*(5), 0-0.

34. Sommerville, I. (2005). *Ingeniería del software*: Pearson Educación.

35. Taboada, M. C. (2009). Diagrama de despliegue.

36. Torres, I. C., Morales, I. C., Hernández, T. R., Matos, M. M., & Palomino, M. P. (2016). La gestión de publicaciones científicas en el ámbito a las Ciencias de la información. *Revista Publicando, 3*(6), 164-174.

37. UCI. (2019). Retrieved from https://www.uci.cu/investigacion-y-desarrollo/productos/xabal/repxos-30

38. Vargas-Arcila, A. M., Baldassarri, S., & Arciniegas, J. L. (2016). Análisis de Esquemas de Metadatos para la Marcación de Contenidos Educativos. *Formación universitaria, 9*(5), 85-96.

Bibliografía

1. Aja Quiroga, L. (2002). Gestión de información, gestión del conocimiento y gestión de la calidad en las organizaciones. *Acimed, 10*(5), 7-8.

2. Alonso, E. M. (2009). Monografias.com. https://www.monografias.com/trabajos73/herramientas-case-proceso-desarrollo-software/herramientas-case-proceso-desarrollo-software2.shtml

3. ÁLVAREZ, A. C. (2009). METODOLOGÍAS DE TESTING DE SOFTWARE Y SU APLICACIÓN EN EL CENTRO DE INFORMÁTICA DE LA UNIVERSIDAD EAFIT. https://repository.eafit.edu.co/xmlui/bitstream/handle/10784/2744/RuizCalle_JuanDavid_2009.pdf;jsessionid=D1293D1288305AB13CA000316D9CDE65?sequence=1

4. ÁLVAREZ, A. C. (2009). METODOLOGÍAS DE TESTING DE SOFTWARE Y SU APLICACIÓN EN EL CENTRO DE INFORMÁTICA DE LA UNIVERSIDAD EAFIT. https://repository.eafit.edu.co/xmlui/bitstream/handle/10784/2744/RuizCalle_JuanDavid_2009.pdf;jsessionid=D1293D1288305AB13CA000316D9CDE65?sequence=1

5. Álvarez, A. P., & Alcolea, P. G. (2004). *Estudio del modelo de productos y servicios de las empresas de servicios documentales.* Paper presented at the Anales de Documentación.

6. Amaya, J. (2016).

7. Artiles Visbal, S. M. (2009). La gestión documental, de información y el conocimiento en la empresa: El caso de Cuba. *Acimed, 19*(5), 0-0.

8. Boulanger, T. (2015). *XML práctico: Bases esenciales, conceptos y casos prácticos (2ª edición)*: Ediciones ENI.

9. Blake, L. (Producer). (2018). Retrieved from http://volaya.github.io/libro-sig/chapters/Metadatos.html

10. Cabreras, A. P. (2015). Módulo Depódito de Documentos para el sistema de Repositorios Digitales Repxos 3.0. Habana.

11. Canela López, J. (2004). La gestión por calidad total en la empresa moderna. *México: ALFA OMEGA GRUPO EDITOR SA de CV*.

12. Cano Inclán, A., de Dios Arias, R. A., García García, O., & Cuesta Rodríguez, F. (2015). Los repositorios institucionales: situación actual a nivel internacional, latinoamericano y en Cuba. *Revista Cubana de Información en Ciencias de la Salud, 26*(4), 0-0.

13. Canós, J. H., & Letelier, M. C. P. P. (2012). Metodologías ágiles en el desarrollo de software. Cervantes, H. (2018). SG. https://sg.com.mx/revista/27/arquitectura-software

14. Canós, J. H., & Letelier, M. C. P. P. (2012). Metodologías ágiles en el desarrollo de software.

15. Cervantes, H. (2018). SG. https://sg.com.mx/revista/27/arquitectura-software

16. de los Baños, A. Desarrollo del proceso de Demanda conjunta del subsistema de Disciplina y derecho laboral del Sistema de Informatización para la Gestión de los Tribunales Populares Cubanos Development of the process of together demand of the Discipline and labor law subsystem of the Computerization System for the.

17. del Carmen Ramírez, L., & Fuentes, A. S. F. (2014). Buenas prácticas, una solución para un mejor desarrollo de software. *Mundo FESC, 2*(8), 37-45.

18. de Dios Arias, R. A., Cano Inclán, A., García, O., & Raposo Villavicencio, R. M. (2015). Diseño de un sistema de gestión documental para organizaciones cubanas. *Revista Cubana de Información en Ciencias de la Salud, 26*(3), 260-272.

19. del Carmen Ramírez, L., & Fuentes, A. S. F. (2014). Buenas prácticas, una solución para un mejor desarrollo de software. *Mundo FESC, 2*(8), 37-45.

20. del Prado Martínez, M. Á., & Esteban Navarro, M. Á. (2016). Propuesta de un modelo de Sistema Integrado de Gestión de la Información Documental para las organizaciones.

21. Diez, T., Domínguez, M. J., Martínez, J. J., & Sáenz, J. (2012). Creación de páginas Web accesibles con HTML5. *Consultado el, 26*.

22. Doria, M. V., Prado, A. M. d., & Haustein, M. C. (2015). Repositorios digitales y software open source. *TE & ET*.

23. Diez, T., Domínguez, M. J., Martínez, J. J., & Sáenz, J. (2012). Creación de páginas Web accesibles con HTML5. *Consultado el, 26*.

24. Eguíluz Pérez, J. (2012). Introducción a JAVASCRIPT.

25. Enriquez, J. l. (2017). Metología de desarrollo de software. https://www.uladech.edu.pe/images/stories/universidad/documentos/2018/metodologia-desarrollo-software-v001.pdf

26. Eguíluz Pérez, J. (2012). Introducción a JAVASCRIPT.

27. Enríquez, J. l. (2017). Metodología de desarrollo de software. https://www.uladech.edu.pe/images/stories/universidad/documentos/2018/metodologia-desarrollo-software-v001.pdf

28. Escalante, L. C. (2013). *El patrón de arquitectura n-capas con orientación al dominio como solución en el diseño de aplicaciones empresariales*.

29. Escalante, L. C. (2013). *El patrón de arquitectura n-capas con orientación al dominio como solución en el diseño de aplicaciones empresariales*.

30. FLORES, W. X. V., & ESCOBAR, J. E. F. (2017). Repositório Digital: GRUPO COMPÁS.

31. Fuster Ruiz, F. (1999). *Archivística, archivo, documento de archivo... Necesidad de clarificar los conceptos.* Paper presented at the Anales de Documentación.

32. Gairin, J. M. R. (2018). *DSpace: Un manual especifico para gestores de la información y la documentación*. 6.

33. Gallo, P. R. (2011). *Gestión documental en las organizaciones*: Editorial UOC.

34. Giménez Chornet, V. (2015). Normas ISO para la gestión de los documentos electrónicos: buenas prácticas para la gestión documental en las empresas. *Fuentes, Revista de la Biblioteca y Archivo Histórico de la Asamblea Legislativa Plurinacional, 9*, 7.

35. Ginestà, M. G., & Mora, O. P. (2012). Bases de datos en PostgreSQL. *Sl]:[sn]*.

36. Gros, B., Escofet, A., & Marimón, M. (2016). Los patrones de diseño como herramientas para guiar la práctica del profesorado/The design patterns as tools to guide the practice of teachers. *Revista Latinoamericana de Tecnología Educativa-RELATEC, 15*(3), 11-25.

37. HERNÀNDEZ, Z. G. (2013). *Herramientas case*.

38. Keefer, A. (2007a). Los repositorios digitales universitarios y los autores. Redalyc. (1575-2437). https://www.redalyc.org/articulo.oa?id=63501011

39. Keefer, A. (2007b). *Los repositorios digitales universitarios y los autores.* Paper presented at the Anales de Documentación.

40. Konetzke, R. (1962). *Colección de documentos para la historia de la formación social de Hispanoamérica* (Vol. 3): Consejo superior de investigaciones cientificas.

41. Larman, C. (2003). *UML y Patrones*: Pearson Educación ^ eMadrid Madrid.

42. López, F.-A. (2013). Visibilidad e impacto de los repositorios digitales en acceso abierto. *De bibliotecas y bibliotecarios... Boletín electrónico ABGRA*(5).

43. López, E. V. (2017). Las herramientas tecnológicas al servicio de gestión empresarial y administrativa. *Visionario Digital, 1*(4), 45-61.

44. López, F.-A. (2013). Visibilidad e impacto de los repositorios digitales en acceso abierto. *De bibliotecas y bibliotecarios... Boletín electrónico ABGRA*(5).

45. Macias, J. (2016). MDAP. https://uv-mdap.com/blog/matriz-trazabilidad-requisitos-del-proyecto/

46. MARTÍN LÓPEZ-ASÚNSOLO, A. (2017). Cobertura de código e Informes de Error en Eclipse: más información para evaluar TESTAR.

47. Martínez, A. C., & de Mingo, A. C. (2018). El impacto de la gestión documental en la transparencia de las Administraciones públicas: la transparencia por diseño. *Gestión y Análisis de Políticas Públicas* (19).

48. Méndez, E. (2006). Dublin Core, metadatos y vocabularios. El profesional de la información, *15*(2), 84-86.

49. Merlino-Santesteban, C. (2012). Repositorios digitales institucionales.

50. Metodología de desarrollo para la Actividad productiva de la UCI. . (2015).

51. Molina, J. G., Ortín-Ibáñez, M.-J., Moros, B., Nicolás, J., & Álvarez, J. A. T. (2000). *De los Procesos del Negocio a los Casos de Uso.* Paper presented at the JISBD.

52. Morgado, E. M. M., Ortuño, R. A. C., & Fernández, T. F. (2018). *Ampliación de metadatos educativos en el repositorio Gredos: Proyecto DIRED.* Paper presented at the Ecosistemas del Acceso Abierto.

53. Muñoz, I. (2018). https://developer.mozilla.org/es/docs/Web/XML/Introducci%C3%B3n_a_XML

54. Nogales Flores, J. T., Martín Galán, B., & Arellano Pardo, M. d. C. (2003). Informática, Derecho y Documentación. Experiencias y posibilidades de aplicación de los lenguajes de marcado de texto (SGML, HTML y XML) a los documentos jurídicos.

55. O'Reilly, T. (2006). Qué es Web 2.0. Patrones del diseño y modelos del negocio para la siguiente generación del software. *Boletín de la Sociedad de la Información: Tecnología e Innovación, 37*, 1-31.

56. Perez, A. (2018). https://developer.mozilla.org/es/docs/Web/JavaScript

57. Perez, A. G. (2001). *La gestión de documentos electrónicos como respuesta a las nuevas condiciones del entorno de información*. Habana.

58. Pollo Cattaneo, M. F., Britos, P., Pesado, P. M., & García Martínez, R. (2009). *Metodología para especificación de requisitos en proyectos de explotación de información.* Paper presented at the XI Workshop de Investigadores en Ciencias de la Computación (San Juan, Argentina).

59. Porto, J. P. (208). Definiciones.de. https://definicion.de/html/

60. PowerData. (2002). https://www.powerdata.es/metadatos

61. Pressman, R. S., & Troya, J. M. (1988). *Ingeniería del software*: McGraw Hill.

62. Raffino, E. (2018). Concepto.de. https://concepto.de/lenguaje-de-programacion/

63. Revelles, J. S. (2014). *Alfresco. La gestión de contenidos empresarial*.

64. Risso, V. G. (2012). Aproximación teórica a la relación entre los términos gestión documental, gestión de información y gestión del conocimiento. *Revista española de documentación científica, 35*(4), 531-554.

65. Rodríguez, A. D. (2007). *El concepto de documento electrónico y su validación.* Paper presented at the La validación de los documentos: pasado, presente y futuro: octavas jornadas archivísticas.

66. Rolando Alfredo, S. C. (2012). *El proceso de investigación cientifica*: Editorial Universitaria.

67. Rovitto, F., & Nisim, A. O. (2018). *Tecnología sustentable IoT-Mobile: sistema de monitoreo y diagnóstico de acuarios.* Paper presented at the XXI Concurso de Trabajos Estudiantiles (EST)-JAIIO 47 (CABA, 2018).

68. Sarduy Domínguez, Y., & Urra González, P. (2006). Herramientas para la creación de colecciones digitales. *Acimed, 14*(5), 0-0.

69. Sicilia, M.-Á. (2007). Más allá de los contenidos: compartiendo el diseño de los recursos educativos abiertos. *International Journal of Educational Technology in Higher Education (ETHE), 4*(1).

70. Sommerville, I. (2005). *Ingeniería del software*: Pearson Educación.

71. Sonia Villa Gomez, M. B. G. (2014). *Repositorio Institucional de la UOC: Estudio comparativo y evaluación de su accecibilidad*.

72. Taboada, M. C. (2009). Diagrama de despliegue.

73. Torres Pombert, A. (2006). ¿ Catalogación en el entorno digital?: una breve aproximación a los metadatos. *Acimed, 14*(5), 0-0.

74. UCI. (2019). Retrieved from https://www.uci.cu/investigacion-y-desarrollo/productos/xabal/repxos-30

75. Vargas-Arcila, A. M., Baldassarri, S., & Arciniegas, J. L. (2016). Análisis de Esquemas de Metadatos para la Marcación de Contenidos Educativos. *Formación universitaria, 9*(5), 85-96.

76. ZAPATA, C. M., TAMAYO, P. A., & Arango, F. (2007). Conversión de Esquemas Preconceptuales a diagrama de casos de uso empleando AToM3. *Dyna, 74*(153), 237-251.

Anexos

Anexo 1: Descripción de casos de uso

Tabla 6. CU2. Listar colecciones

Objetivo	El sistema debe permitir al usuario autenticado visualizar todas las colecciones registradas en el sistema a la cuales tiene permiso.	
Actores	Usuario	
Resumen	El usuario selecciona comenzar un nuevo envío y selecciona la colección donde desea incorporar el documento.	
Complejidad	Alta.	
Prioridad	Alta.	
Precondiciones	Debe estar montado el servicio.	
Postcondiciones	Se listaron las colecciones.	
Flujo de eventos		
Flujo básico Listar colecciones.		
	Actor	Sistema
1.	Selecciona comenzar un nuevo envío.	
2.		El sistema realiza una consulta repxos.
3.		Muestra las colecciones registradas en el sistema.
4.	Selecciona una colección.	
5.		Se termina el caso de uso.
Flujos alternos		

3* Opción “cancelar”		
	Actor	Sistema
1.	***Selecciona la opción “Cancelar”.***	
2.		Retorna a la página que le dio origen.
1.		Elimina un documento de la tabla.
Prototipo elemental de interfaz gráfica de usuario		
Fig 2. Prototipo Listar colecciones.		

Tabla 7. CU3. Añadir adjunto

Objetivo	El sistema debe permitir al usuario autenticado adjuntar uno o más ficheros a un envío.
Actores	Usuario.
Resumen	El usuario adiciona documentos a una tabla para luego subirlos al repositorio.
Complejidad	Alta.
Prioridad	Alta.
Precondiciones	El documento que se va a adjuntar debe estar descrito antes en un formulario.
Postcondiciones	El documento esta adjuntado.
Flujo de eventos	
Flujo básico Añadir adjuntos.	

	Actor	Sistema
1.	Selecciona adjuntar documento.	
2.		Adiciona documentos a una tabla.
3.	Selecciona la opción Enviar	
4.		Termina el caso de uso.
Flujos alternos		
1* Opción “Cancelar”		
	Actor	Sistema
3.	***Selecciona la opción “Cancelar”.***	
4.		Retorna a la página que le dio origen.
Prototipo elemental de interfaz gráfica de usuario		
Fig 2. Prototipo Añadir adjunto.		

Tabla 8. CU4. Borrar adjunto

Objetivo	El sistema debe permitir al usuario autenticado borrar el o los adjuntos de un envío.
Actores	Usuario
Resumen	El caso de uso comienza cuando el usuario desea eliminar un documento de la tabla.
Complejidad	Alta.
Prioridad	Alta.

Precondiciones	Para eliminar un documento de la tabla, debe existir al menos un documento.	
Postcondiciones	El documento fue eliminado.	
Flujo de eventos		
Flujo básico Borrar adjunto.		
	Actor	Sistema
1.	Selecciona eliminar.	
2.		Elimina un documento de la tabla.
3.		Se termina el caso de uso.
Flujos alternos		
1* Opción “Cancelar”		
	Actor	Sistema
5.	***Selecciona la opción “Cancelar”.***	
6.		Retorna a la página que le dio origen.
Prototipo elemental de interfaz gráfica de usuario		
Fig 2. Prototipo Borrar adjuntos.		

Tabla 9. CU5. Iniciar flujo de revisión

Objetivo	El sistema automáticamente debe iniciar el flujo de trabajo Revisión y aprobación cuando se registra un envío.
Actores	Usuario
Resumen	El usuario al seleccionar enviar automáticamente se inicia el flujo de revisión.

Complejidad	Alta.
Prioridad	Alta.
Precondiciones	El documento debe estar descrito en un formulario y adjuntado en la tabla.
Postcondiciones	El envío fue revisado.

Flujo de eventos		
Flujo básico Iniciar flujo de revisión.		
	Actor	Sistema
1.	Selecciona enviar	
2.		El sistema automáticamente inicia el flujo de revisión.
3.		Se termina el caso de uso
Flujos alternos		
3* Envío incorrecto		
	Actor	Sistema
1.		Si existe algún error con el envío el sistema no incorpora.
2.		Regresa el envío al usuario para que lo modifique o lo elimine.
Prototipo elemental de interfaz gráfica de usuario		
Fig 2. Prototipo Iniciar flujo de revisión.		

Tabla 10. CU6 Aprobar envío

Objetivo	El sistema debe permitir al revisor asignado aprobar el envío.	
Actores	Revisor	
Resumen	El revisor recibe el envío, revisa el mismo y aprueba.	
Complejidad	Alta.	
Prioridad	Alta.	
Precondiciones	Debe estar iniciado el flujo de revisión.	
Postcondiciones	El envío fue revisado.	
Flujo de eventos		
Flujo básico Aprobar envío.		
	Actor	Sistema
1.	Revisa el envío.	
2.	Aprueba el envío.	
3.		Se termina el caso de uso.
Fig 2. Prototipo Aprobar envío.		

Tabla 11 CU7. Rechazar envío

Objetivo	El sistema debe permitir al revisor asignado rechazar el envío.
Actores	Revisor
Resumen	El revisor recibe el envío, revisa el mismo y rechaza.
Complejidad	Alta.

Prioridad	Alta.	
Precondiciones	Debe estar iniciado el flujo de revisión.	
Postcondiciones	El envío fue revisado.	
Flujo de eventos		
Flujo básico Rechazar envío.		
	Actor	Sistema
1.	Revisa el envío.	
2.	Rechazar el envío.	
3.		Regresa el envío al usuario.
4.		Se termina el caso de uso.

Fig 2. Prototipo rechazar envío.

Tabla 12. CU8. Archivar envío

Objetivo	El sistema una vez aprobado el envío debe archivar los ficheros asociados al mismo en la colección correspondiente
Actores	Revisor
Resumen	El revisor recibe el envío, revisa el mismo y aprueba o rechaza, en caso de aprobar, archiva los ficheros asociados en la colección correspondiente.
Complejidad	Alta.
Prioridad	Alta.
Precondiciones	Debe estar iniciado el flujo de revisión.

Postcondiciones	El envío fue archivado.	
Flujo de eventos		
Flujo básico Archivar envío.		
	Actor	Sistema
1.	Revisa el envío.	
2.	Aprueba el envío.	
3.		Archiva los ficheros asociados en la colección correspondiente.
4.		Se termina el caso de uso.

Fig 2. Prototipo Archivar envío.

Tabla 13. CU9 Notificar nueva revisión

Objetivo	El sistema debe notificar al grupo de revisores mediante un correo electrónico la asignación de la nueva revisión.
Actores	Usuario
Resumen	El usuario al seleccionar enviar automáticamente el sistema envía una notificación al grupo de revisores de la nueva revisión.
Complejidad	Alta.
Prioridad	Alta.
Precondiciones	Las condiciones para revisar el envío deben estar creadas.
Postcondiciones	El grupo de revisión fue notificado.

Flujo de eventos		
Flujo básico Notificar nueva revisión.		
	Actor	Sistema
1.	Selecciona enviar.	
2.		Automáticamente envía una notificación al grupo de revisores.
3.		Se termina el caso de uso.
Prototipo elemental de interfaz gráfica de usuario		
Fig 2. Prototipo Notificar nueva revisión.		

Tabla 14 CU10. Notificar decisión de revisión

Objetivo	El sistema debe notificar al creador del envío mediante un correo electrónico la decisión de la revisión.
Actores	Revisor
Resumen	El revisor al aprobar o rechazar el envío automáticamente se envía una notificación al creador con el resultado de la revisión.
Complejidad	Media.
Prioridad	Media
Precondiciones	El revisor debe haber aprobado o rechazado el envío.
Postcondiciones	El creador del envío fue notificado.
Flujo de eventos	

Flujo básico Notificar decisión de revisión.		
	Actor	Sistema
1.	Aprueba o rechaza el envío.	
2.		Automáticamente envía una notificación al creador del envío con los resultados de la revisión.
3.		Se termina el caso de uso.
Prototipo elemental de interfaz gráfica de usuario		
Fig 2. Prototipo Notificar decisión de revisión.		

Tabla 15. CU11 Mostrar envíos no terminados

Objetivo	El sistema debe permitir al usuario autenticado listar todos los envíos realizados por él que tiene estado no terminado. De ellos se visualizarán los siguientes metadatos: Enviado por, Título y Enviado a (Colección).
Actores	Usuario
Resumen	El usuario selecciona mostrar envíos sin terminar se muestra una tabla con los envíos sin terminar.
Complejidad	Media
Prioridad	Media
Precondiciones	Para mostrar los envíos sin terminar debe existir al menos un envío sin terminar.
Postcondiciones	Los envíos sin terminar son mostrados.

Flujo de eventos		
Flujo básico Mostrar envíos.		
	Actor	Sistema
1.	Selecciona mostrar envíos sin terminar	
2.		El sistema muestra una tabla con los envíos sin terminar registrados en el sistema.
3.		Se termina el caso de uso.
Prototipo elemental de interfaz gráfica de usuario		
Fig 2. Prototipo Mostrar envíos sin terminar.		

Tabla 16. CU12 Mostrar envíos rechazados

Objetivo	El sistema debe permitir al usuario autenticado listar todos los envíos realizados por él que tiene estado rechazado. De ellos se visualizarán los siguientes metadatos: Fecha de publicación, Título y Autores.
Actores	Usuario
Resumen	El usuario selecciona mostrar envíos rechazados se muestra una tabla con los envíos rechazados.
Complejidad	Media
Prioridad	Media
Precondiciones	Para mostrar los envíos rechazados debe existir al menos un envío rechazado registrado en el sistema.

Postcondiciones	Los envíos rechazados son mostrados.	
Flujo de eventos		
Flujo básico Mostrar envíos.		
	Actor	Sistema
1.	Selecciona mostrar envíos rechazados.	
2.		El sistema muestra una tabla con los envíos rechazados en el sistema.
3.		Se termina el caso de uso.
Prototipo elemental de interfaz gráfica de usuario		
Fig 2. Prototipo Mostrar envíos rechazados.		

Tabla 17. CU13 Mostrar envíos aceptados

Objetivo	El sistema debe permitir al usuario autenticado listar todos los envíos realizados por él que tiene estado aceptado. De ellos se visualizarán los siguientes metadatos: Fecha de publicación, Título y Autores.
Actores	Usuario
Resumen	El usuario selecciona mostrar envíos aceptados se muestra una tabla con los envíos aceptados.
Complejidad	Media
Prioridad	Media

Precondiciones	Para mostrar los envíos aceptados debe existir al menos un envío aceptado registrado en el sistema.	
Postcondiciones	Los envíos aceptados son mostrados.	
Flujo de eventos		
Flujo básico Mostrar envíos.		
	Actor	Sistema
1.	Selecciona mostrar envíos aceptados.	
2.		El sistema muestra una tabla con los envíos aceptados registrados en el sistema.
3.		Se termina el caso de uso.
Prototipo elemental de interfaz gráfica de usuario		
Fig 2. Prototipo Mostrar envíos aceptados.		

Anexo 2 Casos de pruebas

Tabla 18. CP2. Listar colecciones

SC1 Listar colecciones			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Listar colecciones	El escenario de prueba permite al usuario autenticado ver todas las colecciones registradas en el sistema	El sistema muestra un select con las colecciones registradas en el sistema	1-El usuario selecciona comenzar un nuevo envío 2- Se muestran las colecciones

Tabla 19. CP3 Añadir adjunto

SC1 Añadir adjunto			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Añadir adjunto	El escenario de prueba permite al usuario autenticado añadir un documento	El sistema adiciona el documento a una tabla	1-El usuario selecciona adicionar documento 2- Se adiciona el documento a la tabla

Tabla 20. CP4 Borrar adjunto

SC1 Borrar adjunto			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Borrar adjunto	El escenario de prueba permite al usuario autenticado borrar un adjunto de la tabla	El sistema elimina un adjunto de la tabla	1-El usuario selecciona eliminar 2- Se elimina el elemento

Tabla 21CP5. Iniciar flujo de revisión

SC1 Iniciar flujo de revisión			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Iniciar flujo de revisión	El escenario de prueba permite al usuario autenticado iniciar el flujo de revisión	El sistema inicia el flujo de revisión	1-El usuario selecciona finalizar 2- Inicia el flujo de revisión

Tabla 22 CP6 Aprobar envío

SC1 Aprobar envío			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Aprobar envío	El escenario de prueba permite al usuario usuario revisor aprobar un envío	El sistema guarda el envío	1-El usuario revisor aprueba el envío

Tabla 23 CP7 Rechazar envío

SC1 Rechazar envío			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Rechazar envío	El escenario de prueba permite al usuario revisor rechazar el envío	El sistema regresa el envío a quien lo realizó	1-El usuario revisor rechaza el envío

Tabla 24 CP8 Archivar envío

SC1 Archivar envío			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Archivar envío	El escenario de prueba permite archivar un envío	El sistema archiva el envío	

Tabla 25 CP9 Notificar nueva revisión

SC1 Notificar nueva revisión

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Notificar nueva revisión	El escenario de prueba permite notificar al usuario revisor de la nueva tarea	Envia una notificación al usuario revisor	

Tabla 26 CP10 Notificar decisión de revisión

SC1 Notificar decisión de revisión			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Notificar decisión de revisión	El escenario de prueba permite al usuario autenticado ver la decisión tomada por el revisor	El sistema envía una notificación al usuario	

Tabla 27 CP11 Mostrar envíos no terminados

SC1 Mostrar envíos no terminados			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Mostrar envíos no terminados	El escenario de prueba permite al usuario autenticado ver los envíos con estado no terminado	El sistema muestra una tabla con los envíos en estado no terminado	1-El usuario selecciona mostrar envíos sin terminar 2- Se muestra la tabla con los envíos sin terminar

Tabla 28 CP12 Mostrar envíos rechazados

SC1 Mostrar envíos rechazados			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Mostrar envíos rechazados	El escenario de prueba permite al usuario autenticado ver todos los envíos en estado rechazado	El sistema muestra una tabla con los envíos en estado rachazado	1-El usuario selecciona ver envíos rechazados 2- Se muestra una tabla con los envios rechazados

Tabla 29 CP13 Mostrar envíos aceptados

SC1 Mostrar envíos aceptados			
Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1 Mostrar envíos aceptados	El escenario de prueba permite al usuario autenticado ver los envios en estado aceptado	El sistema muestra una tabla con los envíos en estado aceptados	1-El usuario selecciona ver envíos aceptados 2- Se muestra una tabla con los envíos aceptados