Universidad de las Ciencias Informáticas

Facultad 2

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas.

Módulo de Administración de bases de datos J-ISIS para el sistema ABCD v3.0

Autor:

Wilmer Garcia Asin

Tutores:

Ing. Rafael Linares Torres

Ing. Basilio Puentes Rodríguez

La Habana, 21 de junio de 2018

Declaración de autoría

Declaro ser el único autor del trabajo de diploma “Módulo de Administración de bases de datos J-ISIS para el sistema ABCD v3.0”, concedo a la Universidad de las Ciencias Informáticas y en especial al Centro de Informatización de la Gestión Documental a hacer uso del mismo en su beneficio.

Para que conste firmo el presente documento a los ___ días del mes de _____ del año _____.

Wilmer Garcia Asin

_____

Firma del autor

Ing. Rafael Linares Torres

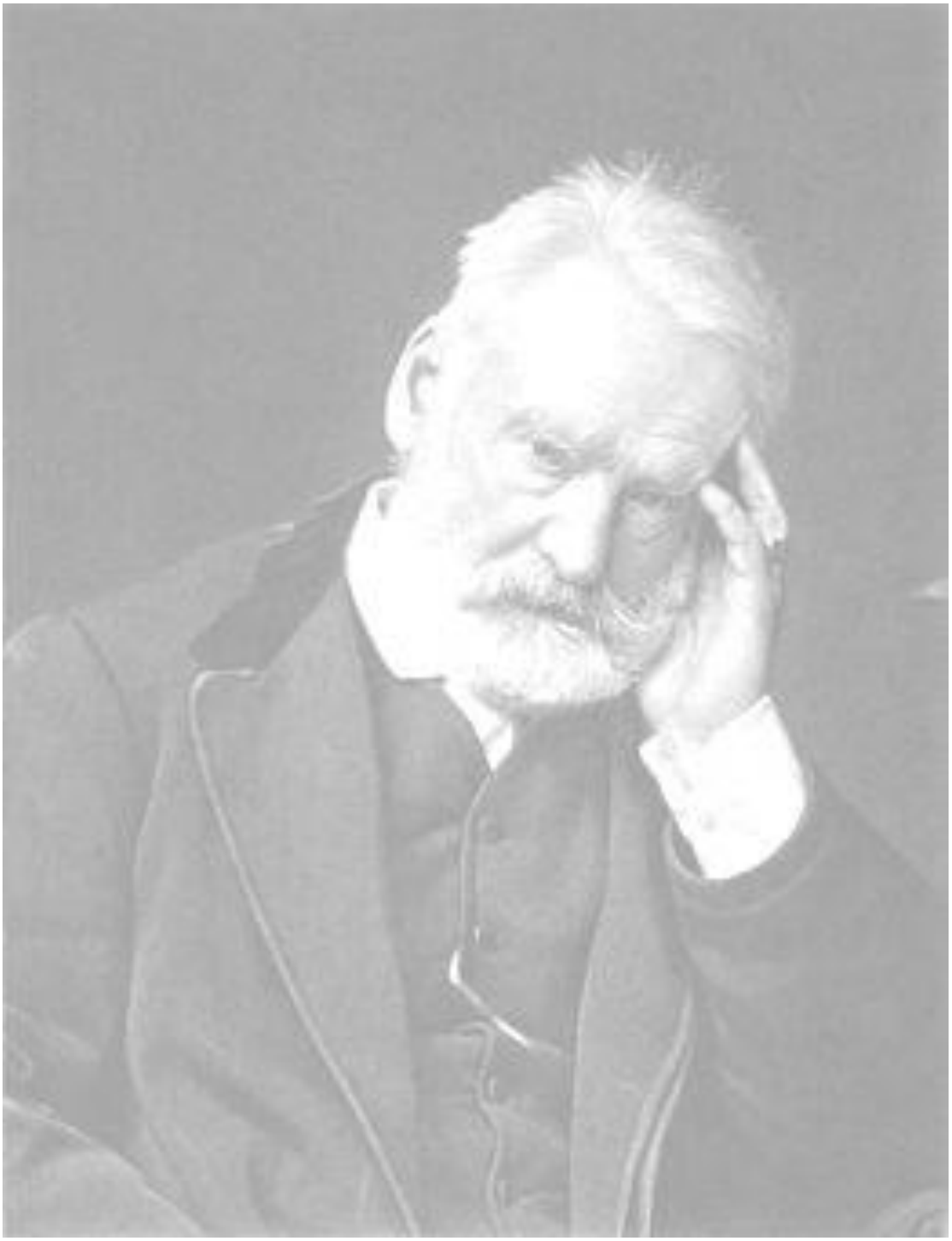
_____

Firma del tutor

Ing. Basilio Puentes Rodríguez

_____

Firma del tutor



El futuro tiene muchos nombres. Para los débiles es lo inalcanzable. Para los temerosos, lo desconocido. Para los valientes es la oportunidad.

Víctor Hugo

Dedicatoria

A toda mi familia que por 5 años estuvieron a mi lado.

A mis padres que han sido capaces de ser magos para apoyarme todo este tiempo, dando lo mejor de ellos. Los AMO más que a nada en este mundo, les debo toda mi vida y aun así es muy poco para pagarles.

A mi hermana por la preocupación y amor que no para hacia mí, te quiero bonita.

A mi tío o primo o como quieran llamarlo, a ti Aramis, que no me dejaste solo y estuviste siempre al tanto haciendo el papel de mis padres, gracias de veras.

A todos lo que confiaron e mi para hacer realidad el sueño de mi vida y lograr este mi mayor deseo. MUCHAS GRACIAS.

Resumen

ABCD v3.0 es un sistema de Automatización de Bibliotecas y Centros de Documentación que se encarga de informatizar los procesos relacionados con las distintas áreas que puedan existir en una biblioteca de forma general, este pertenece al Centro de Informatización de la Gestión Documental (CIGED) de la Universidad de las Ciencias Informáticas (UCI), el cual posee varios proyectos encaminados a la informatización de los procesos inherentes a la gestión de la documentación.

La administración de bases de datos J-ISIS para ABCD v3.0 se realiza mediante gestor de base de datos “J-ISIS Suite”. Este proceso no es óptimo, ya que una vez que el sistema está desplegado para administrar las bases de datos de registros bibliográficos con las que trabaja es necesario interrumpir la disponibilidad del sistema, además se necesita dominar conocimientos técnicos con el gestor de bases de datos J-ISIS, los desarrolladores del sistema no siempre están disponible y el traslado también implica un problema.

Debido a esta situación se decidió desarrollar un módulo de Administración de bases de datos J-ISIS para el sistema ABCD v3.0, y de esta forma contribuir a la gestión de las bases de datos J-ISIS en este. Este módulo permite administrar todo lo relacionado a las bases de datos de los registros bibliográficos de la biblioteca directamente desde el sistema ABCD v3.0.

Palabras claves: administración, base de datos, J-ISIS, módulo, registros bibliográficos.

Abstract

ABCD v3.0 is a system of automation of libraries and documentation centers that is in charge of computerizing the processes related to the different areas that may exist in a library in general, this belongs to the Computing Centre of the Management documentary (CIGED) of the University of Informatics Sciences (UCI), which has several projects aimed at the computerization of processes inherent to the management of the documentation.

The management of J-ISIS for ABCD database v3.0 is performed through "J-ISIS Suite" database manager. This process is not optimal, since once the system is deployed to manage the databases of bibliographic records that works is necessary to discontinue the availability of the system, is also needed to master technical skills with the Manager of J-ISIS databases, the developers of the system are not always available and the transfer also involves a problem.

Due to this situation, it was decided to develop J-ISIS databases for the ABCD system administration module v3.0, and in this way contribute to the management of J-ISIS databases in this. This module allows you to manage everything related to the databases of bibliographic records of the library directly from the ABCD system v3.0.

Keywords: management, database, J-ISIS, module, bibliographic records.

Índice

Introducción 1

Capítulo 1. Fundamentación teórica de la investigación 4

1.1 Sistema Integrado de gestión Bibliotecaria 4

1.1.1 La familia del software ISIS 4

1.2 Sistemas de gestión bibliotecaria 7

1.2.1 ABCD v1.0 7

1.2.2 PHPMyBibli 8

1.2.3 OpenBiblio 10

1.3 Metodología de desarrollo de software y lenguajes 11

1.3.1 Metodología 11

1.3.2 Lenguaje de modelado UML versión 2.0 12

1.3.3 Lenguaje de desarrollo Java 13

1.4 Herramientas utilizadas 13

1.4.1 Visual Paradigm versión 8.0 13

1.4.2 Eclipse versión Kepler Release 4.3 14

1.4.3 Gestores de Bases de Datos 14

1.5 Tecnologías 15

1.5.1 Eclipse Virgo versión 3.6.3 15

1.5.2 RAP versión 3.0.5 15

1.5.3 OSGi Equinox versión 3.8.2 16

1.5.4 Spring Dynamic Modules versión 3.1.0 17

1.6 Conclusiones parciales 17

Capítulo 2. Propuesta de solución 18

2.1 Descripción de la propuesta de la solución 18

2.2 Modelo de Dominio 18
2.3 Especificación de los requisitos funcionales 19
2.3.1 Requisitos funcionales 19
2.3.2 Requisitos no funcionales 22
2.4 Modelo de casos de uso del sistema 23
2.4.1 Actores del sistema 24
2.4.2 Descripción de los casos de uso 24
2.4.3 Arquitectura utilizada 36
2.5 Patrones de diseño GRASP 38
2.6 Diagrama de clases del diseño utilizando estereotipos web 42
2.7 Conclusiones parciales 45
Capítulo 3. Implementación y prueba de resultados 46
3.1 Modelo de implementación del módulo 46
3.2 Estándares de codificación 46
3.3 Diagrama de despliegue 46
3.4 Pruebas al módulo 47
3.5 Conclusiones parciales 56
Conclusiones 57
Recomendaciones 58
Referencias Bibliográficas 59
Bibliografía 62

Introducción

Los adelantos tecnológicos en el sector de la informática, han influido de manera decisiva en el desarrollo del proceso docente educativo, y lo han renovado de manera sustancial en las diversas aristas que lo integran. El advenimiento de las tecnologías de la información, ha hecho posible que las bibliotecas incorporen herramientas para satisfacer necesidades de los usuarios, mediante la introducción de servicios automatizados como lo son la circulación, administración de base de datos, entre otros que perfeccionan los procesos de gestión de materiales bibliográficos en las instituciones, facilitando la accesibilidad a la información almacenada.

La Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura (UNESCO) recomienda la utilización de bases de datos documentales pertenecientes a la familia ISIS para el almacenamiento de información bibliográfica. En 2004, la UNESCO lanzó un programa transorganizacional en gestión documental (*records management*) con el fin de garantizar la responsabilidad y la transparencia en todos los niveles del Secretariado. Este lanzamiento se alcanzó gracias a la aplicación de normas, reglas y prácticas relacionadas con la creación, la organización y la gestión de documentos, así como su transferencia al archivo para su conservación permanente y su utilización por la organización donde esté el sistema. La gestión documental trata el conjunto de documentos electrónicos, audiovisuales o en papel creados por la UNESCO, en la sede y fuera de ella, en los institutos y en los centros. Se ha priorizado la introducción de la gestión de documentos electrónicos en la organización. Entonces hoy día la mayoría de las comunicaciones y de los documentos se producen electrónicamente y se les reconoce en general un valor legal, es indispensable conservar este capital documental en toda su integridad y autenticidad y otorgarle su valor testimonial(UNESCO 2008).

Como parte del programa transorganizacional en gestión documental de la UNESCO la Universidad de las Ciencias Informáticas hace sus aportes. Esta universidad se crea en el año 2002 con la misión de formar profesionales comprometidos con su Patria y altamente calificados en la rama de la informática, para producir aplicaciones y servicios informáticos a partir de la vinculación con la docencia, producción e investigación como modelo de formación, y servir de soporte a la industria cubana de la informática.

En la Universidad de las Ciencias Informáticas se encuentra el Centro de Informatización de la Gestión Documental (CIGED), el cual posee varios proyectos encaminados a la informatización de los procesos inherentes a la gestión de la documentación. Uno de estos proyectos es el de Automatización de Bibliotecas y Centros de Documentación (ABCD) que se encarga de desarrollar un sistema para

informatizar los procesos relacionados con las distintas áreas que puedan existir en una biblioteca de forma general.

El equipo de desarrollo del sistema ABCD v3.0 seleccionó para la gestión de bases de datos, J-ISIS para el almacenamiento de los registros bibliográficos. En las bibliotecas se trabaja con las bases de datos J-ISIS las cuales almacenan los registros bibliográficos, para hacer cambios en estos o agregar alguno, es necesario tener conocimientos técnicos para trabajar con el gestor de bases de datos, por lo que las bibliotecas acuden al grupo de desarrollo de ABCD. Los programadores que trabajan en el sistema en ocasiones no cuentan con la disponibilidad necesaria o el tiempo para ocuparse de estas necesidades, además influyen factores como traslado de una entidad a otra e inclusión de personas externas al proyecto o la biblioteca, para autorizar el acceso a los servidores donde está desplegado el sistema, haciendo el proceso lento, engorroso y complejo.

La situación problemática descrita anteriormente conduce al **problema de investigación:** *¿Cómo gestionar las bases de datos J-ISIS en el sistema ABCD v3.0?22*

El **objeto de estudio** sobre el cual se centra el problema es: *La administración de bases de datos de Sistemas de Gestión Bibliotecaria*, enmarcado en el **campo de acción**: *La administración de bases de datos de Sistemas de Gestión Bibliotecaria usando J-ISIS*.

A partir de lo anterior se determinó como **objetivo general:** *Desarrollar un módulo de Administración para el sistema ABCD v3.0, que contribuya a la gestión de bases de datos J-ISIS.*

Tareas de investigación:

1) Elaboración del marco teórico metodológico referente a los conceptos asociados a la administración de bases de datos de Sistemas de Gestión Bibliotecaria.
2) Caracterización de la administración de bases de datos con J-ISIS.
3) Diseño del módulo de Administración de bases de datos para el sistema ABCD v3.0.
4) Desarrollo del módulo de Administración de bases de datos para el sistema ABCD v3.0 usando J-ISIS.
5) Integración del módulo de Administración de bases de datos J-ISIS al sistema ABCD v3.0.
6) Validar la propuesta de solución a partir de los métodos de pruebas definidos en la investigación.

Los **Métodos de investigación** utilizados para darle solución al objetivo propuesto fueron:

Métodos teóricos:

- **Analítico – Sintético** para descomponer el problema de investigación en elementos, profundizar en su estudio y luego sintetizarlos en la solución propuesta.
- **Histórico – Lógico** se utilizó para analizar el surgimiento y evolución de los sistemas bibliotecarios y así facilitar la comprensión del objeto y campo de estudio.

Métodos empíricos:

- **Análisis documental:** en la revisión de la literatura especializada para extraer la información necesaria que permitió realizar el proceso de investigación.
- **Análisis comparativo:** para detectar similitudes, diferencias e insuficiencias en cuanto a la interpretación que brindan las herramientas existentes para modelar los procesos.
- **Observación:** para adquirir conocimiento del campo de acción a través de la investigación realizada sobre las herramientas.

Con el desarrollo de la investigación se espera el siguiente **aporte**:

1. Un módulo informático que permite la Administración de bases de datos J-ISIS para el sistema ABCD v3.0.

La estructura del documento es la siguiente: introducción, 3 capítulos, conclusiones, bibliografías y los anexos. El contenido de cada capítulo se describe a continuación:

Capítulo 1: Fundamentación teórica de la investigación, dedicado a los fundamentos teóricos de la investigación. Se analiza una variedad de sistemas bibliotecarios presentes en la gestión bibliotecaria. Se introduce y aborda la administración de bases de datos en sistemas bibliotecarios como alternativa de solución.

Capítulo 2: Propuesta de Solución, describe el módulo desarrollado para la Administración de bases de datos J-ISIS en el sistema ABCD 3.0.

Capítulo 3: Implementación y prueba de resultados, se formaliza la implementación de la propuesta de solución y contiene la evaluación de los resultados alcanzados en la investigación.

Capítulo 1. Fundamentación teórica de la investigación

En el presente capítulo se definen los conceptos fundamentales relacionados con los sistemas de gestión bibliotecaria, con el fin de establecer la base teórica necesaria para el desarrollo de la investigación. Además, se realiza un análisis del estado del arte utilizando los métodos científicos de investigación, con el fin de analizar algunos de los sistemas integrados de gestión bibliotecaria existentes a nivel mundial, para determinar características particulares de cada uno de ellos. Se caracterizan también las herramientas, tecnologías y la metodología de desarrollo de software que se utilizarán para la implementación del sistema.

1.1 Sistema Integrado de gestión Bibliotecaria

Según la comunidad catalográfica, se concibe el **registro bibliográfico** como “una agregación de datos que se asocian con las entidades descritas en los catálogos bibliotecarios y en las bibliografías nacionales. Dichos datos describen, representan y posibilitan la organización de materiales textuales, musicales, audiovisuales, cartográficos, gráficos y tridimensionales” (Centro Nacional de Información de Ciencias Médicas. y Ríos Hilario 2005). El autor de esta investigación entiende por registro bibliográfico los datos que permiten organizar de forma eficiente los medios que se encuentran en la institución, que pueden ser consumidos por los usuarios de la biblioteca.

El licenciado en Literatura Española, Luis Ángel García Melero, define un **Sistema Integrado de Gestión Bibliotecaria (SIGB)** como "un conjunto organizado de recursos que utilizan dispositivos y programas informáticos, adecuados a la naturaleza de los datos que deben procesar, para realizar procesos y facilitar los servicios que permiten alcanzar los objetivos de la biblioteca: almacenar de forma organizada el conocimiento humano contenido en todo tipo de materiales bibliográficos, para satisfacer la necesidades informativas, recreativas y de investigación de los usuarios"(Arriola Navarrete y Butrón Yáñez 2008).

Teniendo en cuenta el análisis realizado, el autor de este trabajo asume que un SIGB es un programa informático destinado a administrar todas las funciones realizadas por una biblioteca. Esto se logra con la implementación de servicios que posibiliten a los usuarios el fácil acceso al conocimiento almacenado en forma de información en la institución.

1.1.1 La familia del software ISIS

ISIS Script es un lenguaje de creación de comandos desarrollado por el Centro Latinoamericano y del Caribe de Información en Ciencias de la Salud (BIREME) para diseñar funciones más potentes para el servidor web ISIS “WWW”, para la creación de páginas con elementos de bases de datos ISIS. ISIS

Script fue en realidad uno de los elementos principales en la escalada de WWWISIS a 'WXIS', siendo este último el servidor web subyacente para ABCD.

Las características comunes de **la familia ISIS** tienen que ver con la manera como la información (textual) es almacenada y manejada en campos repetibles de longitud variable, con la posibilidad de subdividir a éstos en sub-campos. Los campos en realidad están formados por parejas de identificador de campo (una "etiqueta") combinado con un valor de campo (un texto, o en la nueva generación ISIS, cualquier objeto, como p.ej. "grandes objetos binarios" o blobs, "*binary large objects*", en inglés)(Luján 2012).

Además de las características tecnológicas comunes, los miembros de la familia ISIS comparten también **características sociales**(Smet y Spinak 2009)**:**

- ✓ Son utilizados principalmente en países en desarrollo, o "el Sur" con, p.ej., una fuerte presencia en América Latina, pero también, - y en una medida difícil de estimar - en toda clase de pequeños centros de información en África y Asia, a menudo en situación precaria y sin conexión a Internet.
- ✓ Son promovidos por muchos miembros y proyectos de la Organización de Naciones Unidas (ONU), por supuesto en primer lugar por los entornos UNESCO, pero - como es el caso de BIREME - también la Organización Mundial de la Salud (OMS) y FAO (los sistemas AGRIS y ASFISIS de FAO son ejemplos válidos, como así también el origen del sistema bibliotecario WEBLIS). Los programas de las Naciones Unidas FIPA y "Sociedad del Conocimiento" no deberían subestimar cuánto impacto real proviene de las herramientas de información promovidas por UNESCO, como ISIS, IDAMS y Greenstone, a veces indicando inclusive que dicho impacto puede ser el reverso de apoyo publicitario o financiero.

Se podría resumir la historia de la familia ISIS afirmando que tiene hasta el 2009 4 generaciones: (Smet y Spinak 2009):

- ✓ La primera generación: CDS/ISIS y Micro-ISIS.
- ✓ La segunda generación: interfaces enriquecidas ISIS/Pascal, herramientas CISIS.
- ✓ La tercera generación: bases de datos múltiples, gráficas, multimedia: WinISIS, ISISDLL.
- ✓ La cuarta generación: versiones adaptadas a WWW (wwwisis, isis3w, openisis).

Características destacadas de La quinta generación – 2008(Smet y Spinak 2009)**:**

- ✓ UNESCO desarrolla "J-ISIS", una interfaz gráfica totalmente nueva usando no sólo tecnología Java sino también la base de datos Berkeley incrustada para la capa de almacenamiento. Este

es un proyecto desarrollado totalmente hacia el concepto Software Libre y de Código Abierto (siglas en inglés FOSS).

- ✓ NBP o *"Network Based Platform"* es la nueva tecnología ISIS con las siguientes características principales:
  - Arquitectura flexible en la cual las "células ISIS" se comunican mediante protocolos conocidos con varias plataformas e interfaces; las células ISIS también permitirán utilizar diferentes modelos de almacenamiento, en tanto y en cuanto éstos estarán contenidos en las células, pero se comportarán de la misma manera estandarizada hacia la tecnología externa empleada.
  - Las bases de datos ISIS no tendrán en el futuro las anteriores limitaciones en cuanto a tamaño de base de datos, registro o campo.
  - Las bases de datos ISIS serán compatibles con UNICODE.
  - La indización se llevará a cabo mediante otras aplicaciones de indización FOSS a texto completo, como Lucene (de la *Apache Software Foundation*).

ISIS está siendo utilizado por aproximadamente diez mil usuarios, la mayoría en países en desarrollo donde es promovido por la UNESCO y BIREME para la mayor parte de América Latina. En esta región está fuertemente representado en bibliotecas y centros de documentación; en África y el sudeste asiático existe un número no estimado pero alto de usuarios, muchos de ellos a menudo no conectados a Internet y que por lo tanto todavía utilizan tecnología antigua y con habilidades computacionales relativamente bajas. Este escenario crea un desafío especial para los que puedan brindar soporte a la comunidad de usuarios.

Durante el 3er. Congreso Mundial sobre ISIS en Río de Janeiro, Brasil, septiembre 2008 la comunidad de usuarios decidió tornar a ISIS completamente "FOSS" y coordinarlo por un Comité Internacional de Coordinación en ISIS (ICCI).

Resumiendo, podría decirse que ISIS combina principios robustos de bases de datos textuales, una fuerte tradición y una comunidad de usuarios extendida a nivel mundial pero insuficientemente coordinada, sin embargo, con un moderno estado de la situación en cuanto a desarrollo tecnológico(Smet y Spinak 2009).

1.2 Sistemas de gestión bibliotecaria

Un **sistema de información** se puede definir técnicamente como un conjunto de componentes relacionados que recolectan o recuperan, procesan, almacenan y distribuyen información para apoyar la toma de decisiones y el control en una organización.

Hay tres actividades en un sistema de información de generación de informaciones que las organizaciones necesitan para tomar decisiones, controlar operaciones, analizar problemas y crear nuevos productos o servicios. Estas actividades son:

- Entrada: captura o recolecta datos en bruto tanto del interior de la organización como de su entorno externo.
- Procesamiento: convierte esa entrada de datos en una forma más significativa.
- Salida: transfiere la información procesada a los usuarios o actividades que las utilizarán.

Un **Sistema de Gestión de Bases de Datos (SGBD)** consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a los mismos. Esta definición es prácticamente idéntica a la que se dio anteriormente de Sistema de Información, de hecho, normalmente en el núcleo de un sistema de información se sitúa un SGBD. Un SGBD permite el almacenamiento, manipulación y consulta de datos pertenecientes a una base de datos organizada en uno o varios ficheros. Existen las bases de datos relacionales y no relacionales, esta última consiste en un conjunto de registros donde no se establecen relaciones entre estos(Sarria 2006).

Con el objetivo de lograr un mejor entendimiento de la administración de base de datos en los SIGB, se realizó un análisis del módulo de administración de base de datos en varios sistemas. A continuación, se muestra la relación de las funcionalidades de este módulo en cada uno de ellos:

1.2.1 ABCD v1.0

El módulo central de ABCD v1.0 comprende módulos para administración de bases de datos (creación de bases de datos, edición de estructuras de bases de datos, utilitarios para bases de datos) catalogación, adquisición, circulación/préstamos y estadísticas. Se está preparando un módulo de gestión de tesauros como parte del módulo de catalogación para una base de datos específica con estructura de tesauro, y con control de consistencia de los niveles jerárquicos. Como parte de este 'módulo central' también deseamos mencionar los servicios de importación y exportación, impresión, y herramientas de bases de datos, como bloquear/desbloquear y “cambios globales” a los campos de los registros (Smet y Spinak 2009).

Este sistema cuenta con un menú principal de gestión de bases de datos, el cual presenta las siguientes opciones presentes en la Figura 1 (Smet y Spinak 2009):

Figura 1: Funcionalidades del módulo de administración.

Hay que señalar que tiene un módulo de circulación, conocido como módulo de circulación avanzada o EmpWeb, este aporta una serie de mejoras como la posibilidad de hacer reservas o la función “mi biblioteca”, mediante la cual los usuarios pueden comprobar en línea la información de su estado (información personal, préstamos); además, también ofrece una serie de mejoras a nivel técnico, relacionadas con la gestión de las bases de datos, uso de servidores, posibilidad de establecer distintas políticas de préstamo. Sin embargo, el uso de este módulo avanzado supone incorporar tecnologías. Necesita un sistema Windows o Linux que además cuente con el software de servidor Apache, el lenguaje PHP y el módulo YAZ (herramienta que permite el uso del protocolo Z39.50) y solamente para EmpWeb, una base de datos de tipo SQL (preferentemente MySQL) y el entorno de ejecución de Java.

Este sistema utiliza tecnología ISIS para la administración de bases de datos y la gestiona con CISIS, que es un software desarrollado por BIREME para manejar bases de datos ISIS desde líneas de comandos en UNIX/Linux o DOS/Windows(Coral y Alberto 2011).

1.2.2 PHPMyBibli

Es un sistema integrado de automatización de bibliotecas de código abierto. Bajo la licencia GNU GPL (Licencia General Pública) diseñada para trabajar en los sistemas operativos Windows y Linux. Ofrece varios módulos como: Adquisiciones, Circulación, Catalogación, Administración, Catálogo de Acceso Público en Línea (OPAC) y Diseminación Selectiva de Información (DSI). Además de monografías,

permite gestionar publicaciones periódicas y posee control de autoridades. Cumple con estándares como el MARC21 y permite la búsqueda de registros bibliográficos mediante el protocolo Z39.50. Además, la importación de registros es en formato MARC(Coral y Alberto 2011).

PHPMyBibli es un software configurable, que puede adecuarse a diversas necesidades, tanto por su naturaleza de software libre, como por la posibilidad de establecer parámetros específicos para su uso. Es un sistema multiplataforma. El módulo Adquisición no se visualiza por defecto cuando se instala el programa, debe activarse desde el módulo de Administración en el cual además se deben hacer una serie de configuraciones y definiciones.

El módulo encargado de la Administración de bases de datos realiza las siguientes tareas(Coral y Alberto 2011):

- Permite el establecimiento de parámetros del sistema.
- Creación de usuarios del sistema.
- Control total de todos los módulos.
- Herramienta de exportación e importación de registros.
- Configuración del protocolo Z/39.50.
- Administración del OPAC.
- Actualización de las bases de datos y nuevas versiones del sistema.
- Control de calendario de atención a la biblioteca.
- Gestión de presupuesto.
- Generación de copias de seguridad de los registros.

Este sistema se basa en la arquitectura cliente-servidor, y utiliza las siguientes características(Coral y Alberto 2011):

- Se debe contar con el lenguaje de programación PHP, con la versión 4.3 como mínimo, aunque es conveniente utilizar PHP 5.
- Un software de servidor, como puede ser el caso de Apache, aunque en principio no habría ningún problema en el caso de utilizar otro software distinto.
- Una base de datos SQL, se utiliza mucho MySQL en al menos, su versión 4.1.
- En cuanto al sistema operativo, PMB es multiplataforma y funciona en sistemas como Linux, Windows y MacOS.

1.2.3 OpenBiblio

OpenBiblio permite administrar la configuración completa del sistema de gestión, en relación a las políticas de préstamos, los tipos de colecciones, materiales, usuarios que posibilitan un sencillo uso de la aplicación. Pero también tiene opciones de importación de registros bibliográficos mediante formato USMARC.

OpenBiblio tiene su origen a principios de 2002, siendo el programador Dave Stevens el creador del proyecto. El objetivo de OpenBiblio es convertirse en un sistema fácil de usar, bien documentado, fácil de instalar y concebido para poseer las prestaciones requeridas para la mayor parte de bibliotecas escolares y públicas, que es el tipo de unidades de información a los que se dirige este proyecto OpenBiblio(Coral y Alberto 2011).

Hay que indicar que este proyecto parecía estar abandonado, pues la versión 0.6.1 se publicó en mayo de 2008 y desde entonces no habían aparecido nuevas iteraciones del producto; sin embargo, en marzo de 2012 se publicó la versión 0.7.1, la cual actualiza el programa para que pueda utilizar las versiones más recientes de MySQL y de PHP, corrige algunos errores y añade algunas prestaciones nuevas, como las opciones de renovar todos los documentos que un lector tiene en préstamo, modificar fechas de devolución, circulación off-line o registro histórico de préstamos.

Es válido mencionar que hubo una iniciativa llamada EspaBiblio, cuyo responsable era Jorge Lara Cravero, y que se trataba de una versión de OpenBiblio traducida al castellano. Para su instalación y uso requiere un entorno basado en Linux o Windows como sistema operativo, con software de servidor, lenguaje PHP y base de datos MySQL. Como cliente utiliza una interfaz de tipo web, de modo que basta con tener acceso a un navegador de Internet. Cuenta con los siguientes módulos: OPAC, catalogación (compatible con MARC 21), circulación, administración del sistema e informes(Coral y Alberto 2011).

	ABCD V1.0	**PHPMYBIBLI**	**OPENBIBLIO**	**ABCD V3.0**
LENGUAJE DE PROGRAMACIÓN	PHP	PHP	PHP	Java
GESTIÓN DE BASES DE DATOS REGISTROS BIBLIOGRÁFICOS	CISIS	MySQL	MySQL	J-ISIS
PLATAFORMA	Multiplataforma	Multiplataforma	Multiplataforma	Multiplataforma

Tabla1: Comparación entre los sistemas estudiados.

Fuente: Elaboración propia.

Los sistemas bibliotecarios estudiados cuentan con módulos de administración de bases de datos de registros bibliográficos, la tecnología que usan para la gestión de bases de datos en estos son distintas a las que usa el sistema ABCD v3.0, además el lenguaje de programación con que se implementaron es PHP, lo cual no es recomendable la utilización de estos para el sistema ABCD v3.0 por la diferencia de tecnologías, luego es necesario implementar el módulo de Administración de bases de datos J-ISIS para el sistema ABCD v3.0.

A continuación, se describe el ambiente de desarrollo de ABCD versión 3.0, cuya selección de metodología, tecnologías y herramientas quedan fuera del alcance de la investigación ya que fueron seleccionadas por el equipo de arquitectura del proyecto al que tributa la presente investigación.

1.3 Metodología de desarrollo de software y lenguajes

Metodología de desarrollo de software se refiere al entorno utilizado para estructurar, planificar y controlar el proceso de desarrollo de un sistema de información.

Un Lenguaje de Programación es un conjunto de reglas, notaciones, símbolos y/o caracteres que permiten a un programador expresar el procesamiento de datos y sus estructuras en la computadora. Cada lenguaje posee sus propias sintaxis. En el presente epígrafe se caracteriza la metodología de desarrollo de software y lenguajes de desarrollo utilizados (Barzanallana 2006).

1.3.1 Metodología

Proceso Unificado Ágil de Scott Ambler (AUP) es una versión simplificada del Proceso Racional Unificado (RUP). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. El AUP aplica técnicas ágiles incluyendo:

- Desarrollo Dirigido por Pruebas (*test driven development*).
- Modelado ágil.
- Gestión de cambios ágil.
- Refactorización de Base de Datos para mejorar la productividad.

Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva.

Para los proyectos en la UCI se utiliza AUP modificada para esta universidad, con las siguientes etapas:

- 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.
- 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.
- Cierre: En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

AUP propone 7 disciplinas (Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyecto y Entorno), se decide para el ciclo de vida de los proyectos de la UCI tener 7 disciplinas también, pero a un nivel más atómico que el definido en AUP. Los flujos de trabajos: Modelado de negocio, Requisitos y Análisis y diseño en AUP están unidos en la disciplina Modelo, en la variación para la UCI se consideran a cada uno de ellos disciplinas. Se mantiene la disciplina Implementación, en el caso de Prueba se desagrega en 3 disciplinas: Pruebas Internas, de Liberación y Aceptación. Integración de modelos de madurez de capacidades o Capability Maturity Model Integration (CMMI) es un modelo para la mejora y evaluación de procesos para el desarrollo, mantenimiento y operación de sistemas de software. Las restantes 3 disciplinas de AUP asociadas a la parte de gestión para la variación UCI se cubren con las áreas de procesos que define CMMI-DEV v1.3 para el nivel 2, serían CM (Gestión de la configuración), PP (Planeación de proyecto) y PMC (Monitoreo y control de proyecto)(Rodríguez 2014).

La metodología propone cuatro escenarios para modelar el sistema en los proyectos, se escoge el escenario 2 para el desarrollo del módulo de Administración de bases de datos J-ISIS por ser el utilizado en el sistema ABCD v3.0, para proyectos que modelen el negocio con el Modelo Conceptual (MC) solo pueden modelar el sistema con Casos de Uso del Sistema (CUS). MC=CUS.

1.3.2 Lenguaje de modelado UML versión 2.0

El Lenguaje de Modelado Unificado (UML) permitió especificar, visualizar, construir y documentar artefactos del sistema de software, así como del modelado del negocio. UML se ha convertido en la notación visual estándar de facto y de iure para el modelado orientado a objetos. Algunas propiedades de UML como lenguaje de modelado estándar son(González Cornejo 2008):

- Modela estructuras complejas.
- Las estructuras más importantes que soporta tienen su funcionamiento en las tecnologías orientadas a objetos, tales como: objetos, clases, componentes y nodos.
- Permite modelar el comportamiento del sistema mediante: casos de uso, diagramas de secuencia y de componentes, entre otros.

1.3.3 Lenguaje de desarrollo Java

Java es un lenguaje compilado ya que todo programa realizado en Java ha de compilarse, e interpretado debido a que el código que se genera (*bytecodes*) es interpretado por la máquina virtual de Java donde existen varios niveles de seguridad, desde el ámbito del programador (permite realizar comprobación estricta de tipos durante la compilación, evitando con ello problemas tales como el desbordamiento de la pila), hasta el ámbito de la ejecución en la máquina virtual (gestiona la memoria dinámicamente). Una fuente común de errores en programación proviene del uso de punteros. En Java se han eliminado los punteros, el acceso a las instancias de clase se realiza a través de referencias. Además, el programador siempre está obligado a tratar las posibles excepciones que se produzcan en tiempo de ejecución. Dicho lenguaje define procedimientos para tratar estos errores, también posee mecanismos para garantizar la seguridad durante la ejecución comprobando, antes de ejecutar código, que no se viola ninguna restricción de seguridad del sistema donde se va a ejecutar. También cuenta con un cargador de clases, de modo que todas las clases cargadas a través de la red tienen su propio espacio de nombres para no interferir con las clases locales. (Franco García 2000).

1.4 Herramientas utilizadas

En el presente epígrafe se caracterizan las herramientas que facilitaron el desarrollo del módulo Administración de bases de datos del sistema ABCD v3.0.

1.4.1 Visual Paradigm versión 8.0

Visual Paradigm es una herramienta CASE profesional que soporta el ciclo de vida completo del desarrollo de software (análisis y diseño, construcción, pruebas y despliegue). Permitió modelar diagramas de clases, secuencia, componentes, despliegue, modelo de BD (para BD relacionales), generar documentación. Posee un diseño centrado en Casos de Uso, enfocado al negocio y una capacidad de ingeniería directa e inversa al integrarse con el IDE de desarrollo. Esta herramienta funciona sobre múltiples plataformas, es fácil de instalar y actualizar, además, es compatible con otras ediciones. Visual Paradigm brinda soporte al modelado visual con UML y posee disponibilidad de

múltiples versiones en dependencia de las necesidades o condiciones del usuario(Appleby y Vandekopple 1998).

1.4.2 Eclipse versión Kepler Release 4.3

Eclipse es un entorno de desarrollo integrado que facilitó las tareas de edición, compilación y ejecución del módulo durante la fase de desarrollo. Este se considera una plataforma ligera, pues puede ser personalizada mediante la incorporación de *plugins* para proporcionar funcionalidades en dependencia de lo que se necesite, lo que lo hace versátil y portable. Para desarrollar el módulo Administración de bases de datos J-ISIS se utilizan los siguientes *plugins*: Virgo Tooling, Subclipse, Spring IDE, WTP y JBoss Tools. Eclipse también provee ayuda para codificar, ejemplo: posee un compilador en tiempo real, permite completamiento de código, generación de plantillas, formateo de código, manejo de notas en código, visualización de la documentación de código(«Eclipse Kepler 4.3» 2018).

1.4.3 Gestores de Bases de Datos

Un **sistema de gestión de bases de datos** (SGBD) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos, además de proporcionar herramientas para añadir, borrar, modificar y analizar los mismos. La colección de dichos datos se denomina Base de Datos, (o en inglés, *Database*)(Ocaña Bueno 2017). El equipo de desarrollo del sistema ABCD versión 3.0 decidió utilizar para la gestión de Bases de Datos J-ISIS (para los datos bibliográficos), a continuación, se caracteriza:

J-ISIS Suite versión de noviembre 2014

ISIS como software, una de las funciones que desempeña es fungir como servidor para proveer acceso a la Base de Datos J-ISIS, archivo en que la información está contenida en registros numerados secuencialmente (en inglés, *MFNs o Master File Numbers*) con valores sobre todo textuales entre otros (*integer, date, time*) almacenados en campos con una "etiqueta" (o identificador numérico) y subcampos (con identificador de un carácter). Subcampos, campos y registros son de longitud variable (se adaptan al tamaño de la información contenida en ellos) y de “ocurrencia variable”, entre 0 (no presente) y cualquier número mayor de ocurrencias, en J-ISIS, sin límite. Puede manejar registros bibliográficos, junto con datos de usuarios y datos de transacción (por ejemplo, préstamos) en una única BD. Para la integración el módulo Administración de bases de datos J-ISIS con las bases de datos J-ISIS se utiliza el componente *J-ISIS DataProvider* (Smet y Spinak 2009).

1.5 Tecnologías

En el presente epígrafe se caracterizan las tecnologías utilizadas en el desarrollo del módulo Administración de bases de datos del sistema ABCD v3.0.

1.5.1 Eclipse Virgo versión 3.6.3

Virgo, es un servidor de aplicaciones completamente modular, basado en Java. Diseñado para ejecutar aplicaciones Java y Spring. Ofrece una plataforma para desarrollar y desplegar aplicaciones Java. Virgo Kernel es compatible con conceptos básicos de Virgo, este está disponible como código abierto. Virgo Kernel también puede usarse independiente como una plataforma de aplicaciones OSGi («Eclipse Virgo» 2018).

Virgo dispone de:

- Extensión de consola de Equinox: Permite gestionar el servidor de aplicaciones Virgo y artefactos desplegados.
- Aprovisionamiento: Suministra automáticamente las dependencias de una aplicación incluyendo bundles, planes, plan de archivos y configuraciones, desde repositorios locales y remotos.
- Spring: Puede ser fácilmente configurado para utilizar una versión diferente de Spring (en el caso de ABCD versión 3.0, soporta versiones menores de Spring 4.0).

El servidor de aplicaciones Eclipse Virgo es una implementación de Eclipse. Permite la adición de bundles (posee mecanismos para gestionar la comunicación entre estos) y la configuración de planes. para su ejecución («Eclipse Virgo» 2018).

1.5.2 RAP versión 3.0.5

Plataforma de Aplicaciones Remotas (por sus siglas en inglés, *RAP*), es un *framework* para aplicaciones modulares y Aplicaciones de Internet Enriquecidas (en inglés, *Rich Internet Application* (RIA.)), desarrollado por Eclipse. RAP se integra con tecnologías de Eclipse (por ejemplo: Equinox y Virgo). Ofrece un amplio conjunto de componentes que posibilitan construir interfaces gráficas y aplicaciones (widgets). Estas pueden ser desarrolladas para el escritorio, el navegador de Internet, y clientes móviles. Mediante la integración de RAP con tecnologías Java los widgets pueden ser desplegados directamente como paquetes OSGi en Virgo y otros servidores de aplicaciones. Los mismos pueden ser instalados de forma remota, iniciados, detenidos, actualizados y desinstalados sin necesidad de reiniciar. RAP implementa un conjunto de herramientas widget estándar (en inglés, *The Standard Widget Toolkit* (SWT)) que forma parte de Eclipse, el programador al hacer uso de este, utiliza el lenguaje de programación Java, muy similar a como si programara para una aplicación de escritorio con SWT. De

esta forma se genera el .html y el .js (archivo de texto plano que contiene scripts de JavaScript, y que permite, ser modificado con cualquier editor de textos) desde Java, el cual es generado con *qooxdoo* («Eclipse Kepler 4.3» 2018).

1.5.3 OSGi Equinox versión 3.8.2

Open Services Gateway Initiative (por sus siglas en inglés, OSGi), es una recomendación que posee diferentes implementaciones (por ejemplo: Apache Félix, *Eclipse Equinox, Knoplerfish*). Su objetivo es definir las especificaciones que permitan diseñar plataformas compatibles que puedan proporcionar múltiples servicios. OSGi intenta solventar los problemas del tradicional "*classloader*" de la máquina virtual y de los servidores de aplicaciones Java. Este *framework* proporciona a los desarrolladores un entorno orientado a servicios y basado en componentes («Equinox | The Eclipse Foundation» 2018).

Equinox proporciona un marco de trabajo Java de uso general, seguro y administrado que soporta el despliegue dinámico de aplicaciones conocidas como "*Bundles*" o módulos.

Algunas de las características que componen este marco de trabajo:

- Es un sistema de módulos para la plataforma Java.
- Incluye reglas de visibilidad, gestión de dependencias y versionado de los *bundles*.
- La instalación, arranque, parada, actualización y desinstalación de *bundles* se realiza dinámicamente en tiempo de ejecución sin tener que detener por completo la plataforma.
- Se trata de una arquitectura orientada a servicios dentro de la Máquina Virtual.

Podríamos destacar algunos de los principales beneficios que proporciona esta tecnología:

- Reutilización del código.
- Simplificar los proyectos en los que participan muchos desarrolladores en diferentes equipos.
- Se puede utilizar en sistemas más pequeños.
- Gestiona los despliegues locales o remotos.
- No es una tecnología cerrada.

Equinox es una implementación de OSGi y es utilizado por la plataforma de Eclipse. El modelo de programación de OSGi (*Equinox*) permite definir componentes de software dinámicos, es decir, los servicios OSGi, que también pueden ser parte de una aplicación basada en Eclipse. OSGi (*Equinox*) determina una arquitectura común para proveedores de servicios y desarrolladores, también permite desarrollar, desplegar y trabajar con servicios de manera coordinada («Equinox | The Eclipse Foundation» 2018).

1.5.4 Spring Dynamic Modules versión 3.1.0

Spring Dynamic Modules (por sus siglas en inglés, *Spring DM*) es un proyecto de Spring que permite utilizar el Framework del mismo nombre para el desarrollo de aplicaciones que pueden ser ejecutadas en la plataforma OSGi, además, simplifica el desarrollo de paquetes OSGi utilizando conceptos y características de este, tales como la inyección de dependencias (la cual permite construir el software con poco acoplamiento) y de archivos XML de configuración (*beans-context y osgi-module-context*) («Spring Dynamic Modules Reference Guide» 2017).

Entre otras funcionalidades de Spring DM se encuentran:

- Busca automáticamente paquetes basados en *Spring* y crea instancias del contexto de la aplicación *Spring* para cada *bundle* buscado.
- Busca e importa dinámicamente una referencia de servicios OSGi como *beans* a través de configuraciones XML.

Ventajas de la Inyección de dependencias:

- Proporciona bajo acoplamiento entre componentes (provee componentes más desacoplados e independientes).
- La responsabilidad de cada uno de los componentes será muy clara y un cambio podrá ser más fácil de implementar.

1.6 Conclusiones parciales

El estudio de los sistemas existentes de gestión bibliotecaria, realizado en este capítulo, permitió conocer las principales características de dichos sistemas. Se lograron fundamentar las bases teóricas de la investigación y se analizaron los conceptos básicos relacionados al objeto de estudio definido. La caracterización de las herramientas, tecnologías, metodología y lenguajes de programación, permitió conocer sus beneficios, así como formar las bases propicias para formular una propuesta de solución, que, a su vez, cumpla con el objetivo de la investigación.

Capítulo 2. Propuesta de solución

En el presente capítulo se describe la propuesta de solución para el sistema a desarrollar, sus características y funcionalidades. Se incluyen los modelos, diagramas, requisitos funcionales y no funcionales que requiere el módulo a implementar, así como los actores y descripciones de los casos de uso del sistema. Además, se describe la arquitectura utilizada y los patrones de diseño.

2.1 Descripción de la propuesta de la solución

Actualmente, el sistema ABCD 3.0 no tiene un módulo de Administración de bases de datos J-ISIS, por lo que los usuarios autorizados no pueden gestionar las bases de datos de registros bibliográficos mediante el sistema ABCD v3.0. Este módulo agiliza la gestión en las bases de datos por el equipo de desarrollo y permite a los usuarios con privilegios gestionar estas bases de datos sin necesidad de tener conocimientos técnicos ajenos sobre el sistema.

2.2 Modelo de Dominio

“El modelo de dominio captura los tipos de objetos de mayor relevancia en el contexto del sistema. Los objetos del dominio representan las “cosas” que existen o los eventos que suceden en el entorno en el que trabaja el sistema” (Sommerville, 2011). Este es el principal motivo por el que el modelo de dominio puede ser tomado como punto inicial para el diseño del sistema. Puede utilizarse para capturar y expresar los conceptos más importantes del contexto del mismo. Es una representación visual de los conceptos u objetos del mundo real, significativos para un problema o área de interés. Los analistas utilizan el modelo de dominio como un medio para comprender los procesos de negocios donde el sistema va a ser utilizado.

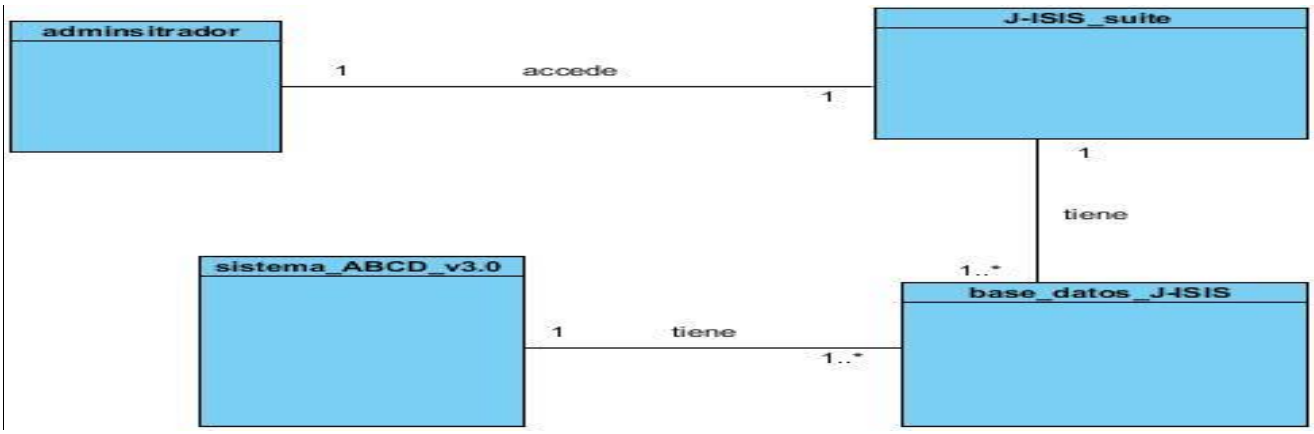


Figura 2: Modelo de dominio

Fuente: Elaboración propia

Descripción de los conceptos del dominio:

sistema_ABCD_v3.0: Sistema de automatización de bibliotecas y centros de documentación.

administrador: usuario con privilegios para acceder al módulo Administración de bases de datos J-ISIS encargado de controlar y velar por el correcto funcionamiento y realizar cambios significativos en este.

J-ISIS_suite: gestor de base de datos J-ISIS con el que se administran actualmente las bases de datos de registros bibliográficos.

base_datos_J-ISIS: bases de datos donde se encuentran todos los registros bibliográficos almacenados en la biblioteca.

2.3 Especificación de los requisitos funcionales

Los requerimientos de software son capacidades o condiciones que tienen que ser alcanzadas o poseídas por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. Define las funciones de qué es lo que el sistema será capaz de realizar, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen. Son el conjunto de propiedades que debe cumplir el software para ser exitoso en el entorno en el cual se usará. Estos deben ser comprensibles por clientes, usuarios y desarrolladores, deben tener una sola interpretación y estar definidos en forma medible y verificable. (Sommerville, 2011).

2.3.1 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones con las cuales el sistema debe cumplir e indican su comportamiento, es decir, son las funciones que el sistema debe realizar. Para una mejor comprensión de los requisitos es necesario tener conocimiento de los siguientes términos:

Tabla de definición de campo: La Tabla de Definición de Campos y los Formatos de Ingreso definen la estructura de los registros de la base de datos(Ascencio, Spinak y Egbert de Smet 2016).

Tabla de selección de campo: Contiene los campos indexados de la base de datos. Una Tabla de Selección de Campos define los criterios para extraer uno o más elementos de un registro del archivo maestro. Dependiendo del contexto en el cual se utilice una tabla de selección de campo, estos elementos pueden usarse para crear términos recuperables en el archivo inverso, correspondientes al registro del cual fueron extraídos, para la clasificación de registros en la secuencia deseada antes de producir un reporte impreso, o para reformatear registros durante una operación de importación o exportación (UNESCO y Hernández 2003).

Indexación: Indexar una base de datos requiere de una buena comprensión de los datos, las funciones de usuario y de cómo está indexada la base de datos. Los índices utilizan componentes clave de los datos de una tabla en una estructura binaria para mejorar la capacidad de búsqueda. Cada registro de datos en la tabla debe estar asociado con datos en el índice. Indexar puede aumentar notablemente la velocidad de búsqueda. Sin embargo, un inconveniente de los índices es que cada operación de inserción, actualización o supresión necesita una actualización de los índices. Cuando una tabla incluye índices múltiples, cada índice puede aumentar el tiempo que lleva procesar las actualizaciones de la tabla. Si se desea reducir el número de índices para mejorar la velocidad de procesamiento, se deben eliminar los índices que son menos valiosos a los efectos de la búsqueda. (UNESCO y Hernández 2003).

Hoja de trabajo: Son un conjunto de campos para la creación personalizada de formularios de registros para una determinada base de datos en la biblioteca, contiene algunos o todos los campos de la tabla de definición de campo. También es una o más pantallas definidas por el usuario son utilizadas para crear y/o actualizar los registros principales de la base de datos (UNESCO y Hernández 2003).

A continuación, se listan y describen los requisitos funcionales que debe cumplir el sistema para la solución planteada:

RF1. Crear base de datos (nombre, lista de campos definidos).

Descripción: permite al usuario crear una base de datos donde se guardan registros bibliográficos, cada base de datos tendrá nombre, se introducirán los campos que se definen en la tabla de definición de campo, los campos que están en la hoja de trabajo por defecto y los campos indexados de la tabla de selección de campo.

RF1.1. Crear tabla de definición de campo de la base de datos (lista de campos definidos).

Descripción: se crea una tabla donde se definen todos los campos que se utilizarán en la base de datos.

RF1.2. Crear tabla de selección de campo de la base de datos (lista de campos definidos).

Descripción: se crea una tabla donde se establecen todos los campos que serán indexados en la base de datos, los cuales se seleccionan de los campos que están definidos en la tabla de definición de campo.

RF1.3. Crear hoja de trabajo por defecto (lista de campos definidos).

Descripción: se crea una hoja de trabajo inicial donde se eligen los campos que será utilizados en esta, según la tabla de definición de campo.

RF2. Mostrar bases de datos.

Descripción: muestra un listado de todas las bases de datos de registros bibliográficos que están disponibles en la biblioteca.

RF3. Editar base de datos (nombre, lista de campos definidos).

Descripción: permite al usuario editar la estructura de una base de datos, aquí se puede modificar la estructura de la base de datos donde se guardan los registros bibliográficos, la base de datos se localizará por el nombre, también se podrá editar los campos que se definen en la tabla de definición de campo, los que están en la hoja de trabajo por defecto y los campos indexados que estarán en la tabla de selección de campo.

RF3.1. Editar tabla de definición de campo de la base de datos (lista de campos definidos).

Descripción: se edita una tabla donde se muestran todos los campos que se utilizarán en la base de datos y permite agregar, eliminar o modificar estos.

RF3.2. Editar tabla de selección de campo de la base de datos (lista de campos definidos).

Descripción: se edita una tabla donde se muestran todos los campos indexados en la base de datos los cuales se pueden cambiar, eliminar o agregar algún otro que pertenezca a la tabla de definición de campo.

RF3.3. Editar hoja de trabajo por defecto (lista de campos definidos).

Descripción: se puede editar la hoja de trabajo inicial donde se eligieron los campos que será utilizados en esta, según la tabla de definición de campo, los cuales pueden cambiarse, eliminarse o agregar algunos nuevos.

RF4. Reindexar base de datos.

Descripción: se selecciona la base de datos que se reindexará y se procede a actualizar los índices a partir de la tabla de selección de campo.

RF5. Crear hoja de trabajo (base de datos, lista de campos definidos).

Descripción: se crea una hoja de trabajo seleccionando la base de datos a la pertenecerá y luego se seleccionan los campos de la tabla de definición de campo de esta base de datos.

RF6. Editar hoja de trabajo (base de datos, lista de campos definidos).

Descripción: se edita una hoja de trabajo seleccionando la base de datos a la que pertenece, se pueden agregar nuevos campos de la tabla de definición de campo, cambiar los existentes en ella o eliminar algunos de los que están.

RF7. Eliminar hoja de trabajo.

Descripción: se elimina una hoja de trabajo de una base de datos seleccionada.

RF8. Editar propiedades (campo o subcampo).

Descripción: permite al usuario editar las propiedades de los campos y subcampos de la hoja de trabajo, se selecciona el campo al que se le modificarán las propiedades y se pueden editar las propiedades siguientes: repetible, obligatorio, texto, valor por defecto, mensaje de ayuda, control visual, generar número de control, generar fecha y hora de la última transacción, lista de selección y base de datos referenciada.

2.3.2 Requisitos no funcionales

"Los requisitos no funcionales son los requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este, como la fiabilidad y el tiempo de respuesta"(Novelo Can y Olivera Sosa 2010). En resumen, los requisitos no funcionales representan aquellas cualidades que debe poseer el sistema pero que no son funcionalidades específicas.

Los requisitos no funcionales que debe tener el componente a desarrollar son:

Usabilidad:

1. El sistema proporcionará una interfaz, con facilidad de uso para usuarios con conocimientos mínimos de informática.
2. Los grupos de botones y vínculos deben estar organizados por funcionalidad, con el objetivo de una mejor experiencia del usuario con el módulo.
3. Los mensajes para interactuar con los usuarios deben describir bien el suceso, en idioma español y no deben revelar información interna.
4. Los colores a utilizar en el desarrollo del módulo deben ajustarse a los colores del sistema ABCD v3.0 disponible en https://abcd.uci.cu.

Restricciones de diseño e implementación:

1. Se debe desarrollar el módulo siguiendo el estándar de codificación utilizado para implementar el sistema ABCD v3.0.
2. Seguir la metodología utilizada en el centro para la implementación de sus proyectos, (AUP-UCI).

Seguridad:

1. Para acceder al módulo los usuarios deben tener permisos especiales como administrador del módulo.

Portabilidad:

1. El módulo debe ser multiplataforma, capaz de ejecutarse sobre cualquier sistema operativo.

Legal:

1. Las herramientas seleccionadas para el desarrollo del módulo están respaldadas por licencias libres.

2.4 Modelo de casos de uso del sistema

El diagrama de casos de uso es un modelo del sistema que contiene actores, casos de uso y sus relaciones. Describe lo que hace el sistema para cada tipo de usuario y permite que los desarrolladores del mismo y los clientes lleguen a un acuerdo sobre los requisitos, es decir, sobre las condiciones y posibilidades que debe cumplir dicho sistema.

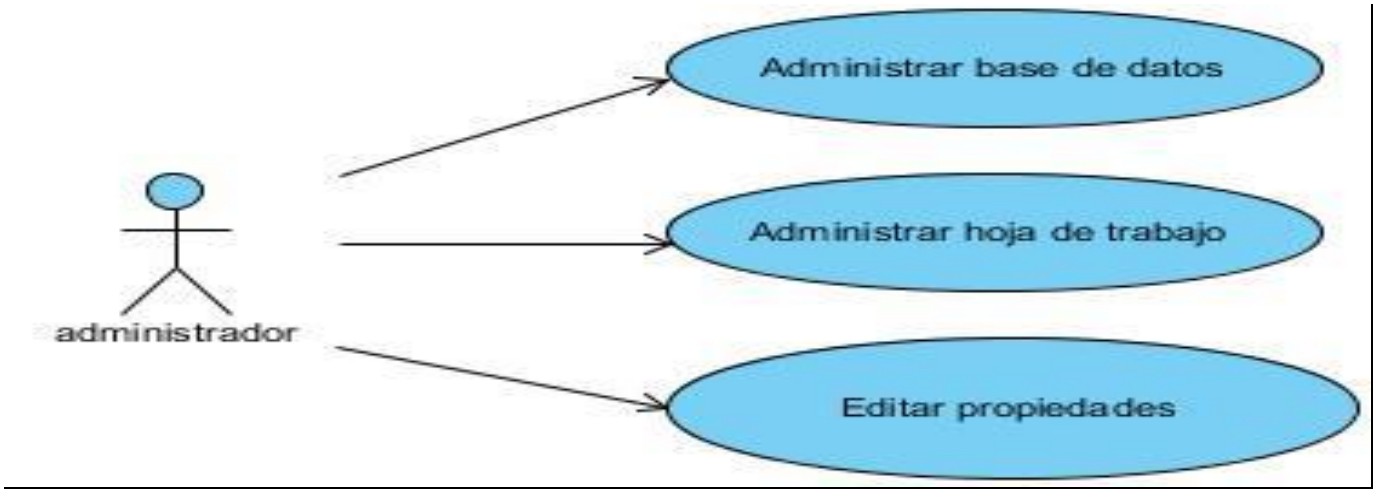


Figura 3: Diagrama de casos de uso del Módulo

Fuente: Elaboración propia

2.4.1 Actores del sistema

Los actores del sistema representan entidades externas que interactúan directamente con el sistema (personas, máquinas u otros sistemas). A continuación, se muestran el actor que interactúa con el sistema:

Actor	Descripción
Administrador	Se encarga de acceder al módulo Administración de bases de datos J-ISIS donde controla y vela por el correcto funcionamiento y realiza cambios significativos en este.

Tabla 2: Actor del sistema

Fuente: Elaboración propia

2.4.2 Descripción de los casos de uso

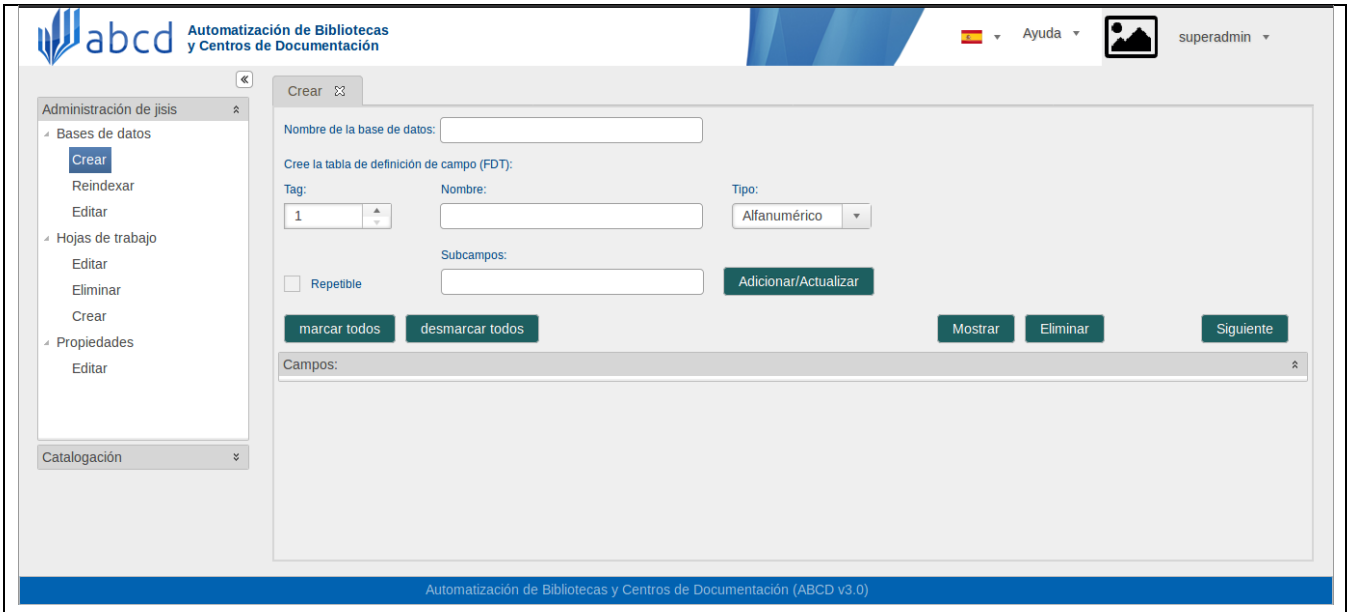
Para una mejor comprensión de la funcionalidad asociada a cada caso de uso es necesario describir cada uno de estos. Dicha descripción puede ser elaborada de forma breve o extendida y debe ir acompañada del prototipo de interfaz de usuario respectivo. Los casos de uso son artefactos narrativos que describen bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario.

A continuación, se realiza la descripción de los casos de uso:

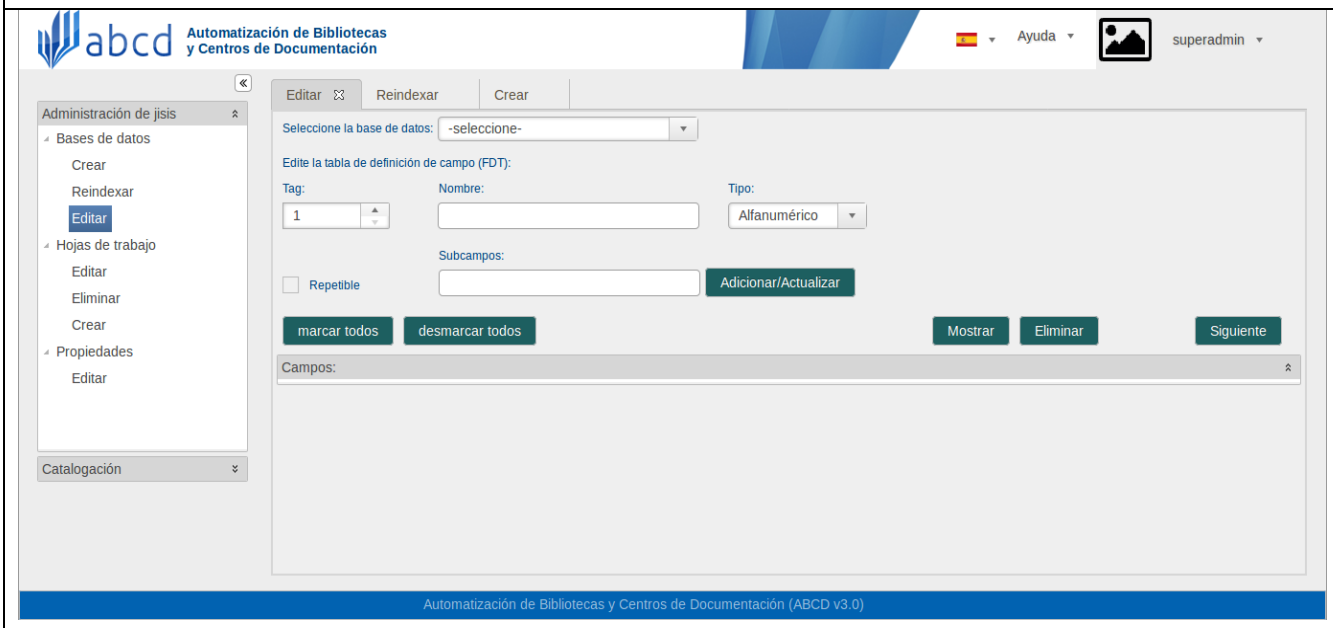
Caso de uso	Administrar base de datos.
Actores	Administrador
Resumen	El caso de uso se inicia cuando el usuario selecciona la opción Base de datos y elige crear, editar o reindexar una base de datos. El sistema solicita los datos en dependencia de la opción seleccionada, el usuario crea una base de datos, realiza las modificaciones deseadas, crea la hoja de trabajo por defecto, reindexa una base de datos y puede ver las bases de datos existentes en la opción editar. El sistema valida y guarda los datos, finalizando así el caso de uso.

Precondiciones	El usuario debe estar autenticado y tener permisos de administrador del módulo.
Referencias	RF1, RF1.1, RF1.2, RF1.3, RF2, RF3, RF3.1, RF3.2, RF4
Prioridad	Media.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción Base de datos.	2. El sistema muestra las opciones: crear base de datos, editar base de datos y reindexar base de datos. 4. Si selecciona la opción crear base de datos (ir a la sección “Crear base de datos”). Si selecciona la opción eliminar base de datos (ir a la sección “Reindexar base de datos”). Si selecciona la opción Editar base de datos (ir a la sección “Editar base de datos”).
Flujo Normal de Eventos	
Sección Crear Base de datos	
Acción del actor	**Respuesta del sistema**
1.Selecciona la opción Crear base de datos	2. Muestra una interfaz donde solicita los datos: nombre de la base de datos y los botones siguiente y cancelar.
3. Introduce el nombre de la nueva base de datos.	
4. Selecciona la opción Siguiente.	5. Muestra una interfaz donde se crea la tabla de definición de campo.

6. Introduce los campos de la tabla de definición de campo.	
7. Selecciona la opción Siguiente.	8. Muestra una interfaz donde se crea hoja de trabajo por defecto.
9. Selecciona los campos que tendrá la hoja de trabajo por defecto, según los definidos en la tabla de definición de campo.	
10. Selecciona la opción Siguiente.	11. Muestra una interfaz donde se crea la tabla de selección de campo.
12. Selecciona los campos de la tabla de definición de campo que tendrá la tabla de selección de campo.	
13. Selecciona la opción Guardar.	14. Valida los datos.
	15. Informa que se guardó la nueva base de datos con el siguiente mensaje “Base de datos guardada correctamente”.
Flujo alterno	
Acción del actor	**Respuesta del sistema**
4. Selecciona la opción Siguiente.	5.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 2.
7. Selecciona la opción Siguiente.	8.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 5.
10. Selecciona la opción Siguiente.	11.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 5.
13. Selecciona la opción Guardar.	14.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 8.
Prototipo de interfaz gráfica de usuario	



Sección Editar Base de Datos	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción Editar Base de datos.	2. Muestra una interfaz donde el usuario puede modificar los campos de la tabla de definición de campo.
3. Edita los datos de la tabla de definición de campo.	
4. Selecciona la opción Siguiente.	5. Muestra una interfaz donde se pueden editar los campos que tendrá la hoja de trabajo por defecto.
9. Selecciona los campos que tendrá la hoja de trabajo por defecto, según los definidos en la tabla de definición de campo.	
6. Selecciona la opción Siguiente.	7. Muestra una interfaz donde el usuario puede modificar los campos de la tabla de selección de campo.
8. Edita los datos de la tabla de selección de campo.	

9. Selecciona la opción Guardar.	10. Valida los datos.
	11. Informa que se guardó la base de datos con el mensaje “Base de datos guardada correctamente”.
Flujo alterno	
Acción del actor	**Respuesta del sistema**
4. Selecciona la opción Siguiente.	5.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 5.
7. Selecciona la opción Siguiente.	8.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 5.
10. Selecciona la opción Guardar.	11.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 8.
Prototipo de interfaz gráfica de usuario	
	
Sección Reindexar Base de Datos	

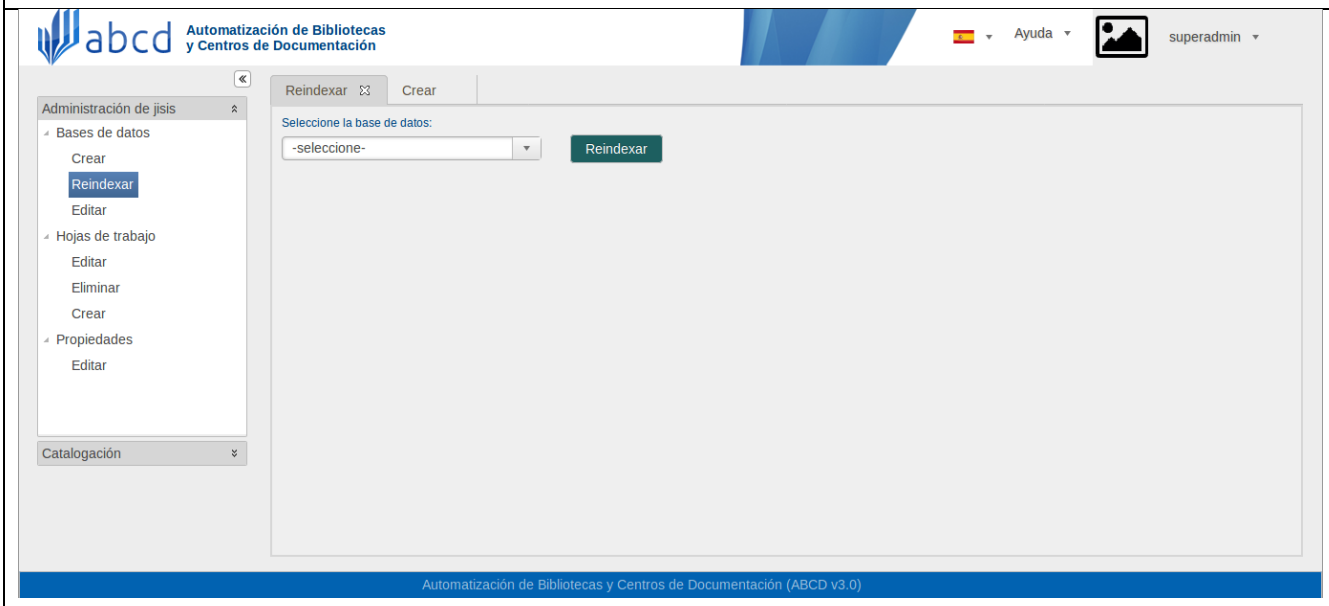
Acción del actor	Respuesta del sistema
1. Selecciona la opción Reindexar base de datos.	2. Muestra una interfaz donde muestra el nombre de cada base de datos y se debe seleccionar una para reindexar.
3. Selecciona la base de datos deseada.	4. Reindexa la base de datos, actualiza los índices a partir de la tabla de selección de campo.
	5. Muestra un mensaje “Base de Datos Reindexada”.
Prototipo de interfaz gráfica de usuario	
	

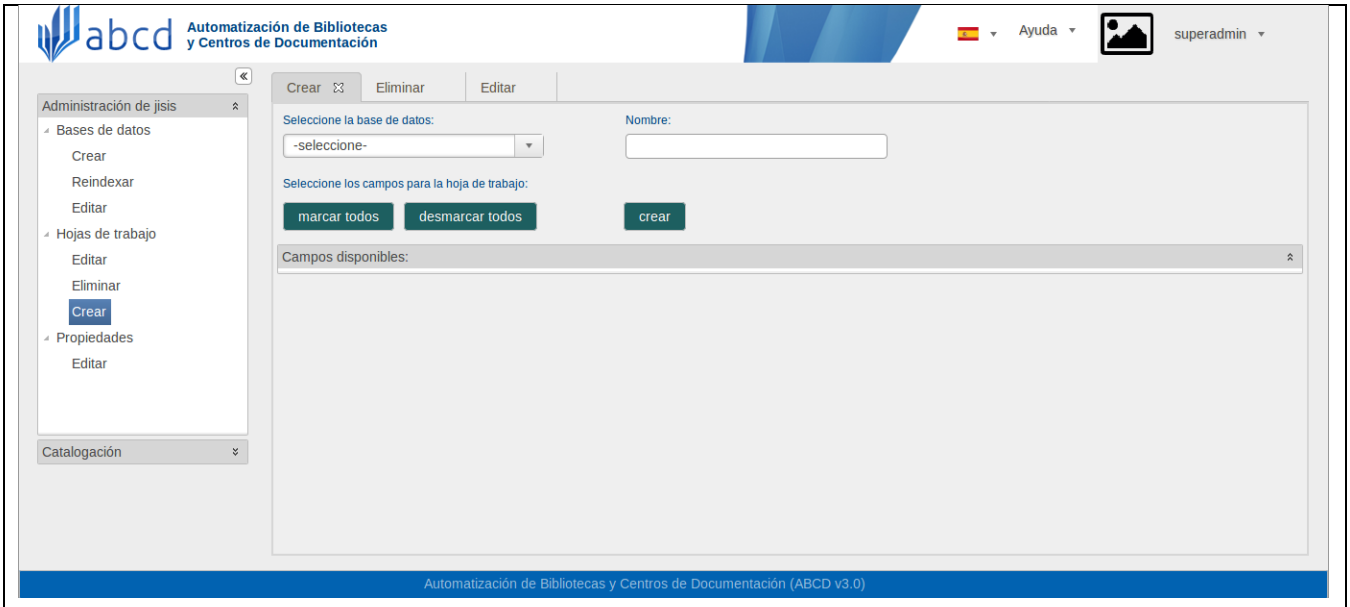
Tabla 3: Descripción del caso de uso Administrar base de datos

Fuente: Elaboración propia

Caso de uso	Administrar Hoja de trabajo.
Actores	Administrador

Resumen	El caso de uso se inicia cuando el usuario selecciona la opción Gestionar Hoja de trabajo y se muestran todas las bases de datos existentes en el sistema, elige la base de datos donde se encuentra la hoja de trabajo y se muestran todas las hojas de trabajo de la base de datos seleccionada. Luego se puede elegir crear, eliminar o editar una hoja de trabajo. El sistema solicita los datos en dependencia de la opción seleccionada, el usuario crea una hoja de trabajo, realiza las modificaciones deseadas, elimina una hoja de trabajo y puede ver las hojas de trabajo existentes. El sistema valida y guarda los datos, finalizando así el caso de uso.
Precondiciones	El usuario debe estar autenticado y tener permisos de administrador del módulo.
Referencias	RF5, RF6, RF7
Prioridad	Media.
Flujo Normal de Eventos	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción Gestionar Hoja de trabajo.	2. El sistema muestra todas las bases de datos existentes en el sistema.
3. Selecciona la base de datos donde se encuentra la hoja de trabajo que se desea Administrar.	4. Se muestran las Hoja de trabajo almacenadas en una tabla y las opciones: crear, editar y eliminar hoja de trabajo dando clic sobre la que desee consultar.

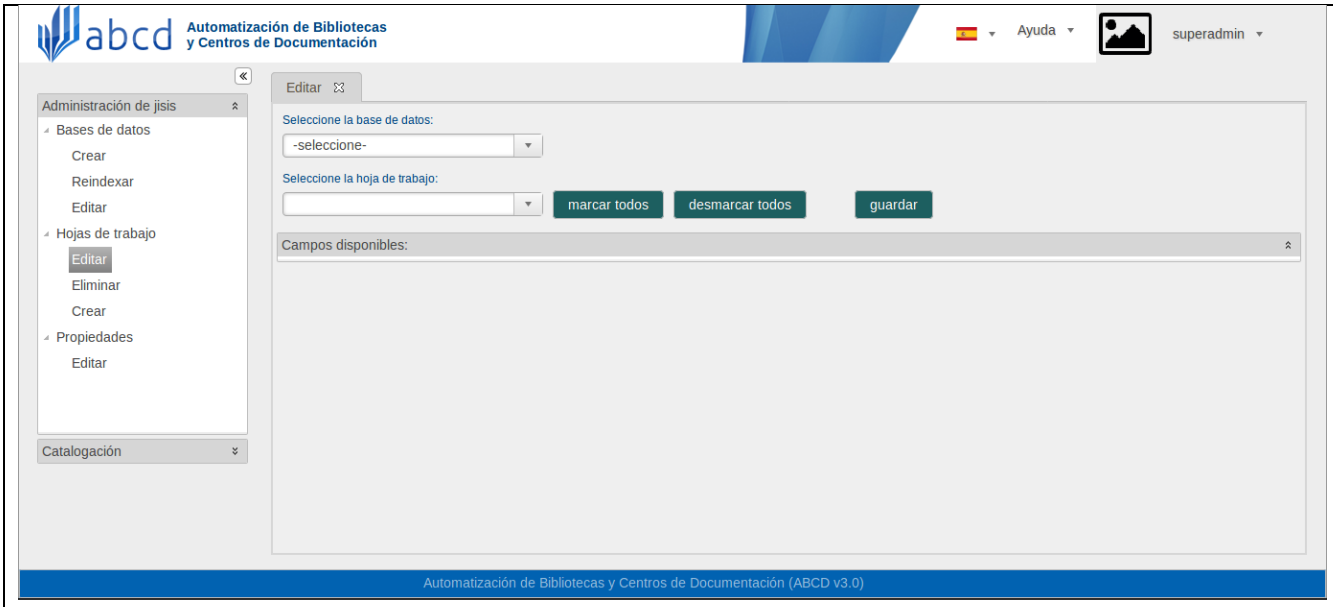
	Si selecciona la opción crear (ir a la sección “Crear Hoja de trabajo”). Si selecciona la opción eliminar (ir a la sección “Eliminar Hoja de trabajo”). Si selecciona la opción Editar (ir a la sección “Editar Hoja de trabajo”).
Flujo Normal de Eventos	
Sección Crear Hoja de trabajo	
Acción del actor	**Respuesta del sistema**
1.Selecciona la opción Crear Hoja de trabajo	2. Muestra una interfaz donde debe poner nombre a la hoja de trabajo y seleccionar los campos de la tabla de definición de campo que tendrá la hoja de trabajo.
3. Selecciona los campos de la nueva Hoja de trabajo según la tabla de definición de campo.	
4. Selecciona la opción Guardar.	5. Valida los datos.
	6. Informa que se guardó la nueva Hoja de trabajo con el mensaje “Hoja de trabajo guardada correctamente”.
Flujo alterno	
Acción del actor	**Respuesta del sistema**
4. Selecciona la opción Guardar.	5.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 2.
Prototipo de interfaz gráfica de usuario	



Sección Editar Hoja de trabajo	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción Editar Hoja de trabajo.	2. Muestra una interfaz donde puede seleccionar los campos de la tabla de definición de campo que desea incorporar a la hoja de trabajo o los que están en ella y desea retirar.
3. Edita campos de la Hoja de trabajo.	
4 Selecciona la opción Guardar.	5. Valida los datos.
	6. Informa que se guardó la Hoja de trabajo con el mensaje “Hoja de trabajo guardada correctamente.

Flujo alterno	
Acción del actor	**Respuesta del sistema**
4. Selecciona la opción Guardar.	5.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 2.

Prototipo de interfaz gráfica de usuario



Sección Eliminar Hoja de trabajo	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción Eliminar Hoja de trabajo.	2. Muestra una interfaz donde muestra el nombre de la Hoja de trabajo y pide que confirme si desea eliminarla.
3. Selecciona Si desea eliminar la Hoja de trabajo.	4. Elimina la Hoja de trabajo.
	5. Informa que se eliminó la Hoja de trabajo con el mensaje “Hoja de trabajo eliminada correctamente”.
Prototipo de interfaz gráfica de usuario	

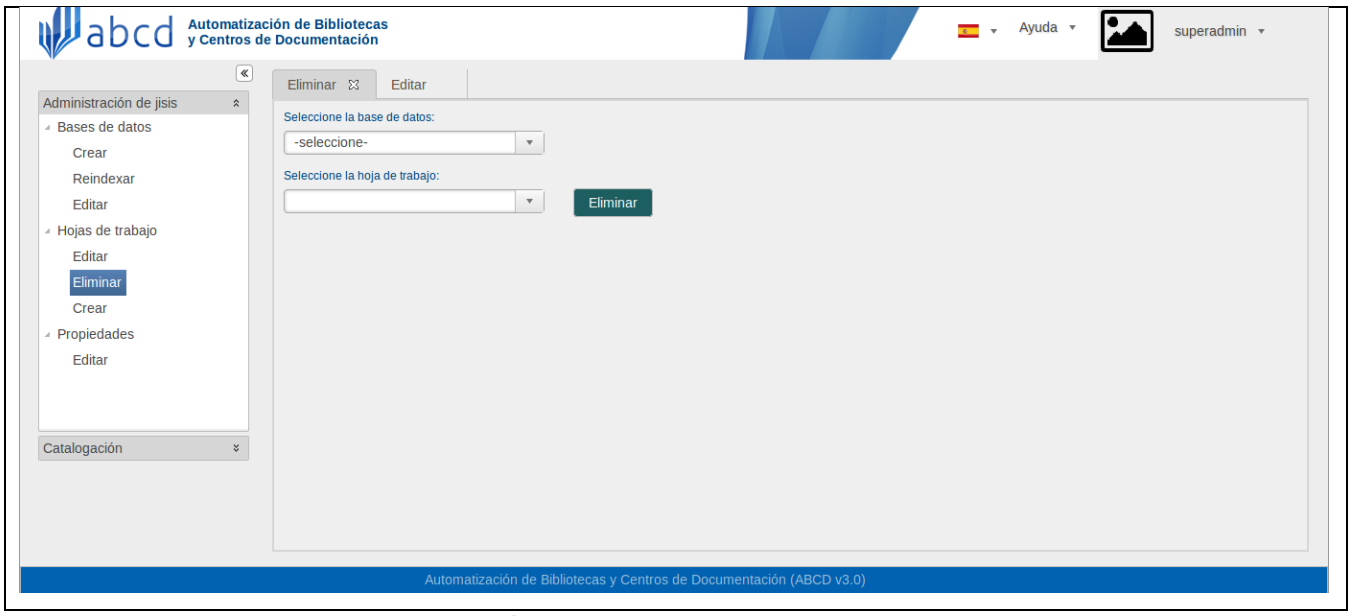


Tabla 4: Descripción del caso de uso gestionar Hoja de trabajo

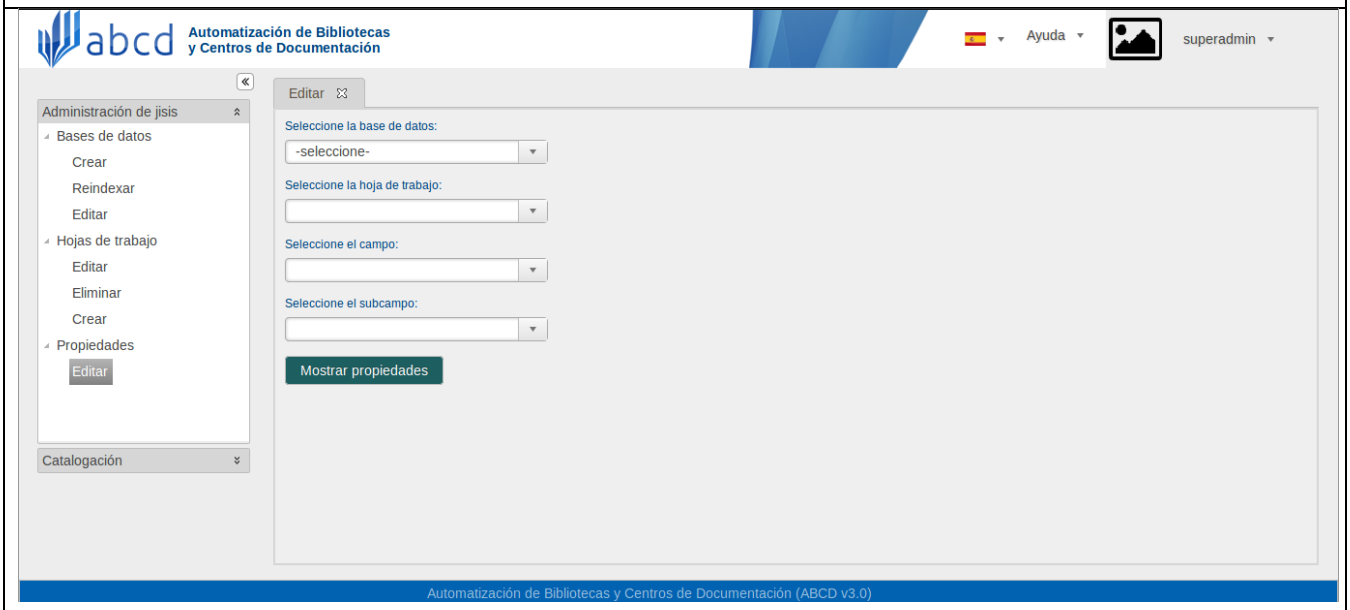
Fuente: Elaboración propia

Caso de uso	Editar propiedades.
Actores	Administrador
Resumen	El caso de uso se inicia cuando el usuario selecciona la opción Gestionar propiedades de Hoja de trabajo, se muestran todas las bases de datos del sistema, se escoge la base de datos donde está la hoja de trabajo en la que se trabajará y se elige la hoja de trabajo, luego se muestran sus propiedades y se selecciona la que se desee editar.
Precondiciones	El usuario debe estar autenticado y tener permisos de administrador del módulo.
Referencias	RF8.
Prioridad	Media.
Flujo Normal de Evento	

Acción del actor	Respuesta del sistema
1. Selecciona la opción Editar las propiedades.	2. El sistema muestra las propiedades de una Hoja de trabajo donde permite la edición de estas.
3. Selecciona la opción Guardar	4. Valida los datos
	5. Informa por un mensaje “Propiedades guardadas correctamente”.

Flujo alterno	
Acción del actor	**Respuesta del sistema**
3. Selecciona la opción Guardar	5.1 Muestra un mensaje de error “Existen campos erróneos” y vuelve a la acción 2.

Prototipo de interfaz gráfica de usuario



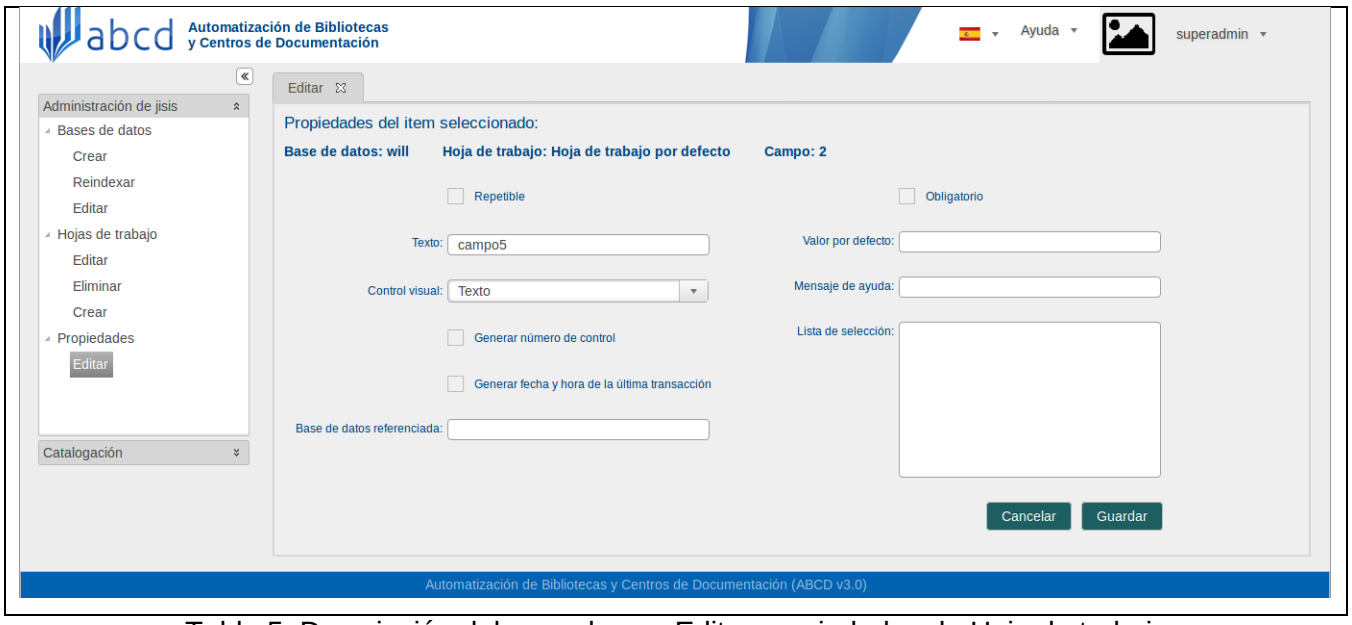


Tabla 5: Descripción del caso de uso Editar propiedades de Hoja de trabajo

Fuente: Elaboración propia

2.4.3 Arquitectura utilizada

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 el 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(Cervantes 2018).

La Arquitectura Base Orientada a Servicios (ABOS), definida por el equipo de desarrollo para el sistema ABCD versión 3.0, posee una estructura de 4 capas, esta, permitió dividir el trabajo modularmente. Los principales beneficios del estilo de arquitectura basado en capas son (Lopez 2018):

- Abstracción. Las capas permiten cambios que se realizan a nivel abstracto. Se puede incrementar o disminuir el nivel de abstracción usado en cada capa de la “pila” jerárquica.
- Aislamiento. El estilo de arquitectura en capas permite asilar los cambios en tecnologías a ciertas capas para reducir el impacto en el sistema.

- Rendimiento. Distribuir las capas entre múltiples sistemas (físicos) puede incrementar la escalabilidad, la tolerancia a fallos y el rendimiento.
- Mejoras en Pruebas. La capacidad de realizar pruebas se beneficia de tener interfaces bien definidas para cada capa, así como de la habilidad para cambiar a diferentes implementaciones de las interfaces de las capas.

En una arquitectura de n-capas se requiere diseñar objetos realmente reutilizables, que puedan usarse para proyectos futuros. Si los requisitos para un proyecto cambian es necesario reescribir el código; aún más importante es el hecho que, dejando la seguridad que proporciona una arquitectura por capas, se corre el riesgo de diseñar un sistema que sea más complejo que el pensado originalmente. Esto evita el avance, puesto que una decisión descuidada en el diseño puede tener aspectos no considerados. Sin embargo, la transmisión de aplicaciones de objetos distribuidos en CORBA (*Common Object Request Broker Architecture*) no es tan rápida como aquellas diseñadas con protocolos de socket estándar; CORBA es lento porque necesita ser general, y su gran fortaleza es su mayor debilidad(Gonzaga et al. 2006).

A continuación, se muestra la estructuración de la arquitectura definida para el sistema ABCD v3.0:

- Capa de presentación: 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 las ventanas, la autentificación, el formato de los reportes, menús, gráficos y demás elementos multimedia. En dicha capa se encuentra el bundle cu.uci.abcd.jisis.db.management.ui donde se localizan las interfaces de usuario, en ellas se debe implementar IContributor para que todas tengan la misma estructura visual, además para la construcción de interfaces.
- La capa de negocios: reúne los aspectos de software que automatizan los procesos de negocio. Conocida también como capa Lógica de la Aplicación aquí estará contenido el bundle cu.uci.abcd.jisis.db.management.impl. Además, 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.
- La capa de datos: en la cual se define la conexión con el servidor y la BD, es en esta capa donde se almacena, actualiza y consultan los datos del sistema. En dicha capa se encuentra el bundle cu.uci.abcd.dataprovider.jisis donde se localizan las interfaces de acceso a datos.

La arquitectura definida para el sistema ABCD v3.0 contiene, además, una capa de infraestructura que se encuentra a lo largo de todas las demás. Esta, añade gestión dinámica de permisos proporcionando

un modelo de seguridad capaz de autenticar el código a través de la ubicación del bundle y de sus paquetes, además provee una infraestructura para desplegar y manejar aplicaciones que deben ejecutarse en un entorno seguro y controlado. La comunicación entre capas se mantiene a través de servicios, donde cada capa proporciona servicios a la capa inmediata superior y se sirve de las prestaciones que le brinda la inmediata inferior. Esto lo posibilita la utilización de OSGI, quien proporciona a los desarrolladores un entorno orientado a servicios y basado en componentes.

2.5 Patrones de diseño GRASP

GRASP significa *General Responsability Asignment Software Patterns* (patrones generales de software para asignar responsabilidades). Estos describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones («Patrones de Diseño» 2015). A continuación, se muestran los patrones utilizados en el diseño del sistema ABCD versión 3.0.

Experto

Experto es un patrón basado en el principio básico de la asignación de responsabilidades; suele utilizarse en el diseño orientado a objetos. Con la utilización de este patrón en el módulo Administración de bases de datos J-ISIS conservó el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. El comportamiento se distribuye entre las clases que cuentan con la información requerida, alentando con ello definiciones de clases cohesivas que se pueden comprender y mantener. Ejemplo de lo planteado es la clase *Field* (Mochorro Reyes 2004).

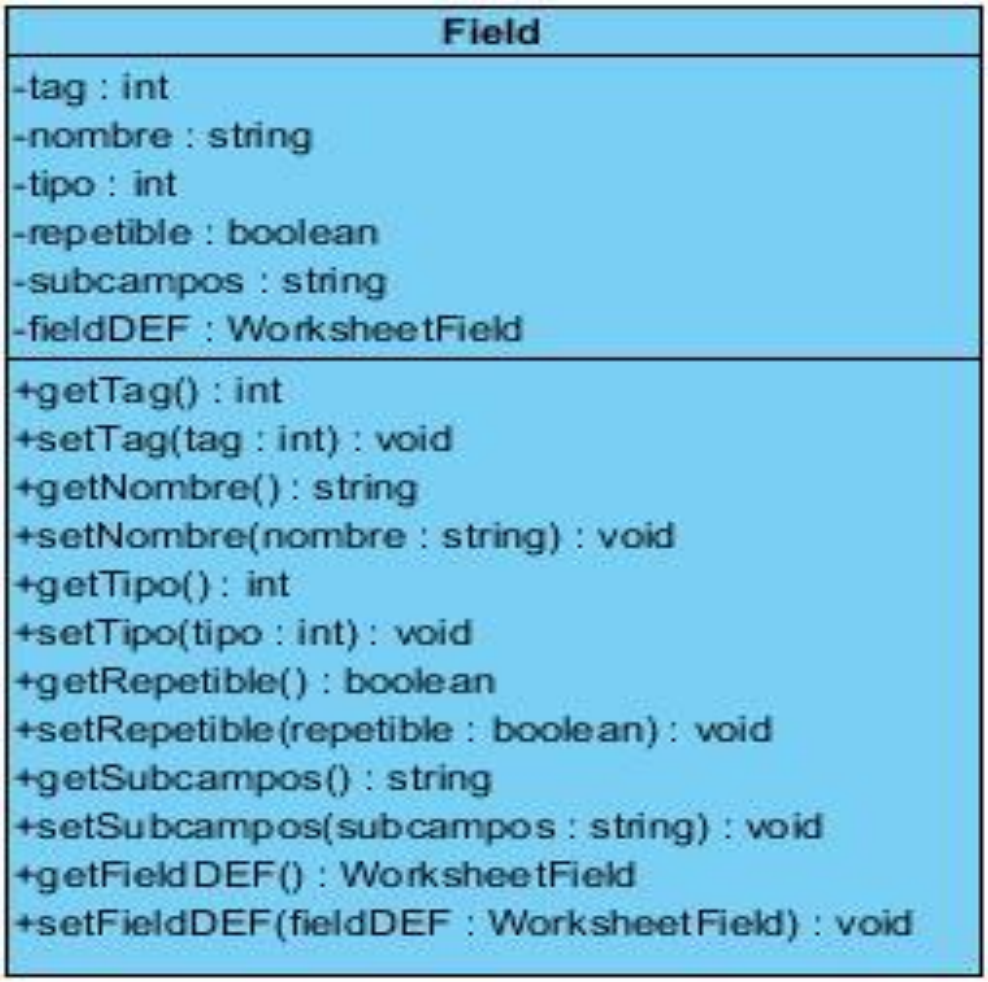


Figura 4: Ejemplo de patrón Experto, Clase *Field*.

Fuente: Elaboración propia

Creador

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. De este modo se obtendrá un diseño con mayor cohesión y la información se mantendrá encapsulada (disminución del acoplamiento). Ejemplo: la clase *DataBaseManager.* (Mochorro Reyes 2004).

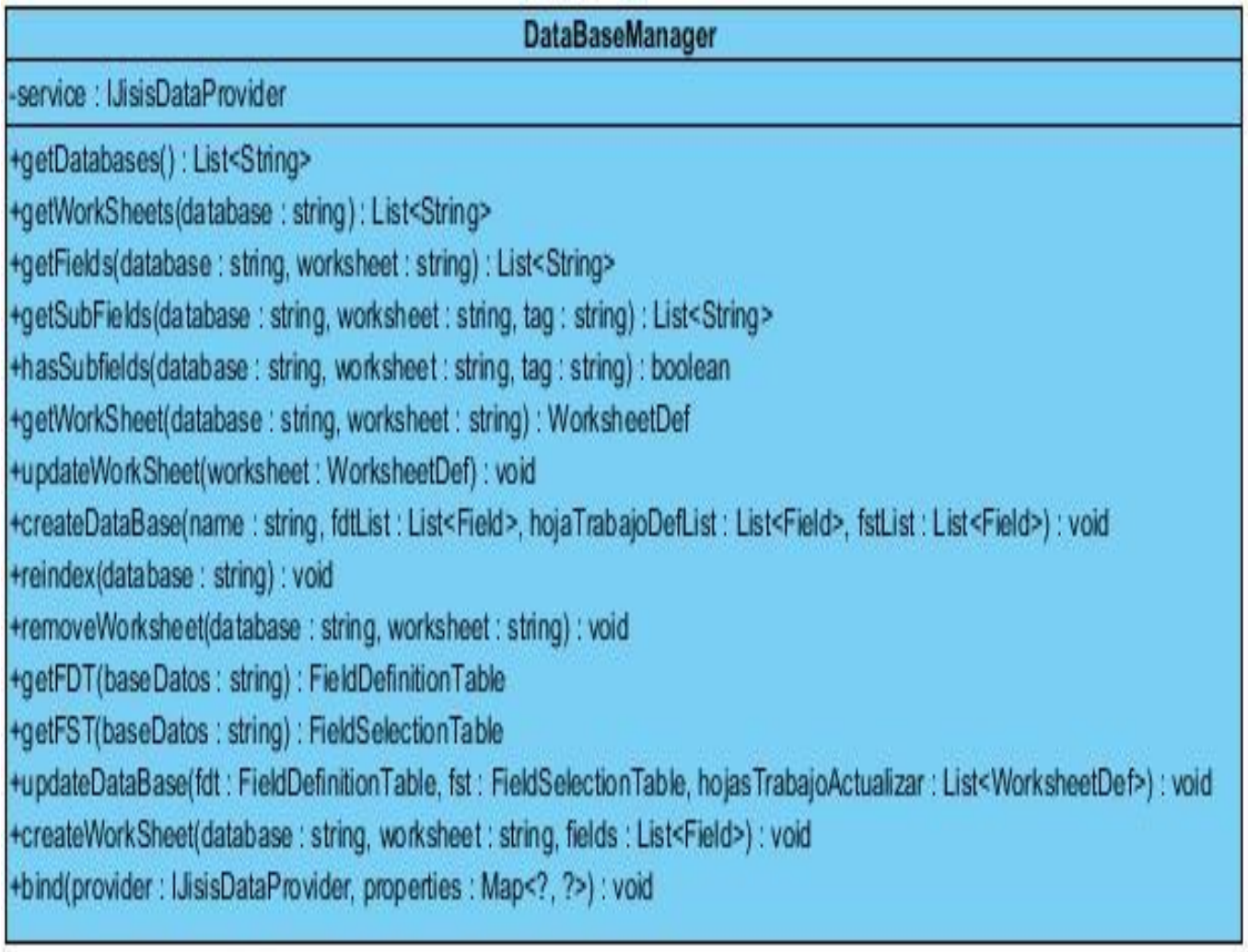


Figura 5: Ejemplo de patrón Creador, Clase *DataBaseManager*.

Fuente: Elaboración propia

Bajo Acoplamiento

Es la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma, en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización, y disminuyendo la dependencia entre ellas. Un ejemplo es la figura siguiente (Mochorro Reyes 2004):

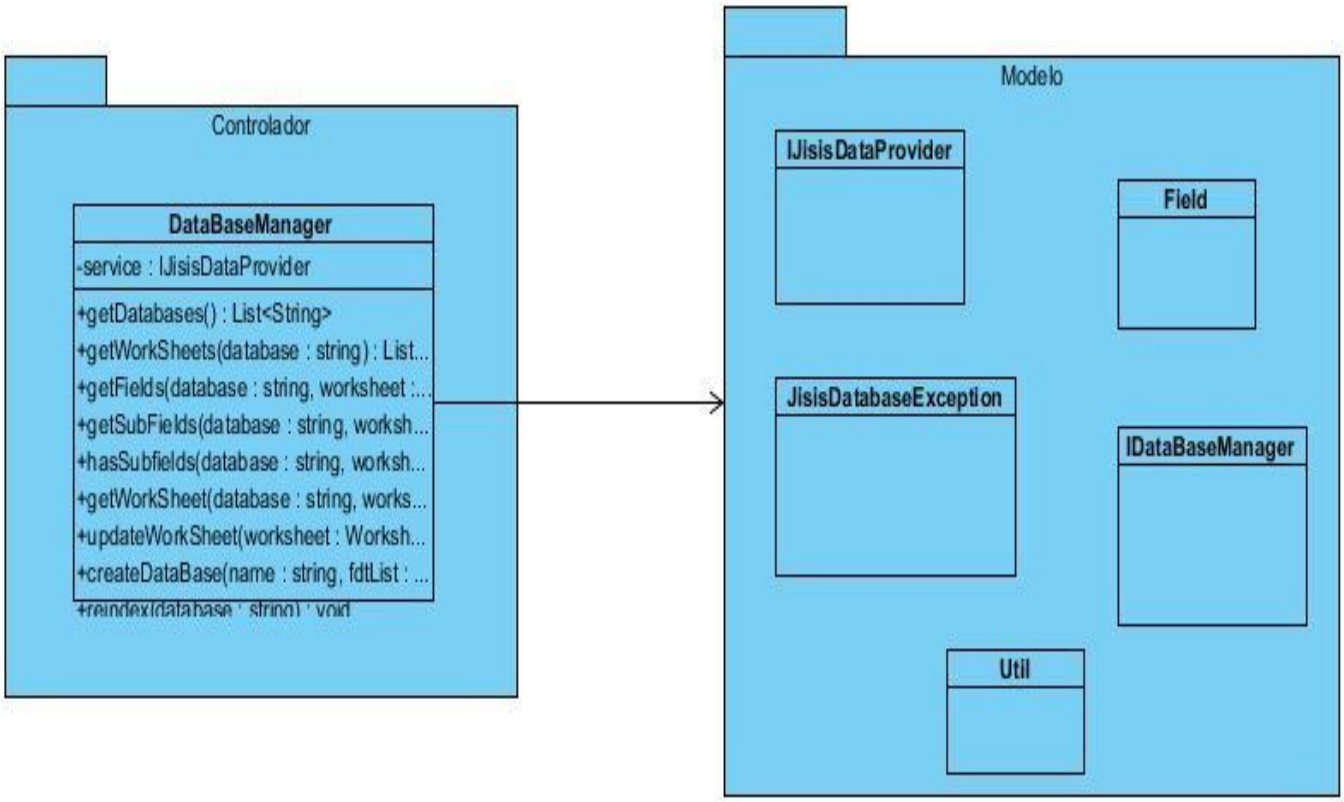


Figura 6: Ejemplo de patrón Bajo Acoplamiento, integración de clases.

Fuente: Elaboración propia

Alta Cohesión

La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase; por lo que puede ser calificada como alta cuando existen clases con responsabilidades estrechamente relacionadas que no realicen mucho trabajo. Este patrón se basa en la asignación de responsabilidades, de modo que la cohesión siga siendo alta. Se ejemplifica en la clase *ProxyController*. (Mochorro Reyes 2004).

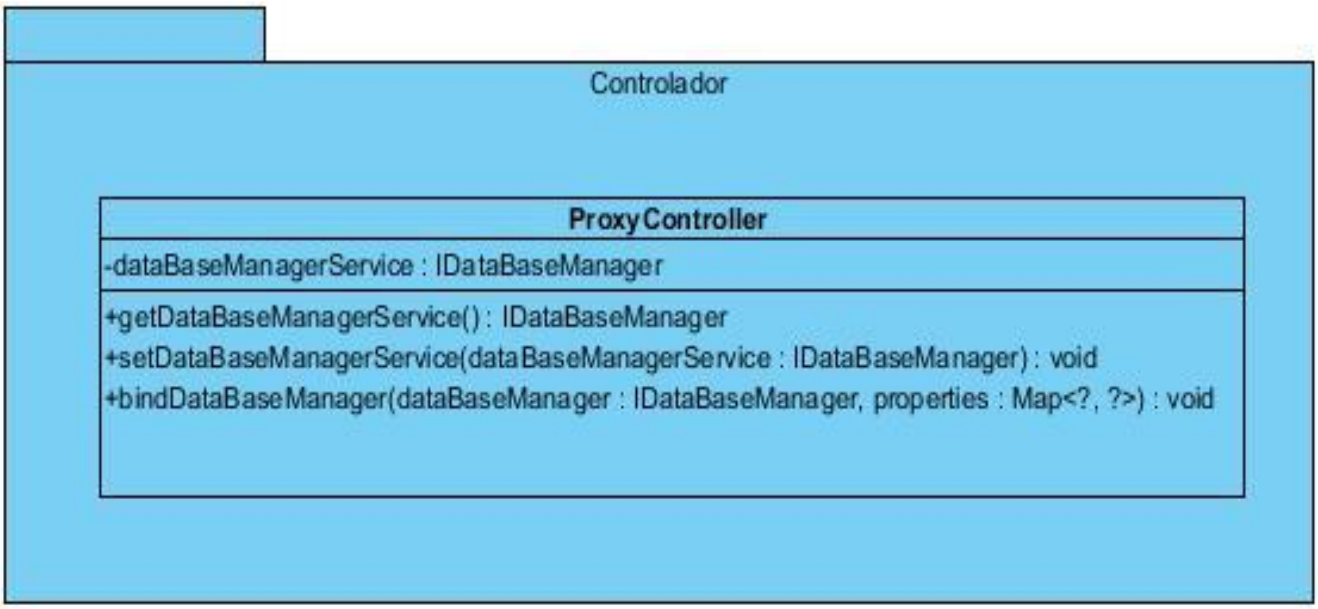


Figura 7: Ejemplo de patrón Alta Cohesión, *ProxyController.*

Fuente: Elaboración propia

Controlador

El patrón controlador 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 para aumentar la reutilización de código y a la vez tener un mayor control. Ejemplo: la clase *DataBaseManager*, esta se encarga de realizar y controlar todas las funcionalidades. Ver Figura 5. (Mochorro Reyes 2004).

2.6 Diagrama de clases del diseño utilizando estereotipos web

Los diagramas de clases del diseño utilizando estereotipos web especifican la estructura de clases de un sistema, así como sus relaciones. Definen de forma correcta las relaciones de dependencia, generalización y asociación de clases que constituyen el sistema.

A continuación, se muestran los diagramas de clases del diseño utilizando estereotipos web de los casos de uso del módulo:

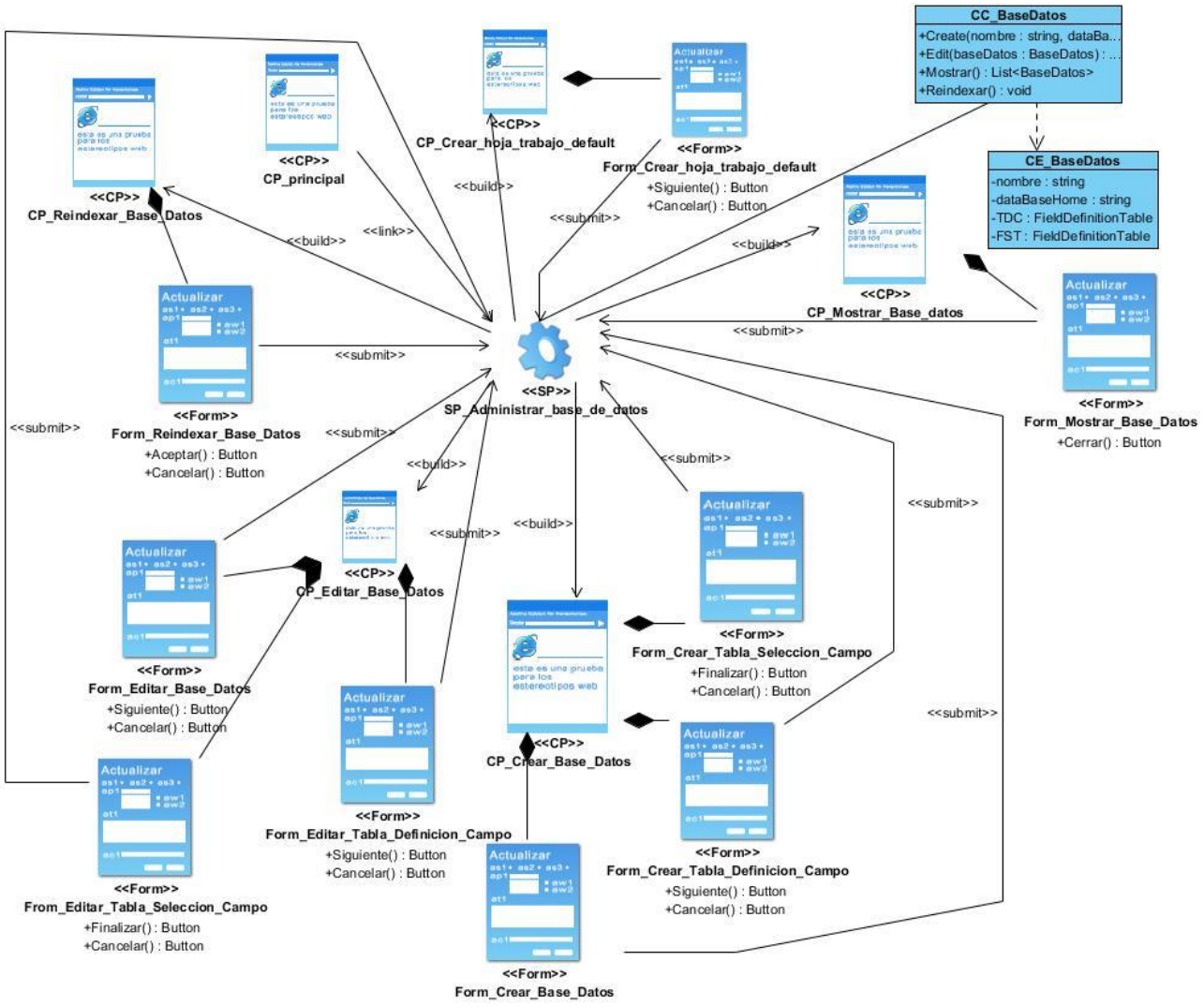


Figura 8: Diagrama de clases del diseño con estereotipos web del caso de uso Administrar base de datos

Fuente: Elaboración propia

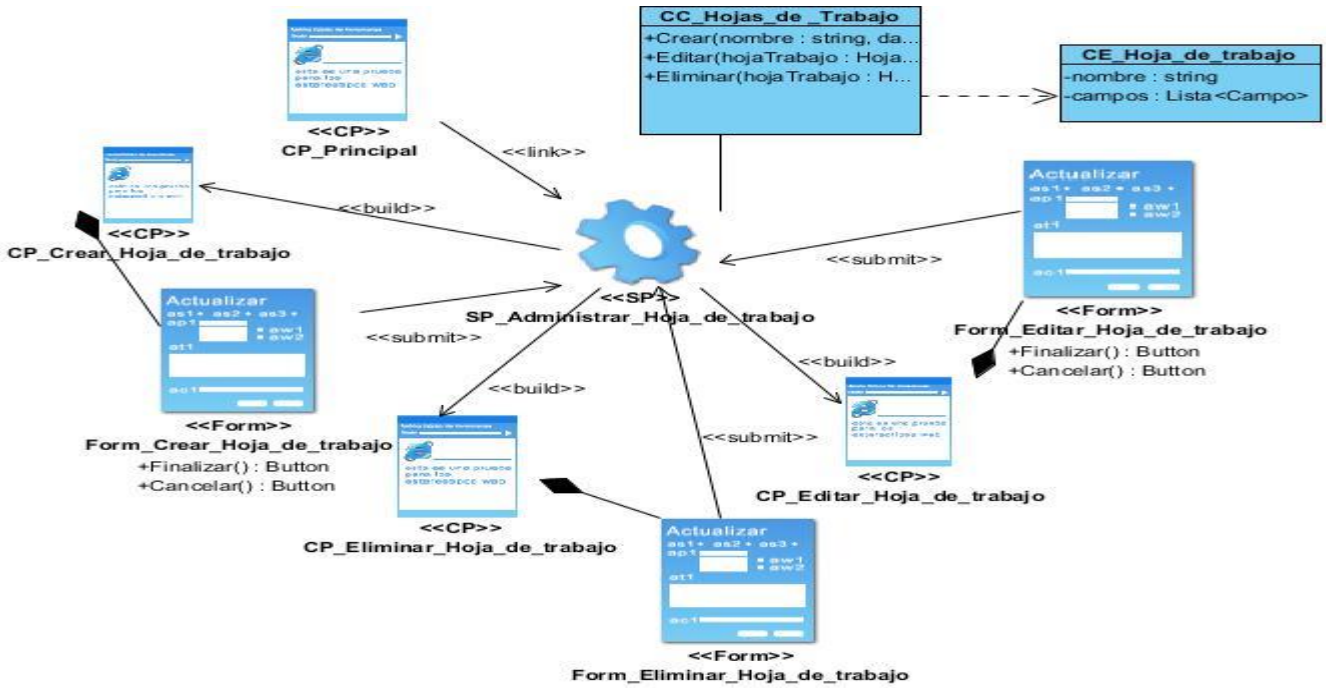


Figura 9: Diagrama de clases del diseño con estereotipos web del caso de uso Administrar hoja de trabajo

Fuente: Elaboración propia

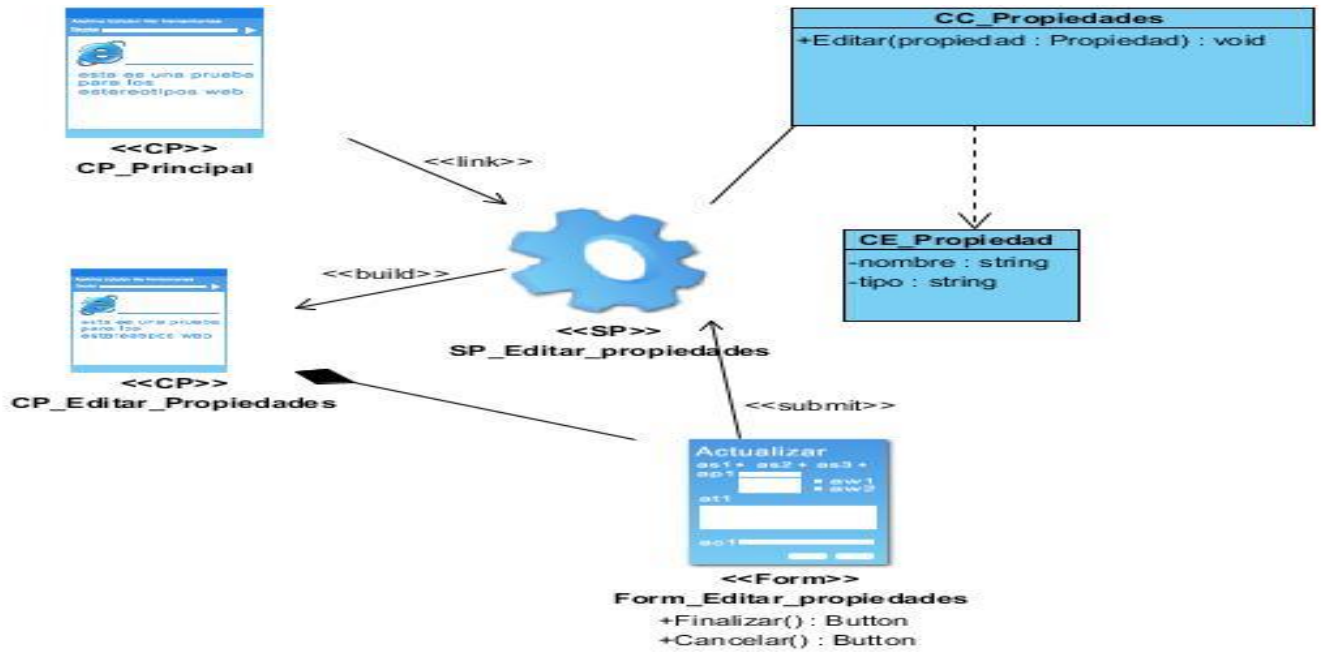


Figura 10: Diagrama de clases del diseño con estereotipos web del caso de uso Editar propiedades

Fuente: Elaboración propia

2.7 Conclusiones parciales

Luego de describir las características que debe cumplir el nuevo módulo del sistema ABCD v3.0, basado en tecnologías libres y de realizar el análisis y diseño correspondiente, se concluye que los requisitos funcionales y no funcionales identificados a partir del proceso de obtención de los requisitos y los artefactos generados, constituyeron una guía fundamental para la construcción de la propuesta de solución. La realización del modelo de dominio permitió la comprensión del proceso de forma ingenieril de la administración de base de datos sin la solución propuesta. Además, con la utilización de la arquitectura 4 capas utilizada en el sistema ABCD v3.0, se garantiza una mayor organización, reutilización de funciones y código más legible.

Capítulo 3. Implementación y prueba de resultados

En el presente capítulo se propone explicar a partir de los resultados del diseño, la implementación del módulo de Administración de bases de datos J-ISIS para el sistema ABCD v3.0. Como parte de la implementación se muestra el diagrama de despliegue en el que se visualiza la situación física del sistema. Se definen las pruebas de software y algunos de los resultados obtenidos de las mismas además de los estándares de codificación. (Cechich y Moore 2000)

3.1 Modelo de implementación del módulo

El modelo de implementación representa cómo los elementos del modelo de diseño se implementan en términos de componentes. De igual forma describe cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y la relación existente entre ellos(Jacobson, Booch y Rumbaugh 2000).

3.2 Estándares de codificación

Los estándares de codificación se definen con vistas a mejorar el entendimiento del código perteneciente al módulo en cuestión por otros desarrolladores y alcanzar una uniformidad en el mismo. A continuación, se listan los elementos pertenecientes al estándar de codificación definido para el desarrollo del módulo de Administración de bases de datos del sistema ABCD v3.0 (Ver Figura 6):

- Utilizar nombres en inglés para las clases y métodos.
- Dejar espacio de separación dentro del código, para que sea entendible.
- Utilizar el estilo de escritura *CamelCase*.
- Escribir en inglés los nombres de las variables.
- Los nombres de las variables deben ser cortos y significativos.

3.3 Diagrama de despliegue

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema. Muestra la configuración de los elementos de hardware (nodos) y cómo los elementos y artefactos del software se trazan en esos nodos(Systems Pty Ltd 2018).

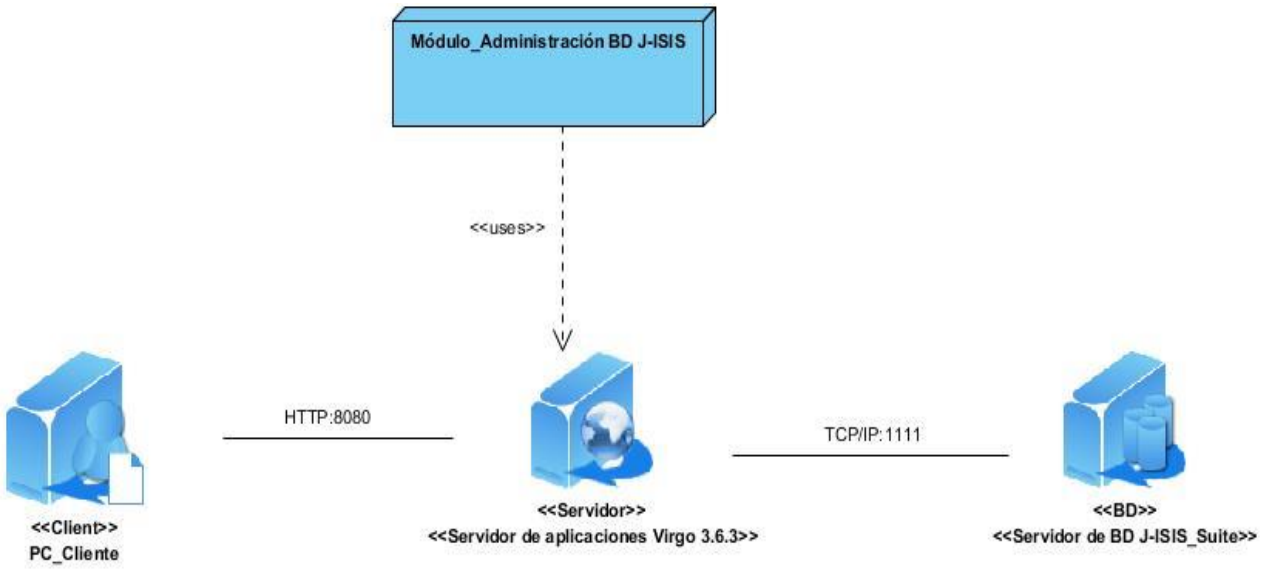


Figura 11: Diagrama de despliegue del módulo de Administración de base de datos J-ISIS

Fuente: Elaboración propia

A continuación, se describen características de los nodos del diagrama de despliegue:

- Nodo PC_Cliente: representa las computadoras que utilizará el usuario para interactuar con la aplicación. Se comunica con el Servidor de Aplicaciones a través del protocolo HTTP mediante el puerto 8080.
- Nodo Servidor de Aplicaciones Virgo 3.6.3: representa el servidor donde se encuentra instalado el sistema, el cual se conecta al servidor de BD J-ISIS mediante el protocolo TCP/IP por el puerto 1111.
- Nodo Servidor de Base de Datos J-ISIS_Suite: es el servidor J-ISIS donde se almacena y gestionan las bases de datos de los registros bibliográficos.
- Nodo Módulo_Administración BD J-ISIS: representa el módulo de Administración de bases de datos J-ISIS del sistema ABCD v3.0, el cual utiliza el servidor de aplicaciones.

3.4 Pruebas al módulo

Un instrumento adecuado para determinar el status de la calidad de un producto software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad. En las pruebas se usan casos de prueba, especificados de forma estructurada mediante Técnicas de prueba («Pruebas de Software» 2016).

La prueba de software es imprescindible para garantizar la calidad de la aplicación representando una revisión final de las especificaciones del diseño y del código.

Elementos a tener en cuenta para que una prueba tenga éxito:

- Estrategia de prueba
- Niveles de prueba
- Tipo de prueba
- Método de prueba
- Caso de prueba

Estrategia de prueba

- Describe el enfoque y los objetivos generales de las actividades de prueba.
- Incluye los niveles de prueba a ser diseccionados, el tipo de prueba a ser ejecutada y los casos de prueba diseñados para lograr los objetivos.
- Define: Técnicas de pruebas (manual o automática) y herramientas a ser usadas. Criterios de éxitos y culminación de las pruebas. Consideraciones especiales relacionadas con los recursos necesarios para realizar las pruebas.

Una estrategia de prueba del software integra los métodos de diseño de caso de pruebas del software en una serie bien planeada de pasos que desembocará en la eficaz construcción del software. La estrategia proporciona un mapa que describe los pasos que se darán como parte de la prueba, indica cuando se planean y cuando se dan estos pasos, además de cuanto esfuerzo, tiempo y recursos consumirán. Por tanto, cualquier estrategia de prueba debe incorporar la planeación de pruebas, el diseño de casos de pruebas, la ejecución de pruebas y la recolección y evaluación de los datos resultantes (Pressman 2010).

Una estrategia de prueba del software debe ser lo suficientemente flexible como para promover un enfoque personalizado. Al mismo tiempo, debe ser lo adecuadamente rígido como para promover una planeación razonable y un seguimiento administrativo del avance del proyecto (Pressman 2010).

Niveles de prueba

La prueba es aplicada para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo. Se distinguen los siguientes niveles de pruebas (González Jorrin 2007):

- Pruebas unitarias

- Prueba de integración
- Prueba de sistema
- Prueba de implantación
- Prueba de aceptación
- Prueba de regresión

Tipos de prueba (González Jorrin 2007):

En la presente investigación se realizan dos tipos de pruebas para determinar la calidad del software:

- Pruebas funcionales:

Se asegura el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. Este tipo de pruebas están basadas en técnicas de caja negra, que es, verificar la aplicación y sus procesos internos mediante la interacción con la aplicación y analizar los resultados. Se realiza específicamente la técnica de partición de equivalencia correspondiente a las pruebas de caja negra, verificando por casos de prueba el cumplimiento de todos los requisitos definidos para la implementación del módulo.

- Pruebas de integración:

Las pruebas de integración se realizan para verificar que exista una correcta comunicación entre los módulos que componen el sistema final. Se utilizan para verificar que los componentes del sistema funcionan correctamente actuando en conjunto. En el caso de la investigación presentada, el objetivo es verificar la integración directa entre los módulos Administración de bases de datos J-ISIS y Circulación, permitiendo comprobar que los dos módulos puedan consumir correctamente los servicios necesarios para su funcionamiento.

- Pruebas de aceptación:

Se realiza una serie de pruebas de aceptación a fin de permitir al cliente validar todos los requerimientos. Realizada por el usuario final en lugar de por los ingenieros de software, una prueba de aceptación puede variar desde una “prueba de conducción” informal hasta una serie de pruebas planificadas y ejecutadas sistemáticamente. De hecho, la prueba de aceptación puede realizarse durante un periodo de semanas o meses, y mediante ella descubrir errores acumulados que con el tiempo puedan degradar el sistema.

Casos de prueba

Los casos de prueba deben diseñarse para descubrir errores debidos a cálculos erróneos, comparaciones incorrectas o flujo de control inadecuado. Se diseñan para garantizar que: se satisfagan todos los requisitos de funcionamiento, se logran todas las características de comportamiento, todo el contenido es preciso y se presenta de manera adecuada, se logran todos los requisitos de rendimiento, la documentación es correcta y se satisfacen la facilidad de uso y otros requisitos (Pressman 2010).

Métodos de Prueba

Caja Negra

Las pruebas de caja negra, también llamadas pruebas de comportamiento, se enfocan en los requisitos funcionales del software; es decir, las técnicas de prueba de caja negra le permiten derivar conjuntos de condiciones de entrada que revisarán por completo todos los requisitos funcionales para un programa (Pressman 2010).

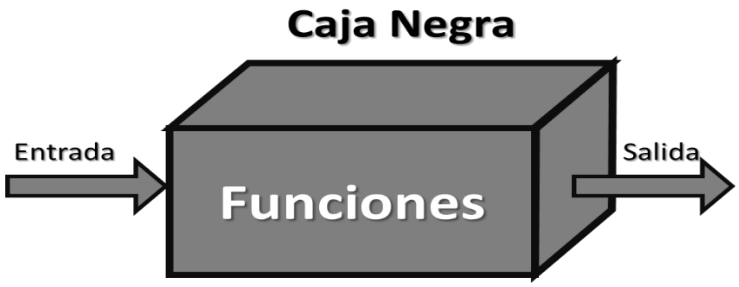


Figura 12: Método de prueba, Caja Negra

Fuente Elaboración propia.

Se basa en técnicas como:

Técnica de partición de equivalencia: es un método de prueba de caja negra que divide el dominio de entrada de un programa en clases de datos de los que pueden derivarse casos de prueba. Un caso de prueba ideal descubre de primera mano una clase de errores que de otro modo podrían requerir la ejecución de muchos casos de prueba antes de observar el error general (Pressman 2010).

Técnica de análisis de valores límites: esta técnica prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables (González Jorrin 2007).

Técnica de Grafos de causa-efecto: es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones (González Jorrin 2007).

La técnica a emplear para desarrollar las pruebas de caja negra es la de partición de equivalencia teniendo en cuenta que es la técnica que utilizan los proyectos de la UCI y por las características antes mencionadas.

Pruebas funcionales:

A continuación, se muestra el caso de prueba de caja negra aplicadas los casos de uso; las celdas de las tablas que se muestran contienen las letras V, I o N/A. V indica válido, I indica inválido y N/A que no es necesario proporcionar un valor de la variable en este caso:

Escenario	Descripción	Nom-bre BD	Tag	Nombre de campo	Tipo	Sub-campo	Repetible	Respuesta del sistema
EC1.1 Crear la tabla de definición de campo de la base de datos nueva	Permite al usuario crear una tabla de definición de campo para la base de datos nueva	V abcd	V 3	V Autor	V Alfa num érico	V abc	V si	Permite introducir de forma obligatoria datos: • Nombre • Tag • Nombre del campo • Tipo de campo Opcionalmente puede el usuario agregar otros datos como. • Sub-campos • Repetible Luego permite: • Continuar con la creación de la BD.

								• Cancelar cerrando la interfaz. • Eliminar campos agregados. • Marcar todos y desmarcar todos los campos agregados. Verifica que no exista una base de datos con el mismo nombre y campos con el mismo número de Tag dentro de esta.
EC1.2 Cancelar la operación	Permite al usuario cancelar la operación.	N/A	N/A	N/A	N/A	N/A	N/A	Cancela las operaciones realizadas.
EC1.3 Datos incorrectos	Acción que realiza el sistema cuando el usuario introduce datos incorrectos.	V Test	I x	V Título	V Alfa num érico	V cfd	V No	Muestra un mensaje de error: “Existen campos incorrectos”. Muestra un indicador sobre los campos incorrectos.

EC1.4 Campos vacíos	Acción que realiza el sistema cuando el usuario deja campos vacíos.	I (Vacío)	V 5	I (Vacío)	V Alfa num érico	V (Vacío)	V Si	Muestra un mensaje de error: “Existen campos vacíos que son obligatorios, por favor complete estos campos”. Muestra un indicador sobre el/los campos vacíos.
EC1.5 Base de datos existente y/o Tag repetido	Acción que realiza el sistema cuando el usuario le da un nombre existente a la base de datos nueva.	I abcd	I 3	V Autor	V Alfa num érico	V abc	V Si	Muestra un mensaje de error:” Existe una Base de datos con el mismo nombre”. Muestra un mensaje de error:” Existe registrado un campo con el mismo Tag”. Muestra el indicador sobre los campos afectados.

Tabla 6: Caso de prueba de caja negra del caso de uso Administrar Base de datos

Fuente: Elaboración Propia

No.	Nombre del campo	Clasificación	Valor Nulo	Descripción
1	Nombre BD	Campo de texto	No	Alfanumérico
2	Tag	Campo numérico	No	Solo números enteros
3	Nombre de Campo	Campo de texto	No	Alfanumérico
4	Tipo	Campo de selección	No	Se selecciona de una lista
5	Sub-Campo	Campo de texto	Si	Alfanumérico
6	Repetible	Campo de si/no	No	Si o No

Tabla 7: Descripción de las variables

Fuente: Elaboración propia.

A continuación, en la Tabla 8 se muestran los resultados de las pruebas de caja negra que fueron realizadas al módulo Adquisición, en las que se puede observar el número de iteraciones que se le aplicaron los casos de pruebas, el total de no conformidades encontradas y las que fueron resueltas, e igualmente se muestra la cantidad que no procedieron.

No. de iteraciones	Total de no conformidades	No procede	Resueltas
1ra	3	0	3
2da	1	0	1
3ra	0	0	0

Tabla 8: Descripción de las variables

Fuente: Elaboración propia

De esta forma quedaron resueltas todas las no conformidades que se detectaron durante las pruebas realizadas al sistema. Se consiguieron resultados satisfactorios y se obtuvo un sistema que responde a las especificaciones de calidad de software. Las no conformidades encontradas se nombran a continuación:

- El formato de los componentes "*Label*" no se encontraba estandarizado para la aplicación.
- El mensaje: "La Base de datos ya existe" no se muestra cuando se comete el error de repetir el nombre de una base de datos existente al crear una nueva.
- Falta de ortografía en varias interfaces.

Prueba de integración:

Al realizarse las pruebas de integración se obtuvieron los siguientes resultados:

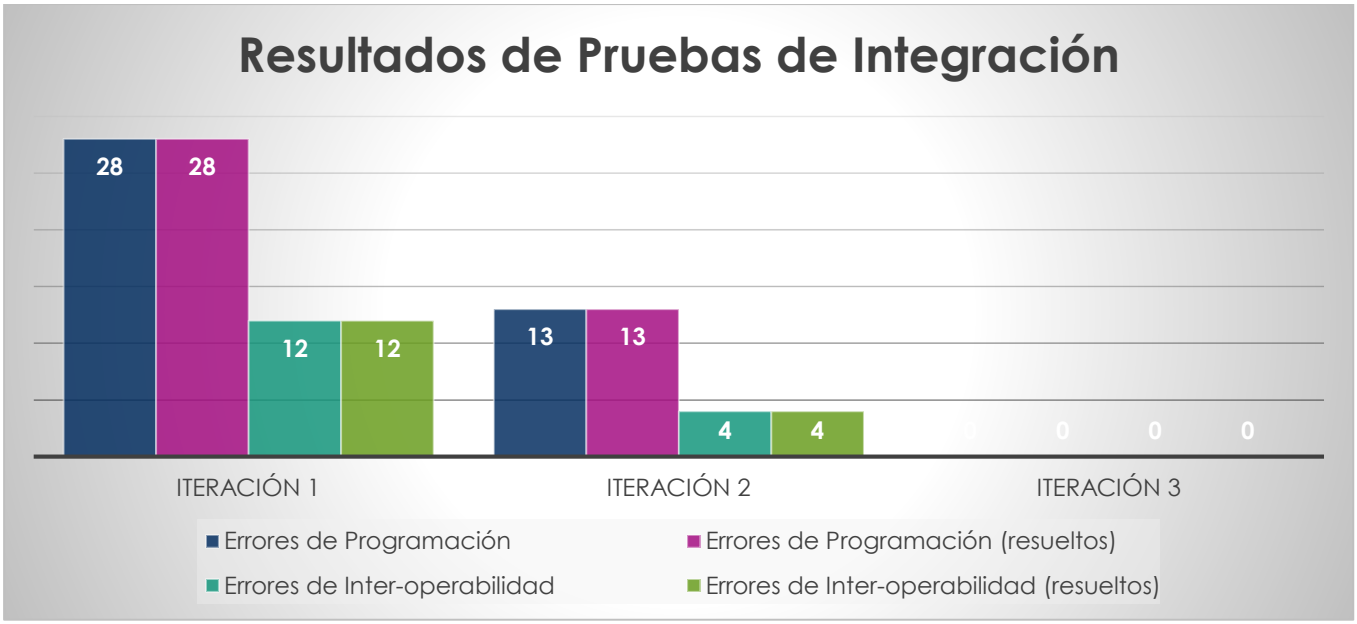


Figura 13: Gráfica de los resultados de las pruebas de integración realizadas.

Fuente: Elaboración propia.

Luego de realizada la tercera iteración de las pruebas de integración, se llegó a la conclusión de que el sistema se encuentra correctamente integrado. Esto es debido a que no existen errores de programación, ni errores de inter-operabilidad, permitiendo esto que los módulos interactúen correctamente como un todo.

Prueba de aceptación:

A continuación, en la figura se muestra el módulo aceptado por el cliente:

Acta de aceptación

ACTA DE ACEPTACIÓN

En cumplimiento del trabajo de diploma **Módulo para la Administración de bases de datos J-ISIS para el sistema ABCD v3.0** se hace entrega de los productos que se relacionan a continuación:

- *Código del módulo.*
- *Artefactos generados durante las fases como parte de la metodología AUP-UCI.*

El arquitecto del proyecto ABCD y el jefe de Departamento Desarrollo de Componentes, luego de haber revisado los productos de trabajo determina que se aceptan y que el mismo cumple con los requisitos definidos y con la calidad necesaria para ser integrada al proyecto.

Entrega	**Recibe**
Nombre y Apellidos: Wilmer Garcia Asin	**Nombre y Apellidos:** Luis Carlos Álvarez Fernández
Cargo: Autor del trabajo de diploma Firma	**Cargo:** Jefe de Departamento Desarrollo de Componentes Firma:
	Nombre y apellidos: Alberto A. Arias Benitez **Cargo:** Arquitecto de Proyecto ABCD. Firma

Fecha: 12 de junio de 2018

Figura 14: prueba de aceptación del módulo.

Fuente: Elaboración conjunta con el cliente.

3.5 Conclusiones parciales

En este capítulo se explicó a partir de los resultados del diseño, la implementación del módulo de Administración de bases de datos J-ISIS para el sistema ABCD v3.0. Mediante el diagrama de despliegue quedó definida la distribución física y lógica de la arquitectura del sistema y sus conexiones. Se definió además los estándares de codificación y las pruebas de software.

En las pruebas funcionales se encontraron en la primera iteración 3 no conformidades y se resolvieron las 3, en la segunda iteración se encontró 1no conformidad que fue resuelta, y en la tercera no se encontraron no conformidades.

En las pruebas de integración del módulo, se encontraron varios errores de programación y de inter-operabilidad en las primeras dos iteraciones, los cuales fueron resueltos en casi toda su totalidad y ya en la tercera iteración el módulo funcionaba sin ningún error.

Conclusiones

La realización de esta investigación permitió arribar a las siguientes conclusiones:

- A partir del análisis de los conceptos asociados a la administración de bases de datos de Sistemas de Gestión Bibliotecaria, y la caracterización de la administración de bases de datos con J-ISIS permitió el diseño de la propuesta de solución.
- Con el empleo de la metodología de desarrollo de software AUP-UCI más las herramientas Visual Paradigm versión 8.0, el ide de desarrollo Eclipse versión Kepler Release 4.3 y J-ISIS Suite versión de noviembre 2014 para la gestión de las bases de datos, se desarrolló el módulo de Administración de bases de datos J-ISIS para el sistema ABCD v3.0.
- Se realizaron pruebas funcionales y de integración al módulo obtenido donde se aplicó el método de caja negra, lo que facilitó la corrupción de las no conformidades identificadas.

Recomendaciones

La administración de bases de datos de Sistemas de Gestión Bibliotecaria hace posible que se desarrollen futuras investigaciones, es por ello que se recomienda:

- Continuar el desarrollo de este módulo, atendiendo principalmente a la interfaz gráfica para una mejor experiencia del usuario con el sistema ABCD v3.0.
- La implementación de otras funcionalidades como: eliminar base de datos, debido a que el Dataprovider no provee esta opción; clonar base de datos y clonar hojas de trabajo, para crear estructuras parecidas.

Referencias Bibliográficas

1. APPLEBY, D. y VANDEKOPPLE, J., 1998. Lenguajes de programación : paradigma y práctica. ,
2. ARRIOLA NAVARRETE, Ó. y BUTRÓN YÁÑEZ, K., 2008. Sistemas integrales para la automatización de bibliotecas basados en software libre. [en línea]. [Consulta: 2 mayo 2018]. Disponible en: http://bvs.sld.cu/revistas/aci/vol18_6_08/aci091208.htm.
3. ASCENCIO, G., SPINAK, E. y EGBERT DE SMET, 2016. Tabla de definición de campos - ABCDWIKI Español. [en línea]. [Consulta: 14 mayo 2018]. Disponible en: http://abcdwiki.net/wiki/es/index.php?title=Tabla_de_definición_de_campos.
4. BARZANALLANA, R., 2006. Apuntes. Lenguajes de programación, metodologias, lenguaje Pascal. Informática Aplicada al Trabajo Social 2004/05-6. Rafael Barzanallana. Universidad de Murcia. [en línea]. [Consulta: 3 mayo 2018]. Disponible en: http://www.um.es/docencia/barzana/IATS/Iats06.html.
5. CECHICH, A. y MOORE, R., 2000. Un modelo formal de patrones orientados a objetos. ,
6. CENTRO NACIONAL DE INFORMACIÓN DE CIENCIAS MÉDICAS., Y. y RÍOS HILARIO, A.B., 2005. *Acimed.* [en línea]. S.l.: 2000, Editorial Ciencias Médicas. [Consulta: 2 mayo 2018]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352005000400005.
7. CERVANTES, H., 2018. Arquitectura de Software | SG Buzz. [en línea]. [Consulta: 9 mayo 2018]. Disponible en: https://sg.com.mx/revista/27/arquitectura-software.
8. CORAL, G. y ALBERTO, L., 2011. Estudio comparativo de los Sistemas Integrados de código abierto para biblioteca: Koha y Phpmybibli. *Universidad Nacional Mayor de San Marcos. Programa Cybertesis PERÚ* [en línea], Disponible en: http://cybertesis.unmsm.edu.pe/handle/cybertesis/1746%5Cnhttp://cybertesis.unmsm.edu.pe/bitstream/cybertesis/1746/1/Gutierrez_cl.pdf.
9. Eclipse Kepler 4.3. [en línea], 2018. [Consulta: 10 mayo 2018]. Disponible en: http://www.eclipse.org/kepler/.
10. Eclipse Virgo. [en línea], 2018. Disponible en: https://www.osgi.org/eclipse-virgo/.
11. Equinox | The Eclipse Foundation. [en línea], 2018. Disponible en: http://www.eclipse.org/equinox/.
12. FRANCO GARCÍA, A., 2000. La Máquina Virtual Java. [en línea]. [Consulta: 10 mayo 2018]. Disponible en: http://www.sc.ehu.es/sbweb/fisica/cursoJava/fundamentos/introduccion/virtual.htm.
13. GONZAGA, A., CEDILLO, Á., ANTONIO, J. y MEJÍA, G., 2006. Arquitecturas en n-Capas : Un

Sistema Adaptivo. *Polibits* [en línea], vol. 34, no. 1870-9044, pp. 34-37. DOI http://dx.doi.org/10.17562/PB-34-7. Disponible en: http://www.redalyc.org/articulo.oa?id=402640447007.

14. GONZÁLEZ CORNEJO, J.E., 2008. El Lenguaje de Modelado Unificado (UML). [en línea]. [Consulta: 10 mayo 2018]. Disponible en: http://www.docirs.com/uml.htm.
15. GONZÁLEZ JORRIN, M., 2007. Tesis de maestría. Proceso de pruebas para la liberación de productos software. ,
16. JACOBSON, I., BOOCH, G. y RUMBAUGH, J., 2000. El Proceso Unificado De Desarrollo de Software. ,
17. LOPEZ, E., 2018. Arquitectura de n capas. [en línea], Disponible en: https://www.academia.edu/10102692/Arquitectura_de_n_capas.
18. LUJÁN, M., 2012. Sistemas bibliotecarios con base de datos isis softwares de biblioteca. [en línea]. [Consulta: 2 mayo 2018]. Disponible en: https://es.slideshare.net/terrydulce/sistemas-bibliotecarios-con-base-de-datos-isis-softwares-de-biblioteca.
19. MOCHORRO REYES, R.A., 2004. Los Patrones como un Medio del Diseño Orientado a Objetos. [en línea], Disponible en: www.repositoriodigital.ipn.mx/bitstream/123456789/5322/4/35-4.pdf.
20. NOVELO CAN, C. y OLIVERA SOSA, Á.G., 2010. Reporte de instalación de apache. ,
21. OCAÑA BUENO, L., 2017. Gestor de Base de Datos. [en línea]. [Consulta: 10 mayo 2018]. Disponible en: https://es.slideshare.net/LisbethOcaaBueno/gestor-de-base-de-datos-80347312.
22. *Patrones de Diseño* [en línea], 2015. Departamento de Informática Universidad Técnica Federico Santa María: s.n. Disponible en: http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf%0A.
23. PRESSMAN, R.S., 2010. *INGENIERÍA DEL SOFTWARE. UN ENFOQUE PRÁCTICO.* Séptima ed. S.l.: s.n. ISBN 978- 607-15-0314-5.
24. Pruebas de Software. [en línea], 2016. Disponible en: http://www.pruebasdesoftware.com/.
25. RODRÍGUEZ, T., 2014. Metodología de desarrollo para la Actividad productiva de la UCI. , pp. 1-16.
26. SARRIA, F.A., 2006. Sistemas de Información Geográfica. *Universidad de Murcia* [en línea], pp. 239. Disponible en: http://www.um.es/geograf/sigmur/sigpdf/temario.pdf.
27. SMET, E. de y SPINAK, E., 2009. El abc del ABCD : Manual del módulo Central. ,
28. Spring Dynamic Modules Reference Guide. [en línea], 2017. [Consulta: 24 mayo 2018]. Disponible en: https://docs.spring.io/spring-osgi/docs/current/reference/html/.
29. SYSTEMS PTY LTD, S., 2018. Diagrama de Despliegue UML. [en línea]. Disponible en:

http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.

30. UNESCO, 2008. Manual de Gestion Documental. , pp. 44.
31. UNESCO y HERNÁNDEZ, A., 2003. *Winisis Manual de Referencia (Versión 1.5)*. 2003. S.l.: s.n.

Bibliografía

1. GONZÁLEZ JORRIN, M., 2007. Tesis de maestría. Proceso de pruebas para la liberación de productos software. ,
2. PRESSMAN, R.S., 2010. *INGENIERÍA DEL SOFTWARE. UN ENFOQUE PRÁCTICO*. Séptima ed. S.l.: s.n. ISBN 978- 607-15-0314-5.
3. Equinox | The Eclipse Foundation. [en línea], 2018. Disponible en: http://www.eclipse.org/equinox/.
4. Pruebas de Software. [en línea], 2016. Disponible en: http://www.pruebasdesoftware.com/.
5. Eclipse Virgo. [en línea], 2018. Disponible en: https://www.osgi.org/eclipse-virgo/.
6. Spring Dynamic Modules Reference Guide. [en línea], 2017. [Consulta: 24 mayo 2018]. Disponible en: https://docs.spring.io/spring-osgi/docs/current/reference/html/.
7. CECHICH, A. y MOORE, R., 2000. Un modelo formal de patrones orientados a objetos. ,
8. MOCHORRO REYES, R.A., 2004. Los Patrones como un Medio del Diseño Orientado a Objetos. [en línea], Disponible en: www.repositoriodigital.ipn.mx/bitstream/123456789/5322/4/35-4.pdf.
9. *Patrones de Diseño* [en línea], 2015. Departamento de Informática Universidad Técnica Federico Santa María: s.n. Disponible en: http://www.inf.utfsm.cl/~visconti/ili236/Documentos/08-Patrones.pdf%0A.
10. SYSTEMS PTY LTD, S., 2018. Diagrama de Despliegue UML. [en línea]. Disponible en: http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.
11. JACOBSON, I., BOOCH, G. y RUMBAUGH, J., 2000. El Proceso Unificado De Desarrollo de Software. ,
12. NOVELO CAN, C. y OLIVERA SOSA, Á.G., 2010. Reporte de instalación de apache. ,
13. UNESCO y HERNÁNDEZ, A., 2003. *Winisis Manual de Referencia (Versión 1.5)*. 2003. S.l.: s.n.
14. Indexación - EcuRed. [en línea], [sin fecha]. [Consulta: 14 mayo 2018]. Disponible en: https://www.ecured.cu/Indexación.
15. ASCENCIO, G., SPINAK, E. y EGBERT DE SMET, 2016. ABCDWIKI Español. [en línea]. [Consulta: 14 mayo 2018]. Disponible en: http://abcdwiki.net/wiki/es/index.php?title=Página_principal.

16. ASCENCIO, G., SPINAK, E. y EGBERT DE SMET, 2016. Tabla de definición de campos - ABCDWIKI Español. [en línea]. [Consulta: 14 mayo 2018]. Disponible en: http://abcdwiki.net/wiki/es/index.php?title=Tabla_de_definición_de_campos.
17. PÉREZ, E., 2018. Sistemas Gestores de Bases de Datos. [en línea]. [Consulta: 10 mayo 2018]. Disponible en: https://emiliopm.com/podcast/03-sistemas-gestores-bases-datos/.
18. OCAÑA BUENO, L., 2017. Gestor de Base de Datos. [en línea]. [Consulta: 10 mayo 2018]. Disponible en: https://es.slideshare.net/LisbethOcaaBueno/gestor-de-base-de-datos-80347312.
19. Eclipse Kepler 4.3. [en línea], 2018. [Consulta: 10 mayo 2018]. Disponible en: http://www.eclipse.org/kepler/.
20. FRANCO GARCÍA, A., 2000. La Máquina Virtual Java. [en línea]. [Consulta: 10 mayo 2018]. Disponible en: http://www.sc.ehu.es/sbweb/fisica/cursoJava/fundamentos/introduccion/virtual.htm.
21. GONZÁLEZ CORNEJO, J.E., 2008. El Lenguaje de Modelado Unificado (UML). [en línea]. [Consulta: 10 mayo 2018]. Disponible en: http://www.docirs.com/uml.htm.
22. MONTES DE OCA HERNÁNDEZ, Y. y BRITO DÍAZ, Y., [sin fecha]. Módulo para la Gestión de Información de Trámites Protocolizables Complejos en la notaría Buen Viaje de Santa Clara. Yanirys Montes de Oca Hernández y Yuliesky Brito Díaz. [en línea], pp. 100. Disponible en: http://www.eumed.net/libros-gratis/2012b/1232/arquitectura-N-capas.html.
23. LOPEZ, E., 2018. Arquitectura de n capas. [en línea], Disponible en: https://www.academia.edu/10102692/Arquitectura_de_n_capas.
24. BARRAZA, F., 2013. Modelado y Diseño de Arquitectura de Software. , pp. 39.
25. GONZAGA, A., CEDILLO, Á., ANTONIO, J. y MEJÍA, G., 2006. Arquitecturas en n-Capas: Un Sistema Adaptivo. *Polibits* [en línea], vol. 34, no. 1870-9044, pp. 34-37. DOI http://dx.doi.org/10.17562/PB-34-7. Disponible en: http://www.redalyc.org/articulo.oa?id=402640447007.
26. TAFUR DELGADILLO, C.A., 2017. Implementación de un software web CRUD bajo una base de datos no relacional. [en línea], pp. 58. Disponible en: repository.udistrital.edu.co/bitstream/11349/7284/1/TafurDelgadilloCarlosAndrés2017.pdf.
27. *Requisitos Funcionales de los Registros Bibliográficos: informe final*, 2004. 2004. Madrid: Ministerio de Cultura, Secretaria General Técnica: s.n.

28. ESCOLAR, H., 1990. Historia de las bibliotecas. , pp. 83.
29. BURNS, A., 2003. Sistemas de tiempo real y lenguajes de programación. ,
30. BIONDI, J. y CLAVEL, G., 1985. Algorítmica y lenguajes. ,
31. APPLEBY, D. y VANDEKOPPLE, J., 1998. Lenguajes de programación: paradigma y práctica. ,
32. CERVANTES, H., 2018. Arquitectura de Software | SG Buzz. [en línea]. [Consulta: 9 mayo 2018]. Disponible en: https://sg.com.mx/revista/27/arquitectura-software.
33. RODRÍGUEZ, T., 2014. Metodología de desarrollo para la Actividad productiva de la UCI. , pp. 1-16.
34. SOMMERVILLE, I., 2010. *Software Engineering*. S.l.: s.n. ISBN 9780137035151.
35. CORAL, G. y ALBERTO, L., 2011. Estudio comparativo de los Sistemas Integrados de código abierto para biblioteca: Koha y Phpmybibli. *Universidad Nacional Mayor de San Marcos. Programa Cybertesis PERÚ* [en línea], Disponible en: http://cybertesis.unmsm.edu.pe/handle/cybertesis/1746%5Cnhttp://cybertesis.unmsm.edu.pe/bitstream/cybertesis/1746/1/Gutierrez_cl.pdf.
36. SARRIA, F.A., 2006. Sistemas de Información Geográfica. *Universidad de Murcia* [en línea], pp. 239. Disponible en: http://www.um.es/geograf/sigmur/sigpdf/temario.pdf.
37. BARZANALLANA, R., 2006. Apuntes. Lenguajes de programación, metodologías, lenguaje Pascal. Informática Aplicada al Trabajo Social 2004/05-6. Rafael Barzanallana. Universidad de Murcia. [en línea]. [Consulta: 3 mayo 2018]. Disponible en: http://www.um.es/docencia/barzana/IATS/Iats06.html.
38. LUJÁN, M., 2012. Sistemas bibliotecarios con base de datos isis softwares de biblioteca. [en línea]. [Consulta: 2 mayo 2018]. Disponible en: https://es.slideshare.net/terrydulce/sistemas-bibliotecarios-con-base-de-datos-isis-softwares-de-biblioteca.
39. ARRIOLA NAVARRETE, Ó. y BUTRÓN YÁÑEZ, K., 2008. Sistemas integrales para la automatización de bibliotecas basados en software libre. [en línea]. [Consulta: 2 mayo 2018]. Disponible en: http://bvs.sld.cu/revistas/aci/vol18_6_08/aci091208.htm.
40. UNESCO, 2008. Manual de Gestión Documental. , pp. 44.
41. CENTRO NACIONAL DE INFORMACIÓN DE CIENCIAS MÉDICAS., Y. y RÍOS HILARIO, A.B., 2005. *Acimed.* [en línea]. S.l.: 2000, Editorial Ciencias Médicas. [Consulta: 2 mayo 2018].

Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352005000400005.

42. SMET, E. de y SPINAK, E., 2009. El abc del ABCD : Manual del módulo Central. ,