Universidad de las Ciencias Informáticas

Facultad 2

“Reimplementación del catálogo en línea del sistema ABCD 3.0”

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Miguel Yasel Morales García

Tutores: Ing. Alberto Alejandro Arias Benítez

Ing. Leandro Tabares Martín

La Habana, 2018

Dedicatoria

Dedico el presente trabajo de diploma a mi madre Nancy, y a mis padres, Miguelito y Gabriel por ser pilares importantes en mi vida, estar siempre a mi lado y brindarme su apoyo y sus consejos, para hacer de mí una mejor persona.

A toda mi familia, y a mis amigos, a todos los que creyeron en mí.

Declaración de autoría

Se declara que Miguel Yasel Morales García es el único autor del presente Trabajo de Diploma que tiene

por título *"Reimplementación del módulo OPAC del sistema ABCD 3.0"*, se concede a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo con carácter exclusivo.

Para que así conste firmo la presente a los __ días del mes de _____ del año ____.

Autor:

_____

Miguel Yasel Morales García

Tutores:

_____ _____

Ing. Alberto Alejandro Arias Benítez Ing. Leandro Tabares Martín

Resumen

La gestión de la información es de vital importancia en la sociedad actual del conocimiento, donde a través del uso de las tecnologías el hombre ha puesto a disposición de la sociedad disímiles recursos informativos. En la Universidad de las Ciencias Informáticas se desarrolla el Sistema Integrado de Gestión Bibliotecaria (SIGB) ABCD, este sistema ofrece herramientas para la recuperación de la información de bases de datos, control de estadísticas, circulación, catalogación, control de publicaciones periódicas, entre otras. La investigación actual contribuye a la recuperación de los registros bibliográficos almacenados por ABCD en su versión 3.0. Para ello se plantea como objetivo general desarrollar un catálogo en línea para el sistema ABCD 3.0 con un diseño adaptativo que permita la búsqueda facetada. La investigación consta de tres capítulos que abarcan desde la teoría, el análisis y diseño, y la implementación. Finalmente se pude constatar el cumplimiento del objetivo planteado a partir de las pruebas realizadas a la aplicación.

Palabras claves: búsqueda facetada, información, SIGB, tecnologías

Índice

Introducción 1
Capítulo 1: Fundamentos teóricos 6
Introducción 6
1.1. Conceptualización 6
1.1.1. Surgimiento de los SIGB 7
1.1.2. Características de los SIGB 8
1.1.3. Ventajas del uso de los SIGB 9
1.1.4. Módulos básicos de un SIGB 9
1.2. Los OPAC en los sistemas de gestión bibliotecaria 11
1.2.1. Proceso de recuperación de registros bibliográficos 12
1.2.2. SIGB existentes, análisis del OPAC 14
1.3. Metodología, herramientas y lenguajes utilizados 23
1.3.1. Metodología 23
1.3.2. Herramientas 26
1.3.3. Lenguajes 28
1.4. Conclusiones parciales 29
Capítulo 2: Análisis y diseño de la solución 30
Introducción 30
2.1. Modelo de Dominio 30
2.2. Especificación de los requisitos de software 31
2.2.1. Requisitos funcionales 31
2.2.2. Requisitos no funcionales 33
2.3. Casos de usos del sistema 34
2.3.1 Matriz de trazabilidad 36
2.3.1 Descripción de Casos de Usos 37
2.4 Arquitectura del software 45
2.5 Patrones de diseño 48
2.6 Conclusiones parciales 49

Capítulo 3: Implementación y pruebas 50
Introducción 50
3.2.1 Declaración de variables y constantes 51
3.2.2 Métodos 51
3.2.3 Clases e interfaces 52
3.3.1 Estrategia de prueba 53
3.3.2 Niveles de prueba 53
3.3.3 Tipos de pruebas 54
3.3.4 Métodos de pruebas 55
3.3.5 Diseño de casos de prueba 56
3.3.6 Resultados de las pruebas 57
3.4 Conclusiones del capítulo 59
Conclusiones 60
Referencias Bibliográficas 61
Bibliografía 65
Anexos 69

Índice de tablas

Tabla 1. Caracterización de los módulos OPAC de los SIGB. 22
Tabla 2: Matriz de trazabilidad Requisitos_Casos de Uso 36
Tabla 3: CU_1 Autenticar usuario. 37
Tabla 4: CU_2 Consultar Catálogo 38
Tabla 5: CU_3 Consultar Catálogo (Avanzado) 39
Tabla 6: CU_4 Gestionar Selección 41
Tabla 7: CU_8 Gestionar lista de selección 42
Tabla 8: CU_10 Administrar Listas 44
Tabla 9. Caso de Prueba Autenticar usuario. 56
Tabla 10. Caso de Prueba Consultar catálogo 57
Tabla 11. Descripción de las variables de entrada 57
Tabla 12. Resultados de las pruebas de funcionalidad 58
Tabla 13: Matriz de trazabilidad Requisitos_Casos de Prueba 70
Tabla 14: Matriz de trazabilidad Requisitos_Requisitos 71

Índices de Figuras

Figura 1 Modelo de dominio. Fuente: Elaboración propia 30
Figura 2 Arquitectura. Fuente: Elaboración propia 46
Figura 3 Diagrama de despliegue. Fuente: Elaboración propia 50

Introducción

En todo momento ha sido vital para todo grupo de personas, acumular información. Los hombres primitivos lo hacían a través de la pintura rupestre, actividad donde tallaban en las paredes de las cavernas sus más disímiles experiencias y predicciones (Mason 2006). Asimismo, todas las sociedades hasta la actualidad, han tenido la imperiosa necesidad de hacer persistir la información, puesto que es un elemento fundamental para el desarrollo.

En la era de la información, "*uno de los principales problemas es su exceso*" (Aja 2012), por lo que necesita de mejores herramientas de recopilación y socialización de la misma, en aras de lograr una mayor dinámica, surge también la necesidad de crear un espacio donde se pudiera resguardar toda la información como patrimonio del conocimiento humano. Este espacio tomó como nombre biblioteca. Según el diccionario de la Real Academia Española (RAE) se puede definir el término biblioteca como una "institución cuya finalidad consiste en la adquisición, conservación, estudio y exposición de libros y documentos. Lugar donde se tiene considerable número de libros ordenados para la lectura" (RAE 2017).

Las bibliotecas constituyen una de las formas de almacenamiento de información más antiguas utilizadas por el hombre, estas, tradicionalmente, han solventado las carencias del cerebro humano en el proceso de acopio y conservación de datos, cristalizados en documentos. *"Conjuntamente con los centros de documentación se han convertido en uno de los organismos fundamentales en la difusión de los documentos y la información, siendo además centros de recursos para el aprendizaje, la docencia y la investigación"* (Brown y Arias 2015).

"Las bibliotecas se consideran puertas de acceso a los conocimientos, a la cultura y desempeñan una función fundamental en la sociedad. Los recursos y los servicios que ofrecen dan la oportunidad de aprender, sirven como apoyo a la alfabetización, a la educación, y ayudan a dar forma a las nuevas ideas y perspectivas que son vitales dentro de una sociedad creativa e innovadora" (White 2012). Asimismo, garantizan la existencia de un registro auténtico de los conocimientos creados y acumulados por las generaciones pasadas. Gracias a las bibliotecas es más fácil avanzar en la investigación y los conocimientos humanos, preservar los conocimientos acumulados y el patrimonio cultural para generaciones futuras.

Con la aparición de las nuevas tecnologías de la información y las comunicaciones (TIC) en el contexto bibliotecario, se ha producido en los últimos tiempos una verdadera revolución digital que impacta en todos los ámbitos de la sociedad, lo que trae consigo el surgimiento de nuevas formas de trabajo y nuevos servicios como bibliotecas digitales, consultas en línea, así como la informatización de los procesos que se realizan en las bibliotecas.

Es de gran ayuda auxiliarse de las TIC para gestionar procesos en las organizaciones, procesar grandes volúmenes de datos que estas manejan. Así surgen los Sistemas Integrados de Gestión Bibliotecaria (SIGB), que "*son herramientas tecnológicas que permite automatizar las operaciones bibliotecarias más comunes. Típicamente abarca la catalogación, circulación, consulta y adquisición de materiales*" (Martín 2008). La evolución de los SIGB va conjuntamente con la rápida evolución de la tecnología y ha ido adaptándose a contextos de información cada vez más amplios, a usuarios cada vez más exigentes y a entornos de trabajo cada vez más tecnificados. *"La irrupción de Internet a principios de los años 90 marcó un antes y un después en el desarrollo tecnológico y de procesos en las bibliotecas"* (Hernández y García 2002).

Son muchos los SIGB destinados a informatizar los entornos bibliotecarios, los cuales se pueden aplicar a las funciones y servicios, internos o públicos. Uno de los módulos de un SIGB son los OPAC (del inglés *Online Public Access Catalog*), también conocidos como catálogos en línea, los cuales tienen funciones de búsqueda y recuperación de gran desarrollo, lo que dinamiza y agiliza el acceso del público a la hora de realizar las búsquedas (Brown y Arias 2015).

En la Universidad de Ciencias Informáticas (UCI), se encuentra el Centro de Informatización de la Gestión Documental (CIGED). El sistema de Automatización de Bibliotecas y Centros de Documentación (ABCD) surge como parte de un proyecto de colaboración entre varias universidades, el mismo fue asignado al centro CIGED, el cual es el encargado de trabajar en el desarrollo. El sistema ABCD ofrece herramientas para la recuperación de la información de bases de datos, control de estadísticas, circulación, catalogación, control de publicaciones periódicas, funciones de búsqueda, entre otras funcionalidades. Este sistema posee además un catálogo el cual presenta funcionalidades de búsqueda y recuperación de información.

El continuo avance de la tecnología ha propiciado la proliferación de dispositivos electrónicos, así como sus capacidades de conectividad, mediante los cuales es posible acceder a internet, hecho que los sistemas

informáticos contemporáneos deben tener en consideración debido a la gran variedad de prestaciones: la resolución de pantalla, capacidad de la memoria RAM y almacenamiento, la frecuencia del procesador, etc. que poseen dichos dispositivos. Es por este motivo que reviste una importancia significativa contar con un catálogo acorde a las especificaciones actuales, el cual deberá tener un diseño adaptativo, que brinde facilidades de interacción con el usuario, que incorpore buenas prácticas de experiencia de usuario como funcionabilidad, navegabilidad, usabilidad, confiabilidad, eficiencia y mantenibilidad.

El catálogo en línea del sistema ABCD 3.0 presenta varias dificultades, las cuales se mencionan a continuación.
El uso de dispositivos móviles ha aumentado notablemente en los últimos años, en particular el uso de tabletas y teléfonos inteligentes, y con ello se ha popularizado la navegación en internet mediante estos dispositivos, por lo que es necesario adaptar la resolución del módulo al tamaño de la pantalla del dispositivo.

El catálogo en línea del sistema ABCD 3.0 está implementado con RAP[1], la tecnología con la cual dificulta el soporte para las plataformas móviles, ya que habría que crear un tema personalizado para adaptarse a pantallas más pequeñas, al desplazarse por áreas más grandes deberán utilizarse las barras de desplazamiento, pero tendrían que ser los suficiente anchas como para moverlas con los dedos ya que el desplazamiento táctil no está habilitado.

El catálogo del sistema ABCD 3.0 no aplica la técnica de búsqueda por facetas, le permite al usuario ingresar un término de búsqueda y luego refinar los resultados de acuerdo con distintos grupos de filtros, se podrá acceder a la información de forma organizada, mejorando la velocidad y calidad con la que se muestran los resultados.

A partir de la situación planteada anteriormente, se define como **problema a resolver**: ¿Cómo contribuir a la recuperación de los registros bibliográficos almacenados por el Sistema Automatización de Bibliotecas y Centros de Documentación en su versión 3.0?

[1] *Remote Aplication Plattaform* (RAP): Plataforma de Aplicaciones Remotas, es un framework para aplicaciones modulares que se integra con tecnologías de Eclipse.

El **objeto de estudio** lo constituye, por tanto, el proceso de recuperación de los registros bibliográficos.

Para brindarle una solución efectiva al problema en cuestión, se plantea como **objetivo general** desarrollar un catálogo en línea para el sistema ABCD 3.0 con un diseño adaptativo que permita la búsqueda facetada.

Como **campo de acción**, los catálogos en línea de los Sistemas Integrados de Gestión Bibliotecaria.

Para alcanzar el objetivo general se plantean los siguientes **objetivos específicos**:

- Definir los presupuestos teóricos que sustentan la implementación de un catálogo en línea para el sistema ABCD 3.0.
- Realizar el análisis y diseño del catálogo en línea para el sistema ABCD 3.0 a partir de la metodología definida.
- Implementar el catálogo en línea para el sistema ABCD 3.0.
- Probar el funcionamiento del catálogo en línea para detectar posibles errores.

Métodos para la búsqueda y procesamiento de la información:

Los métodos **teóricos** permiten revelar las relaciones esenciales del objeto de investigación, no perceptibles de manera inmediata. Los métodos aplicados en esta investigación se describen a continuación:

- Análisis documental: Para la revisión bibliográfica, la revisión de las fuentes básicas de información, el estudio de documentos, entre otros.
- Analítico-sintético: Al descomponer el problema de investigación en elementos por separado, propició el proceso de construcción del marco teórico de la investigación.
- Modelación: Para lograr una abstracción del objeto de estudio con vista a explicar la realidad de manera simplificada. Específicamente se utilizó el modelo analógico, el cual permite, a través de un diagrama o esquema, reflejar la estructura de relaciones y determinadas propiedades de la realidad del objeto.

Los métodos **empíricos** revelan y explican las características fenomenológicas del objeto. Aportan la información que no se recoge en las memorias escritas. La utilizada en esta investigación es la entrevista.

- Entrevista: Para dialogar con especialistas y trabajadores del centro CIGED y recoger la información necesaria que tributa a la refactorización del módulo OPAC de ABCD 3.0**.** Esta entrevista permitió conocer el estado actual de los procesos que en este centro se llevan a cabo. (Ver Anexo 1)

La investigación consta de tres capítulos:

Capítulo 1: Fundamentos teóricos. En este capítulo se abordan los principales presupuestos teóricos asociados a la investigación, se realiza un estudio de varios SIGB utilizados en Cuba y el mundo, se realiza una caracterización del catálogo en línea actual y sus principales deficiencias. Se describen la metodología, herramientas y lenguajes para implementar la propuesta de solución.

Capítulo 2: Análisis y diseño de la solución. En este capítulo se hace un levantamiento de los requisitos funcionales y no funcionales necesarios para el correcto funcionamiento del sistema. Se generan los artefactos según la metodología.

Capítulo 3: Implementación y pruebas. En este capítulo se muestra el modelo de implementación como resultado del diseño anteriormente desarrollado. Se describen las pruebas a realizar, con el objetivo de probar el correcto funcionamiento del catálogo en línea, así como los resultados de las mismas.

Capítulo 1: Fundamentos teóricos

Introducción

En este capítulo se abordan los conceptos y definiciones que están relacionados con el dominio del problema, en aras de lograr una mejor comprensión de la investigación; también se presentan las tecnologías y herramientas empleadas en el proceso de desarrollo, además de la fundamentación de la metodología que guía el proceso.

1.1. Conceptualización

Para una mayor comprensión de la presente investigación, es importante definir algunos conceptos fundamentales vinculados con el objeto de estudio de la investigación.

"La información es un recurso que es vital para las organizaciones" (Rivero y Galarza 2015) por lo que es necesario gestionarlo eficazmente pues de ello depende que puedan tomarse decisiones certeras. Al aumentar considerablemente el volumen de información en la actualizad, las TIC constituyen medios eficaces para el mejoramiento de la gestión de la misma en las organizaciones.

Como afirman Vidal y Araña la gestión de la información no es más que el proceso de organizar, evaluar, presentar, comparar los datos en un determinado contexto, controlando su calidad, de manera que esta sea veraz, oportuna, significativa, exacta y útil y que esta información esté disponible en el momento que se le necesite. Ella se encamina al manejo de la información, documentos, metodologías, informes, publicaciones, soportes y flujos en función de los objetivos estratégicos de una organización (2012).

Torres[2] define también la gestión de la información como el proceso mediante el cual se obtienen, despliegan o utilizan recursos básicos (económicos, físicos, humanos, materiales) para mejorar información dentro y para la sociedad. Tiene como elemento básico, la gestión del ciclo de vida de este recurso y se desarrolla en cualquier organización (2015).

[2] Lic. Levin Torres Lebrato. Licenciada en Gestión de Información en Salud. Profesor Asistente. Centro Provincial de Información de Ciencias Médicas. Camagüey, Cuba.

A partir del análisis de los conceptos de gestión de información anteriormente expuestos, el autor, define como gestión de información al proceso donde se obtiene, almacena, organiza, evalúa y distribuye la información para apoyar la toma de decisiones, el control, análisis y visión en una institución.

La gestión de la información está incluida en los SIGB, son varios los autores que se han referido en esta dirección:
García Melero y García Camarero[3] definen los SIGB como *"un conjunto organizado de recursos humanos 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 las bibliotecas: 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"* (1999).

Según Ariola y Butrón los SIGB son un *"conjunto de módulos de aplicación integrados en un solo programa y que comparten una base de datos bibliográfica común"* (2008). De acuerdo con Flores son *"un grupo de programas informáticos (módulos) interrelacionados que automatizan múltiples operaciones y funciones bibliotecarias basados en datos centralizados e intercambiables; esto con el objetivo de facilitar la gestión de las actividades llevadas a cabo en la biblioteca"* (2011).

Luego de analizar las definiciones anteriores, el autor define los SIGB como un sistema informático multifuncional, que integra los módulos básicos para la gestión bibliotecaria, donde cada módulo se encargaría de un proceso bibliotecario, estos comparten una base de datos bibliográfica común lo que evita la redundancia de los datos y aumenta la eficacia de las funciones y servicios bibliotecarios.

1.1.1. Surgimiento de los SIGB

En las décadas de los años 60 y 70 no existían soportes tecnológicos suficientemente avanzados para implementar los SIGB en las organizaciones, esto hizo que se implementasen en diferentes organizaciones subsistemas informatizados aislados.

[3] Matemático, informático, bibliotecólogo e historiador de la ciencia. Presidente del II Congreso Internacional sobre Bases de Datos en Humanidades y Ciencias Sociales. Ha desarrollado el sistema de automatización de la Biblioteca Nacional

Los primeros Sistemas Integrados de Gestión Bibliotecaria fueron pensados de forma modular: solamente se implementaban en la biblioteca aquellos módulos más necesarios, y éstos eran, casi obligatoriamente los de catalogación y circulación (Arriola y Montes de Oca 2014).

La evolución, no solo de las aplicaciones, sino del trabajo bibliotecario, hacía necesaria la incorporación de nuevas utilidades y herramientas, que fueran más allá que el propio control bibliográfico, de los fondos y de los préstamos realizados.

Tras una larga historia de la informatización de las bibliotecas, iniciada en la Biblioteca del Congreso de los Estados Unidos, y en algunas universidades como Harvard y Stanford dieron paso a un producto más completo, que serían los actuales SIGB, y que forman parte del núcleo de las operaciones bibliotecarias.

Existen numerosos SIGB en la actualidad, los sistemas privativos tienen costos elevados y muchas veces se dificulta la aplicación de estos en las bibliotecas, también se encuentran los SIGB basados en *software* libre, una muy buena opción para las bibliotecas que cuentan con bajos presupuestos y que al igual que los privativos facilitan la gestión de las bibliotecas a través de sus módulos

1.1.2. Características de los SIGB

Independientemente de que se utilice un *software* propietario o *software* libre, éste debe tener ciertas características que garanticen que es confiable su uso. *The Parliamentary Office of Science and Technology*[4] menciona algunas características que deben tener (2005):

- Fiabilidad: se define como el tiempo que un sistema puede permanecer en operación sin intervención del usuario.
- Calidad: comúnmente se define como el número de errores en un número fijo de líneas de código.
- Seguridad: lo resistente que es el *software* para no autorizar acciones fuera de protocolos, como virus.
- Flexibilidad: la facilidad con que el software puede ser personalizado para satisfacer las necesidades específicas y que se pueden ejecutar en diferentes tipos de dispositivos.

[4] La Oficina Parlamentaria de Ciencia y Tecnología (POST) es la fuente interna del Parlamento del Reino Unido para el análisis independiente, equilibrado y accesible de cuestiones de política pública relacionadas con la ciencia y la tecnología.

- Usabilidad: la facilidad de uso del software para con los usuarios.
- Confiabilidad: garantiza que el software sea de calidad, que funcione correctamente.
- Se ajustan a las normas y estándares internacionales de gestión bibliográfica como Marc21, Unimarc, el uso de protocolos como Z39.50 entre otros.
- Capacidad para ejecutarse en distintos sistemas operativos.
- Arquitectura cliente-servidor

1.1.3. Ventajas del uso de los SIGB

Con la implementación de un SIGB en una biblioteca estos ofrecen numerosas ventajas entre las cuales están:

- ✓ No redundancia de datos e información: cada dato o información existirá una única vez, y podrán ser actualizados o consultados de acuerdo con los criterios de accesibilidad y confidencialidad vigentes en la organización.
- ✓ Consistencia de información: al no existir redundancia en los datos, la coherencia del sistema será total, se puede tener la certeza que los datos sean correctos.
- ✓ Toda la documentación de sus módulos se almacena en una misma base de datos.
- ✓ Los centros de información serán más eficaces, con capacidad de agilizar los procesos y servicios bibliotecarios.

1.1.4. Módulos básicos de un SIGB

Los módulos de los SIGB modernos ayudan en las tres grandes etapas de todo proceso documental: entrada, tratamiento, y difusión de la información. Los más frecuentes son (Córdoba 2015):

Administración del sistema: se trata de un módulo que permite a los bibliotecarios administradores del sistema configurar las distintas opciones y adaptarlo a las necesidades de la organización. Desde este módulo se llevan a cabo tareas como son el control del funcionamiento general del sistema; la incorporación de una biblioteca o red de bibliotecas; administración y control de solicitudes de adquisición; la recepción de los materiales adquiridos y el control de devoluciones, entre otras.

Catalogación: Es el módulo por excelencia y también el más complejo de todos, pues de él depende el buen funcionamiento del catálogo. Su manejo implica utilizar la mayoría de los estándares oficiales.

Entre sus principales funciones, debe permitir la catalogación de todo tipo de materiales (monografías, publicaciones seriadas, grabaciones sonoras, vídeos), la producción ilimitada de fichas, tejuelos, códigos de barras, etc., la visualización de registros desde diversos puestos de trabajo, el intercambio de datos con otras bibliotecas y la generación de ficheros de autoridad para homogeneizar los encabezamientos, aunque algunos SIGB cuentan con un módulo propio e individualizado de control de autoridades para validar y normalizar los puntos de acceso al catálogo.

Circulación: Facilita el seguimiento de los documentos que circulan entre las unidades del servicio y el exterior, gestionando préstamos, renovaciones, devoluciones y reservas. Además, permite generar carnés de usuarios, avisos sobre vencimientos, cartas de reclamación por documentos no devueltos, informes sobre materiales perdidos o estropeados, estadísticas, etc. Resulta especialmente útil en la tramitación del préstamo interbibliotecario, por su mayor complejidad.

Adquisiciones: este módulo ayuda a gestionar la adquisición de nuevos documentos en la biblioteca, y el proceso básico sobre el que funciona este subsistema es la realización de pedidos a los proveedores. Este módulo utiliza información bibliográfica existente en el catálogo para realizar los pedidos, o permite la incorporación en el catálogo de descripciones suficientes para realizar el pedido. El módulo permite además gestionar los proveedores y los fondos presupuestarios destinados a la adquisición, lanzar reclamaciones de pedidos no recibidos o cancelar peticiones, además de gestionar desideratas[5].

Control de publicaciones seriadas: este módulo está pensado esencialmente para el control de la recepción de este tipo de publicaciones, dando una información lo más precisa posible de las existencias que dispone la biblioteca y controlando mejor el desarrollo de la colección. Utiliza la información bibliográfica del catálogo y comparte ciertas funciones del módulo de adquisiciones (gestión de proveedores, fondos presupuestarios, control de reclamaciones) puesto que la recepción de una publicación seriada no es más que una adquisición extendida en el tiempo. Permite además la creación de registros de fondos, el establecimiento de modelos predictivos de recepción con posibilidad de actualización automática de los fondos, o el control de las encuadernaciones (Martín 2008).

[5] Desiderata: Conjunto o lista de cosas que se desea adquirir u obtener. (RAE 2017)

OPAC: es el módulo que muestra públicamente el contenido del catálogo. Es decir, es la interfaz que permite a los usuarios acceder al corazón del sistema (el catálogo) e interactuar con él. Las últimas generaciones de OPAC son sistemas web, e incorpora a las tradicionales operaciones de búsqueda y recuperación de la información, otros servicios de valor añadido, algunos de ellos personalizados.

El OPAC es el módulo que gestiona las interacciones entre los usuarios finales y el sistema. El principal servicio del OPAC es la consulta bibliográfica del catálogo, pero no es el único, coordina todas las operaciones necesarias para que estos puedan acceder a la información, como la disponibilidad de los ejemplares en la misma, con el historial de préstamos, etc. Para ello ofrece diferentes opciones de búsqueda y recuperación de datos (búsquedas simples, avanzadas, mediante operadores booleanos u otras combinaciones), puede incluso, incluir en los resultados de búsquedas recursos que se encuentren en otras bibliotecas.

1.2. Los OPAC en los sistemas de gestión bibliotecaria

Los OPAC aparecieron en las bibliotecas hace más de cinco décadas. Su introducción coincide con la aceptación de los bibliotecarios con respecto a la informatización de las principales tareas de las bibliotecas. Con los constantes avances tecnológicos se han ido adaptando a un entorno tecnológico en continuo cambio. Han tenido una evolución que se sintetiza en tres generaciones (Miralles 2009) :

- Primera generación: los primeros catálogos en línea, sustitutos de los sistemas manuales, aparecieron en EE.UU hacia los años 60. Fueron diseñados para facilitar la catalogación y estaban orientados a los conocimientos de los bibliotecarios y de usuarios muy expertos. Fueron concebidos según el modelo manual y permitían el acceso a través de los encabezamientos tradicionales de los ficheros manuales (autores, títulos y materias). Presentaban un único formato de visualización de los registros. La consulta resultaba fallida si no había correspondencia exacta entre los términos utilizados en la catalogación y los términos empleados en la búsqueda. Carecían de la uniformidad de criterios que más tarde proporcionaría el formato MARC, lo cual dificultaba las búsquedas y el intercambio de registros bibliográficos. En estos catálogos de primera generación cambia el medio de consulta, a través de interfaces con menús, pero no la forma de consulta.
- Segunda generación: poseen una interfaz más amigable para los usuarios, los métodos de descripción bibliográfica se van normalizando paulatinamente, las opciones de recuperación

aumentan, permiten la realización de búsquedas por palabras clave en todos los campos o búsquedas en texto libre, admiten operadores booleanos, truncamientos y permiten refinar las búsquedas bibliográficas por fechas, lengua, formato, etc. Ofrecen la visualización de la información en formatos ISBD o MARC y disponen de pantallas de ayuda. Estos catálogos presentan importantes deficiencias en la búsqueda por materias. Los usuarios no aprovechan todas las posibilidades de los catálogos, al no conocer todas las opciones de búsqueda.

- Tercera generación: intentan mitigar las deficiencias de los catálogos de la segunda generación desarrollando las opciones de la búsqueda avanzada, mejorando la visualización, presentación y navegación a través de los resultados. La mayoría de los catálogos en línea incorporan la interfaz gráfica de Windows, lo que simplifica su uso. A finales de los 90, con la aparición de Internet, se facilita la consulta del catálogo independientemente del lugar donde se encuentre el usuario, por lo que se extiende y populariza la consulta de éste traspasando los límites físicos de la biblioteca. De hecho, los catálogos se han beneficiado enormemente del crecimiento de Internet y han adoptado el entorno web como soporte para ser implementados. En esta generación, además, los catálogos ofrecen asistencia al usuario con el desarrollo de la ayuda en línea. Se vincula el catálogo con el módulo de circulación, uniendo los datos bibliográficos y los datos del estado de los documentos, y con el módulo de publicaciones periódicas de los SIGB.

1.2.1. Proceso de recuperación de registros bibliográficos

Los catálogos en línea permiten leer y recuperar los datos almacenados en una base de datos de registros bibliográficos por medio de numerosos puntos de acceso, donde el usuario recupera información acerca de materiales existentes en una biblioteca.

Recuperar significa volver a tener. Recuperar información significa volver a tener una información que alguna vez, hace unos minutos o hace unos años, ha sido producida con anterioridad.

El término recuperación de información puede ser muy amplio, como un campo académico de estudio, la recuperación de información (RI) es encontrar materiales de una naturaleza no estructurada que satisface una necesidad de información dentro de grandes colecciones (Abadal y Codina 2005).

La RI es la disciplina que estudia la representación, la organización y el acceso eficiente a la información que se encuentra registrada en documentos (Lafferty y Chengxiang 2017).

En los catálogos de las bibliotecas los elementos principales para recuperar y poder acceder a los materiales buscados previamente son los registros bibliotecarios, estos son un *"conjunto de elementos informativos, organizados conforme a unas normas, que permiten identificar a una unidad documental de manera unívoca en vistas a su localización y posterior recuperación"* (Rios 2003).

Un proceso de recuperación en un catálogo en línea comienza cuando el usuario realiza una consulta al catálogo obteniendo no solo un objeto dentro de la colección, sino varios objetos que sean respuesta a la consulta con diferentes grados de relevancia.

Para recuperar efectivamente la información, los documentos son transformados en una representación lógica de los mismos. Cada estrategia o técnica de recuperación incorpora un modelo específico para sus propósitos.

Los **modelos** están categorizados de acuerdo a dos dimensiones:

1. Base Matemática:
    - *Modelos basados en teoría de conjuntos:* Los documentos se representan como un conjunto de palabras o frases. Los más comunes son, el método Booleano, el Fuzzy, y otros.
    - *Modelos Algebraicos*: En estos modelos los documentos y las consultas se representan como vectores, matrices o tuplas. La similitud entre un documento y una consulta se representa por un escalar. Dentro de ellos se tienen el modelo Vectorial, el Vectorial Extendido, y otros.
    - *Modelos Probabilísticos*: Tratan el proceso de recuperación de documentos como una inferencia probabilística. Las similitudes son calculadas como las probabilidades de que un documento sea relevante dada una consulta. Ejemplo de estos, son las Redes de Inferencia y las Redes de Creencia.
2. Propiedades de los modelos:
    - *Modelos sin independencia entre términos:* Tratan a los términos como si fueran independientes.
    - *Modelos con dependencia entre términos*: Permiten representar las interdependencias entre términos.

Para que la recuperación de la información sea con calidad, se deben cumplir un conjunto de criterios básicos(Villén-Rueda 2006):

- Consistencia: Capacidad que tiene un sistema de búsqueda en coordinar su sistema de clasificación con el lenguaje de búsqueda, permitiendo de esta manera establecer ecuaciones de búsqueda sobre términos admitidos.
- Exhaustividad: Es la cualidad de un sistema de información para recuperar la totalidad de los documentos relevantes que posee una colección, conforme a los requerimientos establecidos en la estrategia de búsqueda.
- Tasa de acierto: coeficiente que surge de dividir el número de documentos relevantes recuperados, sobre el número total de documentos relevantes de la colección
- Relevancia: Característica de un documento recuperado que cumple con las necesidades de información.
- Tasa de relevancia: coeficiente que surge de dividir el número de documentos relevantes recuperados, sobre el número total de documentos recuperados
- Pertinencia: Es la cualidad que tiene el documento recuperado de adaptarse a las necesidades de información.
- Tasa de pertinencia: coeficiente que surge de dividir el número de documentos pertinentes recuperados, sobre el número total de documentos recuperados
- Precisión: es la capacidad que tiene el sistema de búsqueda en coordinar la ecuación con los documentos más relevantes. De otra forma son aquellos documentos relevantes recuperados.
- Tasa de precisión: coeficiente que surge de dividir el número de documentos relevantes recuperados, sobre el número total de documentos de la colección

1.2.2. SIGB existentes, análisis del OPAC

ABCD 3.0

ABCD fue creado con el propósito de brindar funciones de automatización para las bibliotecas y para cualquier proveedor de información, como los centros de documentación. Este paquete de software existe como un conjunto de módulos relativamente independientes.

Cubre las funciones principales en una biblioteca:

- Adquisiciones
- Gestión de usuarios
- Gestión de préstamos

- Control de publicaciones periódicas

ABCD se denomina una 'suite' de software para bibliotecas y centros de documentación porque está formado por módulos relativamente independientes, que pueden cooperar entre sí, pero también pueden existir de manera aislada. De hecho, softwares avanzados ya existentes, que han mostrado su potencial en entornos exigentes como son las aplicaciones de BIREME (dentro del contexto de la Biblioteca Virtual en Salud), han sido adoptados y adaptados en ABCD (Ascencio 2015).

Características del módulo OPAC:

- Una amigable interfaz de usuario.
- Realiza búsquedas usando operadores lógicos.
- Muestra íconos para identificar el tipo de material.
- Permite la visualización de los resultados en tres vistas, normal, Marc e ISBD.
- Mostrar la cantidad (número) de resultados coincidentes de la búsqueda realizada.
- Presentar la opción de autenticación y autorización de usuario para poder realizar funcionalidades sobre los materiales existentes en el catálogo.

Absys

Es una solución integrada para la gestión de bibliotecas, que lleva evolucionando muchos años para aportar a las bibliotecas una herramienta que responda a todas sus necesidades, así como ayudar a potenciar su eficacia y mejorar su servicio.

Es desarrollado por Servicios de Teledocumentación S.A. BARATZ y distribuido por tres empresas en Europa:

- IRIS (Bélgica)
- SINORG (Francia)
- Santa Cruz Informática (Portugal)

Como resultado Absys se ha convertido en un referente en el mundo de la automatización de bibliotecas, en una aplicación cuyas soluciones, además de novedosas, son sólidas y fiables, basadas en la utilización de las herramientas más avanzadas (Cháfer 2015).

Absys es un producto ya consolidado pero que continúa evolucionando y adaptándose a las necesidades y peculiaridades de los diferentes tipos de bibliotecas para garantizar una rápida y eficaz puesta en marcha de su proyecto de automatización.

Características (Cháfer 2015):

- Herramienta completa, potente y amigable, además de presentar facilidades de implantación y uso.
- Sistemas operativos en los que funciona: Linux, Windows.
- Multilenguaje (castellano, inglés, portugués, holandés…).
- Posibilidad de uso en internet o intranet, en función de las necesidades del usuario.
- Flexible y con capacidad de adaptación a las características de la biblioteca.
- Personalizable y ajustada a estándares oficiales.
- Expandible, ya que pueden instalarse los módulos por separado.

Entre las características del módulo OPAC están:

- Interfaz amigable y fácil de usar.
- Implementa la búsqueda facetada.
- Realiza búsquedas usando operadores lógicos.
- Permite la visualización de los resultados en tres vistas, normal, Marc e ISBD.
- Adaptable a la imagen de la administración bibliotecaria que gestiona.
- Accesibilidad al catálogo a través de internet o intranet desde cualquier tipo de navegador.
- Alto grado de personalización, para crear diferentes modelos acordes con los grupos de usuarios.

PMB

Sistema bibliotecario integrado, iniciado por François Lemarchand en octubre de 2002, director de la biblioteca pública de Agneaux. Desarrollado por PMB Services y distribuido bajo la Licencia CeCILL29 (licencia francesa de *software* libre, la cual es una adaptación de la licencia *General Public License* a la legislación francesa y a los tratados internacionales), es un SIGB de código abierto que funciona bajo plataformas Linux y Windows, y requiere servidor web apache, PHP y MySQL (Olivert 2015).

Este sistema integrado que ha tenido éxito en las bibliotecas de Francia. Está traducido a diversos idiomas, entre ellos el castellano.

Es un software configurable, es decir, que puede adecuarse a diversas necesidades, tanto por su naturaleza de software libre, como por la posibilidad de establecer parámetros específicos para su uso.

Se describe como un SIGB de código abierto, tratándose de *"un software robusto, potente y con muchas opciones diferentes de configuración, lo que le convierte en un programa ideal para cualquier tipo de organización"* (Senso 2011). Este mismo autor señala que es perfectamente utilizable tanto en grandes como en pequeñas bibliotecas, y que es lo suficientemente flexible como para poder adaptarlo a las necesidades de cada centro.

Características:

- Sistema de gestión de Bases de Datos: Emplea el administrador de bases de datos MySQL, un gestor de bases de datos de tipo relacional que trabaja con diversos sistemas operativos entre los que se encuentran Linux y Windows. Al mismo tiempo, es un sistema gestor de bases de datos relacional cliente/servidor de coste mínimo que incluye un servidor SQL, programas de cliente que permiten acceder al servidor, herramientas y una interfaz de programación para escribir programas. Además, está agrupada en la licencia GLP (Sarma 2016).
- Capacidad de la base de datos: Se puede decir que la capacidad de la base de datos, depende del tamaño que se les asigne a las tablas de la misma, así como al tamaño del disco del sistema operativo. La capacidad total de la base de datos en MySQL lo define la delimitación del tamaño de los ficheros ubicados en el sistema operativo.
- Lenguaje de programación: PHP es un lenguaje de programación, que se emplea para entornos web. Es un lenguaje *open source* interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor.
- Plataforma de hardware: utiliza un procesador con su respectivo disco duro, en el cual está instalado el servidor Apache (servidor que trabaja independientemente del sistema operativo en que se encuentre y su adquisición es de bajo costo).
- Arquitectura cliente-servidor: es una estructura para elaborar sistemas de comunicación que permita la interacción entre seres humanos y recursos. En una arquitectura DBMS cliente-servidor básica, la funcionalidad del sistema se distribuye entre dos módulos, la parte del servidor y la parte del cliente.
- Conexión de sistemas externos: Utiliza el protocolo Z/39.50, que es un estándar que se utiliza principalmente en el módulo de catalogación, para intercambio de información entre sistemas informáticos.

  Se trata de un sistema de capacidad multilingüe no sólo en lo referente a la interfaz (aunque hay que señalar que algunas traducciones no están completas), sino también a nivel de los datos que maneja,

pues soporta el estándar de codificación de caracteres UNICODE. Además, en su funcionamiento también incorpora otros estándares: cumple con las especificaciones de los formatos MARC, con XML, con el protocolo OAI-PMH y también permite la consulta y recuperación de registros bibliográficos a través del protocolo Z39.50.

- Interfaz: Formato web como elemento de presentación, entendiendo por web, una interfaz que necesita un navegador web para su uso, como es el caso de Mozilla Firefox o Internet Explorer, por poner un ejemplo.
- El sistema de búsqueda sencilla y avanzada además de amigable y de fácil uso permite ubicar el material de forma más rápida y precisa. La interfaz es uno de los elementos más importantes, ya que mediante una estructura tipo web permite el acceso tanto al área administrativa como a la parte pública de forma sencilla (Senso 2011).

Entre las características del módulo OPAC están:

- Búsqueda por tipo de biblioteca.
- Realiza búsquedas usando operadores lógicos.
- Envía bibliografías por correo electrónico.
- Muestra íconos para identificar el tipo de material.
- Permite la visualización de los resultados en tres vistas, normal, Marc e ISBD.
- Posibilidad de cambiar el interfaz en varios idiomas.
- Muestra datos de la biblioteca.
- Ingreso a cuenta de usuario.
- Gestión de publicaciones periódicas
- Realizar reservas.
- Creación de RSS.

Koha

Koha es un sistema integrado para bibliotecas de código abierto, cuyo nombre significa regalo o donación. El proyecto se inició en 1999 en Nueva Zelanda, siendo elaborada la primera versión libre en el año 2000, que fue financiada por *Katipo Communications* (Madhusudhan y Singh 2016).

Koha es un sistema de código abierto que trabaja con la licencia GNU (Licencia General Pública), diseñada para trabajar en Linux. Es uno de los SIGB de fuente abierta y software libre que ha logrado un mayor alcance geográfico, registrándose su adopción en países de África, América Latina, América del Norte y Europa.

Características:

- Interfaz intuitiva.
- Listados de lecturas de los usuarios.
- Sistema completo de adquisiciones (con presupuestos y tasaciones incluidos); y más simple si la biblioteca es más pequeña.
- Sistema de seriales para diarios y revistas.
- Es un sistema multiplataforma.
- Posee interfaz web
- Administración remota por parte del bibliotecario, desde un móvil o un asistente personal.
- Manejo de Informes, Reportes y Estadísticas gracias a las bases de datos relacionales.

Características del módulo OPAC:

A través de este portal web, el usuario puede interactuar con el centro de documentación y/o biblioteca. Este módulo les permite a los usuarios:

- Consultar el catálogo.
- Implementa la búsqueda facetada.
- Realiza búsquedas usando operadores lógicos.
- Envía bibliografías por correo electrónico.
- Muestra íconos para identificar el tipo de material.
- Permite la visualización de los resultados en tres vistas, normal, Marc e ISBD.
- Incluir comentarios de los libros o documentos.
- Consultar listas de recomendados.
- Crear bibliografías e imprimirlas y/o enviarlas por correo electrónico.
- Leer noticias que se editen desde la biblioteca.
- Entrar en un espacio privado desde el que se puede:

- ✓ Crear una lista de documentos favoritos.
- ✓ Renovar préstamos.

Kobli

Kobli en un Sistema Integrado de Gestión Bibliotecaria de fuentes abiertas adaptado por el grupo de trabajo de las Bibliotecas de la Administración General del Estado de España, a partir de Koha, uno de los principales SIGB basados en *software* libre en cuanto a capacidad e implantación.

Entre sus principales características destacan:

- Es un sistema informático diseñado para ser instalado en un servidor de la organización, con el fin de dar servicio a usuarios internos y externos a la misma.
- Está compuesto por módulos específicos integrados, compartiendo un núcleo común (repositorio, sistema de análisis e indización, etc.) e interrelacionados entre sí.
- A través de la funcionalidad específica de cada uno de los módulos que incluye, soporta los procesos y servicios necesarios para la gestión bibliotecaria, entre otros: OPAC, Circulación, Catalogación, Publicaciones periódicas, Adquisiciones, Informes, Socios, etc.
- Está licenciado mediante una licencia libre, concretamente GNU GPL, y se apoya en el uso de estándares y herramientas de fuentes abiertas. La característica de fuentes abiertas o código abierto está relacionada con la posibilidad de consultar, modificar y adaptar el código del sistema. Gracias a esta característica, la plataforma es altamente extensible, personalizable y con mantenimiento frecuente, por técnicos y administradores de sistemas.

Características del módulo OPAC:

- Consultas al catálogo (localizar recursos):
  - ✓ Búsqueda simple.
  - ✓ Búsqueda avanzada (limitar y redefinir la búsqueda)
  - ✓ Búsquedas en otros catálogos.
  - ✓ Búsqueda por nube de etiquetas.
- Multilenguaje
- Muestra el historial de búsquedas.
- Implementa la búsqueda facetada.
- Realiza búsquedas usando operadores lógicos.

- Envía bibliografías por correo electrónico.
- Muestra íconos para identificar el tipo de material.
- Permite la visualización de los resultados en tres vistas, normal, Marc e ISBD.
- Permite la reserva de materiales.

OpenBiblio

Es un sistema integrado de gestión bibliotecaria de código abierto. Creado en el 2002 por Dave Stevens, utilizando PHP como lenguaje de programación.

El sistema ha sido traducido al español y se utiliza en el sistema de educación primaria de Chile, y ha llamado la atención de otros como Cuba, Colombia, etc.

Contiene un catálogo en línea, una sección para préstamo, informes, catalogación, y la funcionalidad de administración de personal (Jiménez y Gutiérrez 2010).

Características:

- Una interfaz de usuario simple y clara.
- Es un sistema multiplataforma.
- Visualizar los detalles de un resultado de búsqueda en formato Normal, Marc e ISBD.
- Sugerir materiales a la biblioteca.
- Recomendar un material a otros usuarios del sistema.
- Reservar un material.
- Listados de lecturas de los usuarios.
- Sistema de seriales para diarios y revistas.

Características del módulo OPAC:

- Realiza búsquedas usando operadores lógicos.
- Muestra íconos para identificar el tipo de material.
- Permite la visualización de los resultados en tres vistas, normal, Marc e ISBD.
- Mostrar la cantidad (número) de resultados coincidentes de la búsqueda realizada.
- La posibilidad de gestionar etiquetas.

- La opción de emitir comentarios sobre un material seleccionado.
- Crear listas de selección.
- Es multilenguaje.
- Permite realizar búsquedas configurables (incorporación de nuevas opciones de búsquedas).

Luego de realizar el estudio de los diferentes SIGB y el análisis de los módulos OPAC se realizó un resumen de algunos aspectos pertenecientes a cada uno de ellos:

Tabla 1. Caracterización de los módulos OPAC de los SIGB.

	Sistemas existentes					
Indicadores	ABCD	Absys	PMB	Koha	Kobli	Openbiblio
Envía bibliografías por correo electrónico	No	No	Si	Si	Si	No
Implementa la búsqueda facetada	No	Si	No	Si	Si	No
Muestra íconos para mostrar el tipo de material	Si	No	Si	Si	Si	Si
Realiza búsqueda usando operadores lógicos	Si	Si	Si	Si	Si	Si
Vista de los resultados de búsqueda	N, M e I	N, M e I	N, M e I	N, ME e I	N, M e I	N, M e I
Base de datos	R y NR	R	R	R	R	R

N: vista de los resultados en formato normal, **M:** vista de los resultados en formato Marc, **ME:** vista de los resultados en formato Marc extendido **I:** vista de los resultados en formato ISBD, **R:** base de datos relacional, **NR:** base de datos no relacional
Fuente: Elaboración propia.

Con la caracterización de los módulos OPAC de los SIGB estudiados se detectaron algunas funcionalidades que pueden ser parte de la propuesta de solución, para así mejorar a la gestión bibliotecaria y al usuario en sus investigaciones.

La propuesta de solución puede incluir, entre otras, algunas características y funcionalidades tales como:

- Interfaz simple al usuario
- Menú interactivo

- Búsqueda de recursos:
  - ✓ Búsqueda simple
  - ✓ Búsqueda avanzada (limitar y redefinir la búsqueda)
  - ✓ Búsqueda por facetas
- Visualizar detalles de un material.
- Mostrar la cantidad de resultados encontrados

1.3. Metodología, herramientas y lenguajes utilizados

En este epígrafe se explicarán la metodología, herramientas y lenguajes de programación empleados en la realización de este trabajo.

1.3.1. Metodología

Variante AUP-UCI

La metodología de desarrollo de software Proceso Unificado Ágil (AUP) para la UCI, es una variedad creada por la universidad de la metodología AUP y está definida como el escrito oficial que rige la actividad de producción de software en la universidad.

Según Rodríguez, la variación AUP-UCI tiene tres fases (2015):

1. 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.
2. 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. Durante esta fase el producto es transferido al ambiente de los usuarios finales o entregado al cliente. Además, en la transición se capacita a los usuarios finales sobre la utilización del software.

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

Descripción de las disciplinas:

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 8 disciplinas, 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 y la disciplina Despliegue se considera opcional. 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 Gestión de la configuración (CM), Planeación de proyecto (PP) y Monitoreo y control de proyecto (PMC) (Arias y Calvache 2016):

- Modelado de negocio (opcional): El Modelado del Negocio es la disciplina destinada a comprender los procesos de negocio de una organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el software desarrollado va a cumplir su propósito.
- Requisitos: El esfuerzo principal en la disciplina Requisitos es desarrollar un modelo del sistema que se va a construir. Esta disciplina comprende la administración y gestión de los requisitos funcionales y no funcionales del producto. Existen tres formas de encapsular los requisitos [Casos de Uso del Sistema (CUS), Historias de usuario (HU) y Descripción de requisitos por proceso (DRP)], agrupados en cuatro escenarios condicionados por el Modelado de negocio.
- Análisis y diseño: En esta disciplina, si se considera necesario, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema (incluyendo su arquitectura). Además, en esta disciplina se modela el sistema y su forma (incluida su arquitectura) para que soporte todos los requisitos, incluyendo los requisitos no funcionales. Los modelos desarrollados son más formales y específicos que el de análisis.
- Implementación: En la implementación, a partir de los resultados del Análisis y Diseño se construye el sistema.

- Pruebas internas: En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas. Se deben desarrollar artefactos de prueba como: diseños de casos de prueba, listas de chequeo y de ser posible, componentes de prueba ejecutables para automatizar las pruebas.
- Pruebas de liberación: Pruebas diseñadas y ejecutadas por una entidad certificadora de la calidad externa, a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación.
- Pruebas de Aceptación: Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido.

Descripción de los escenarios

Escenario No 1: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que puedan modelar una serie de interacciones entre los trabajadores del negocio/actores del sistema (usuario), similar a una llamada y respuesta respectivamente, donde la atención se centra en cómo el usuario va a utilizar el sistema.

Escenario No 2: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los conceptos fundamentales del negocio. Se recomienda este escenario para proyectos donde el objetivo primario es la gestión y presentación de información.

Escenario No 3: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio con procesos muy complejos, independientes de las personas que los manejan y ejecutan, proporcionando objetividad, solidez, y su continuidad. Se debe tener presente que este escenario es muy conveniente si se desea representar una gran cantidad de niveles de detalles y la relaciones entre los procesos identificados.

Escenario No 4: Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio muy bien definido. El cliente estará siempre acompañando al equipo de desarrollo

para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos. Se recomienda en proyectos no muy extensos, ya que una HU no debe poseer demasiada información.

De los cuatro escenarios que define la metodología, el que se adapta a esta investigación es el escenario 2, ya que el mismo está destinado para proyectos donde el principal objetivo es la gestión y presentación de la información. Este escenario exige para la modelación del negocio el Modelo Conceptual y para encapsular los requisitos los Casos de Uso del Sistema.

1.3.2. Herramientas

Visual Paradigm 8.0

Es una herramienta profesional que se utiliza para el modelado, soporta el ciclo de vida completo del desarrollo de software: análisis y diseño, construcción, pruebas y despliegue. Permite el diseño centrado en casos de uso y enfocado al negocio posibilitando la generación de un software de gran calidad. Posee un lenguaje estándar y común. Proporciona la ventaja de realizar ingeniería directa e inversa. El modelo y el código permanecen sincronizados en todo el ciclo de desarrollo. El Visual Paradigm puede construir diferentes tipos de diagramas que permiten ver el sistema desde diferentes perspectivas, entre ellos se encuentran los de casos de uso, de clase, actividad, estado, componentes, secuencia, entre otros.

Otra de sus ventajas es que permite hacer paquetes de trabajo que proveen de un mecanismo de organización de los modelos agrupando elementos de modelado, siendo esto de gran ayuda a la hora de desarrollar sistemas de gran envergadura y complejidad.

PostgreSQL v9.4

Es un Sistema Gestor de Bases de Datos libre. Corre en la mayoría de los Sistemas Operativos más utilizados incluyendo, GNU/Linux y Windows. Cumple el principio ACID (Atomicidad, Consistencia, Integridad, Durabilidad) y tiene soporte completo para llaves foráneas, unión, vistas, subconsultas, disparadores (*triggers*), y procedimientos almacenados (en varios lenguajes), incluye la mayoría de los tipos de datos de los estándares SQL92 y SQL99, así como herencia entre tablas, por lo que se le considera un gestor de BD relacionales. Posee una documentación completa. Mediante un sistema denominado Acceso Concurrente Multiversión (por sus siglas en inglés, *Multiversion Concurrency Control* o MVCC) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.

J-ISIS

Es un software de código abierto utilizado para crear, actualizar y buscar bases de datos. Se basa en Unicode, una ventaja para los usuarios que requieren caracteres extendidos o alfabetos diferentes. Ha sido desarrollado por la UNESCO como una suite multiplataforma modular, de fácil mantenimiento y ampliable FOSS ISIS que proporcionaría los mismos conceptos y funcionalidades exitosas que la suite existente de ISIS (CDS-ISIS, Win-ISIS, etc.) mientras elimina las restricciones. La funcionalidad de indexación es proporcionada por Lucene. Puede importar bases de datos compatibles con ISO o MARC.

Angular v5.2.8

Es un framework para el desarrollo de aplicaciones web desarrollado en TypeScript de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador(MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles (Google 2016).

Bootstrap v4.0

Es un framework web o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.

Node.js 8.9.4 LTS

Node.js es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, npm (del inglés *Node Package Manager*), es el más grande de librerías de código abierto en el mundo. Es un gestor de paquetes, el cual hará más fácil el desarrollo, podremos tener cualquier librería disponible con solo una línea de código, npm nos ayudará a administrar nuestros módulos, distribuir paquetes y agregar dependencias de una manera sencilla (Node.js Foundation 2018).

Visual Studio Code v1.18.1

Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux y MacOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, los atajos de teclado y las preferencias. Es gratuito y de código abierto, es compatible con varios lenguajes de programación y un conjunto de características que pueden o no estar disponibles para un idioma determinado.

1.3.3. Lenguajes

TypeScript v2.6.2

TypeScript es un lenguaje de programación libre y de código abierto desarrollado y mantenido por Microsoft. Es un superconjunto de JavaScript, que esencialmente añade dos nuevos tipos, estático y objetos basados en clases. TypeScript extiende la sintaxis de JavaScript, por tanto, cualquier código JavaScript existente debería funcionar sin problemas. Está pensado para grandes proyectos, los cuales a través de un compilador de TypeScript se traducen a código JavaScript original (De Wolff y Hage 2017).

JavaSE 7

Java fue desarrollado por James Gosling de Sun Microsystem en 1995, es un lenguaje de programación de propósito general, orientado a objetos, fue diseñado para tener pocas dependencias de implementaciones. Este lenguaje permite realizar comprobaciones de tipos durante la compilación, evitando problemas tales como el desbordamiento de la pila. Gestiona la memoria dinámicamente. Otra característica es que está diseñado para la programación recurrente sin necesidad de utilizar ninguna biblioteca (Eckstein 2007).

UML

UML es un lenguaje estándar para especificar, visualizar, construir y documentar los artefactos de sistemas informáticos, así como para el modelado de negocios y otros sistemas. UML brinda variados elementos de esquematización para representar las diferentes partes de un sistema, de ellos se utilizan en el desarrollo de la aplicación los diagramas de casos de uso, de clases del diseño, de despliegue, de componentes, de colaboración y el resto de los diagramas.

1.4. Conclusiones parciales

En este capítulo se analizaron los principales conceptos relacionados con el tema central de la investigación lo que permitió entender el proceso de recuperación de los registros bibliográficos en los SIGB, obteniendo la base teórica necesaria para el desarrollo de la solución. Además, se caracterizó parte de los OPAC más usados en el mundo, lo que permitió identificar funcionalidades que podrían ser añadidas al módulo OPAC a desarrollar. Una vez conocidas las herramientas, técnicas y metodología especificadas para la elaboración de la solución, se realizó un estudio de la misma, haciendo énfasis en sus características.

Capítulo 2: Análisis y diseño de la solución

Introducción

En este capítulo se evidencian las principales características de la solución propuesta. Se hace un levantamiento de los requisitos funcionales y no funcionales necesarios para lograr que el módulo funcione correctamente. Además, se construyen los artefactos correspondientes al análisis y diseño acorde con la metodología seleccionada en el capítulo anterior.

2.1. Modelo de Dominio

Un modelo de dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. También se le denomina modelo conceptual, modelo de objetos del dominio y modelo de objetos de análisis. Al utilizar la notación UML, un modelo de dominio se representa con un conjunto de diagramas de clases en los que no se define ninguna operación. Su principal objetivo es definir las interrelaciones de los objetos más importantes representados mediante clases. Además, desempeña un papel clave en la comprensión del entorno actual (Larman 2003).

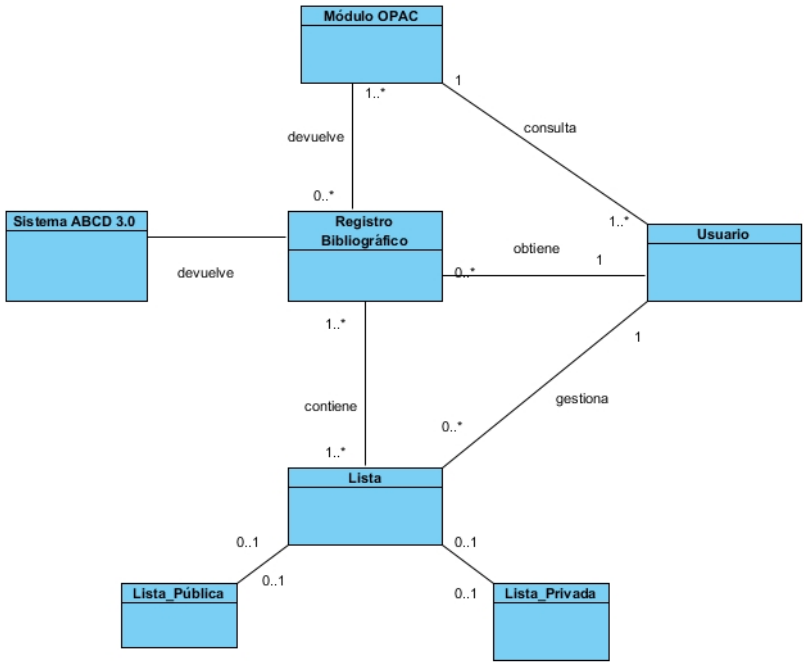

Figura 1 Modelo de dominio. Fuente: Elaboración propia

Descripción de los elementos del dominio

Sistema ABCD 3.0: es la entidad que representa al sistema ABCD 3.0

Base de Datos: representa la base de datos donde se encuentran almacenados los registros bibliográficos, devuelve los registros bibliográficos que cumplan con la búsqueda del usuario.

Usuario: es la entidad que representa a los usuarios, son los únicos actores del negocio.

Módulo OPAC: representa al módulo OPAC del sistema ABCD, con el que interactúa el usuario.

Registro Bibliográfico: esta entidad representa los resultados que obtiene el usuario al realizar una consulta.

Lista: la entidad representa las listas de selección de los usuarios.

Lista pública: representa las listas públicas de todos los usuarios.

Lista privada: son las listas que define cada usuario con sus registros de su preferencia, y a las cuales solo él tiene acceso.

2.2. Especificación de los requisitos de software

En el proceso de ingeniería de software, los requisitos de utilizan como datos de entrada en la etapa de diseño del producto que se desea lograr. Estos representan qué debe hacer el sistema, pero no cómo hacerlo y, además, incorporan las cualidades o propiedades que el mismo debe poseer (Somerville 2007)

2.2.1. Requisitos funcionales

Los requisitos funcionales (RF) definen el comportamiento interno de un software, son condiciones que el sistema ha de cumplir. Estos muestran las funcionalidades que deben satisfacerse para cumplir con las especificaciones de software (Sommerville, Arakaki y Melnikoff 2008).

RF_1: Realizar búsqueda simple

RF_1.1 Cualquier campo

RF_1.2 Por título

RF_1.3 Por autor

RF_2: Realizar búsqueda avanzada

RF_2.1 Empleando operadores booleanos

RF_3: Ordenar resultados

RF_3.1 Alfabéticamente por autor

RF_3.2 Alfabéticamente por título

RF_3.3 Alfabéticamente por tipo

RF_3.4 Por *rating*[6]

RF_4: Gestionar selección

RF_4.1 Agregar un registro a la selección

RF_4.2 Eliminar un registro de la selección

RF_5: Evaluar material

RF_6: Recomendar material

RF_7: Visualizar detalles de un material

RF_8: Gestionar lista de selección

RF_8.1 Agregar materiales de la lista de selección

RF_8.2 Visualizar materiales de una lista de selección

RF_8.3 Eliminar materiales de la lista de selección

RF_9: Administrar listas

RF_10: Autenticar usuario

RF_11: Gestionar reservación

RF_11.1 Registrar reservación

RF_11.2 Visualizar reservación

RF_11.3 Editar reservación

RF_12: Listar Reservas

RF_13: Gestionar sugerencias

RF_13.1 Agregar sugerencia

RF_13.1 Visualizar sugerencia

RF_13.1 Editar sugerencia

RF_13.1 Eliminar sugerencia

RF_14: Visualizar historial del usuario

[6] *Rating*: Valoración, evaluación

RF_15. Filtrar resultados

RF_2.8.1 Por tipo

RF_2.8.2 Por fecha

2.2.2. Requisitos no funcionales

Los requisitos no funcionales (RnF) son aquellos requisitos que no describen información a guardar, ni funciones a realizar, sino que son propiedades que debe cumplir el producto, un conjunto de características de calidad que es necesario tener en cuenta al diseñar e implementar el software (Díaz, Sánchez y Pastor 2005).

Requerimientos no funcionales de software

Usabilidad:

- RNF1: El sistema podrá ser utilizado para realizar búsquedas simples y avanzadas, para obtener registros bibliográficos de interés para el usuario.
- RNF2: Debe tener un diseño adaptativo para que pueda adecuar a diferentes resoluciones de los dispositivos móviles.
- RNF3: Debe implementar la búsqueda por facetas y así refinar los resultados de búsqueda, dependiendo de los resultados obtenidos.

Soporte

- RNF4: Se debe implementar el sistema siguiendo el estándar de codificación.
- RNF5: Los componentes de software que integran la solución se organizarán de forma modular.

Portabilidad

- RNF6: El sistema debe ser capaz de ejecutarse en diferentes plataformas.

Seguridad

- RNF7: El módulo debe ser seguro en función a la autenticación de usuarios

Confiabilidad

- RNF8: Se debe mostrar solo información necesaria en las notificaciones a los usuarios.

Eficiencia

- RNF9: Debe proporcionar apropiados tiempos de respuesta al usuario, y al procesar las peticiones del mismo.

Requerimientos no funcionales de hardware

- RNF10: Requisitos del sistema operativo: Linux y Windows
- RNF11: Máquina Virtual de Java (JVM) 1.7 (implementación de Oracle) y PostgreSQL 9.3 o superior.
- RNF12: Navegador: Mozilla Firefox 38+ o Google Chrome.

2.3. Casos de usos del sistema

Una vez conocidos los requisitos funcionales y no funcionales se definen los casos de uso del sistema (CUS), entre sus especificaciones están los detalles del flujo de eventos a ejecutar, así como el actor que lo inicializa. Las mismas establecen una guía para el desarrollador.

Para facilitar el desarrollo de la investigación se dividieron las diferentes funcionalidades en dos grupos. El primer grupo se encuentras las funciones que un usuario tiene disponible sin autenticarse en el módulo OPAC. EL segundo grupo se encuentran las funcionalidades que tiene el usuario una vez se autentica en el módulo OPAC.

A continuación, se describen el actor y los casos de usos para el primer y segundo grupo respectivamente

Tabla 2. Descripción del actor Consultor del catálogo.

Actor	Descripción del autor
Consultor del catálogo	Realiza búsquedas en el catálogo, y de los resultados arrojados en la misma puede efectuar operaciones sobre ellos

Fuente: Elaboración propia

Tabla 3. Descripción de los casos de uso del sistema

Caso de uso	Nombre	Descripción del caso de uso
CU_1	Autenticar usuario	Permite al usuario autenticarse en el OPAC.

CU_2	Consultar catálogo	Permite realizar una búsqueda de un ejemplar en el catálogo especificando un criterio de búsqueda.
CU_3	Consultar catálogo (avanzado)	Permite buscar un material en el catálogo por diferentes criterios de búsqueda, usando operadores booleanos.
CU_4	Gestionar selección	Permite al actor agregar, visualizar, eliminar un registro de la selección.
CU_5	Evaluar material	Permite al actor evaluar un ejemplar, según su criterio.
CU_6	Recomendar material	Permite al actor recomendar un material a un usuario. Permite ver las recomendaciones que se le han hecho al usuario autenticado.
CU_7	Visualizar detalles de un material	Permite ver los detalles de un material seleccionado
CU_8	Gestionar lista de selección	Permite agregar, visualizar o eliminar materiales a una lista de selección.
CU_9	Ordenar resultados	Permite al usuario, ordenar los resultados de búsqueda por diferentes criterios

Fuente: Elaboración propia

Tabla 4. Descripción del actor Usuario de la biblioteca.

Actor	Descripción del autor
Usuario de la biblioteca	Realiza búsquedas en el catálogo, y de los resultados arrojados en la misma puede efectuar operaciones sobre ellos. Permite hacer reservas, sugerencias, evaluar materiales.

Fuente: Elaboración propia

Tabla 5. Descripción de los casos de uso del sistema

Caso de uso	Nombre	Descripción del caso de uso
CU_10	Administrar listas	Permite al usuario crear listas de selección públicas o privadas, además de poder enviar las listas por correo electrónico. El usuario solo podrá editar y/o eliminar las listas creadas por él.
CU_11	Gestionar reservación	Permite al actor listar, registrar, visualizar, editar, o eliminar una reservación de un material.
CU_12	Gestionar sugerencias.	Permite al usuario agregar, visualizar, editar, eliminar las sugerencias.
CU_13	Visualizar historial del usuario	Permite al usuario revisar su resumen histórico con las actividades del mismo en el sistema.

Fuente: Elaboración propia

2.3.1 Matriz de trazabilidad

La matriz de trazabilidad relaciona dos elementos esenciales para la buena ejecución de las labores de un proyecto: los requisitos establecidos para dicha ejecución y el valor que cada uno de ellos agrega al conjunto del proceso. Es una herramienta clave para la ingeniería de los proyectos, así como para el seguimiento de los diversos elementos que los componen (Guerrero y Martín 2016).

A continuación, se muestra la matriz de trazabilidad de los requisitos funcionales con los casos de uso Requisitos_Casos), la cual establece la relación entre ellos. Las restantes matrices se encuentran anexadas al final del documento.

Tabla 2: Matriz de trazabilidad Requisitos_Casos de Uso

	CU_1: Autenticar usuario	CU_2: Consultar catálogo	CU_3: Consultar catálogo (avanzado)	CU_4: Gestionar selección	CU_5: Evaluar material	CU_6: Recomendar material	CU_7: Visualizar detalles de un material	CU_8: Gestionar lista de selección	CU_9: Ordenar resultados	CU_10: Administrar listas	CU_11: Gestionar reservación	CU_12: Gestionar sugerencias	CU_13: Visualizar historial del usuario
RF_1: Realizar búsqueda simple		X											
RF_2: Realizar búsqueda avanzada			X										
RF_3: Ordenar resultados		X	X						X				
RF_4: Gestionar selección				X									
RF_5: Evaluar material					X								
RF_6: Recomendar material						X							
RF_7: Visualizar detalles de un material							X						
RF_8: Gestionar lista de selección								X					
RF_9: Administrar listas										X			
RF_10: Autenticar usuario	X												
RF_11: Gestionar reservación											X		
RF_12: Listar Reservas											X		
RF_13: Gestionar sugerencias												X	
RF_14: Visualizar historial del usuario													X

RF_15. Filtrar resultados		X	X										

Fuente: Elaboración propia

2.3.1 Descripción de Casos de Usos

Para una mejor comprensión de las funcionalidades asociadas a cada caso de uso es necesario describirlos. Dicha descripción puede ser elaborada de forma breve o extendida. Una especificación de caso de uso proporciona detalles textuales de un caso de uso. A continuación, se describen detalladamente varios de los casos de uso del sistema:

Tabla 3: CU_1 Autenticar usuario.

Caso de Uso	***Autenticar usuario***	
Descripción	Permite a los usuarios validar su identidad para acceder al catálogo en línea a través de un *modal*[7], para que inserte usuario y contraseña.	
Actor	Consultor del catálogo	
Resumen	El caso de uso inicia cuando el usuario que consulta el catálogo en línea desea acceder al mismo para realizar otras acciones.	
Complejidad	Media	
Prioridad	Alta	
Precondiciones	El usuario debe estar previamente registrado	
Postcondiciones	El sistema permite el acceso al catálogo en línea	
Flujo de eventos		
Flujo básico Autenticar usuario		
	Actor	**Sistema**
1.	Selecciona la opción para iniciar sesión.	
2.		Muestra un *modal* para que el usuario inserte su usuario y contraseña.
3.	El usuario inserta los datos para acceder al catálogo en línea.	

[7] *Modal:* Son cuadros de diálogo que aparecen sobre la página, bloqueando todas las funciones para concentrar el foco en una acción particular.

	• Usuario • Contraseña Da clic en el botón “Iniciar sesión”	
4.		Verifica que los datos insertados por el usuario sean correctos: • Si los datos son correctos el usuario accede al sistema • Si los datos no son correctos, se muestra un mensaje de error “Usuario o contraseña incorrectos”, y comienza de nuevo en el paso 3.

Tabla 4: CU_2 Consultar Catálogo

Caso de Uso	**Consultar catálogo**	
Descripción	Permite a los usuarios realizar búsquedas simples en el catálogo para encontrar materiales de su interés.	
Actor	Consultor del catálogo, Usuario de la biblioteca	
Resumen	El caso de uso inicia cuando el usuario desea realizar una búsqueda en el catálogo en línea y culmina una vez el usuario obtiene los resultados de la búsqueda.	
Complejidad	Media	
Prioridad	Alta	
Precondiciones	El usuario debe proporcionar un valor para realizar la búsqueda	
Postcondiciones	El sistema muestra los resultados de la búsqueda	
Flujo de eventos		
Flujo básico <Consultar catálogo>		
	Actor	**Sistema**
1.	Selecciona un criterio de búsqueda.	
2.	Si no selecciona un criterio de búsqueda, se buscará en cualquiera de ellos	
3.	Proporciona un valor para realizar la búsqueda	

4.	Da clic en el botón “Buscar”.	
5.		Verifica que el valor para realizar la búsqueda no este vacío.
6.		Si el valor para buscar es vacío, muestra un mensaje informándolo y comienza de nuevo el paso 3
7.		Si el valor de búsqueda no es vacío, se procede a buscar.
8.		Si no encuentra materiales que coincidan con el valor de búsqueda proporcionado se muestra un mensaje informándolo para que establezca un nuevo valor de búsqueda y comienza de nuevo el paso 3.
9.		Si encuentra materiales que coincidan con el valor de búsqueda proporcionado se muestra un listado con los mismos.
10.	Obtiene un listado con los resultados de búsqueda	

Tabla 5: CU_3 Consultar Catálogo (Avanzado)

Caso de Uso	**Consultar catálogo (Avanzado)**	
Descripción	Permite a los usuarios realizar búsquedas avanzadas en el catálogo para encontrar materiales de su interés, puede emplear operadores booleanos.	
Actor	Consultor del catálogo, Usuario de la biblioteca	
Resumen	El caso de uso inicia cuando el usuario desea realizar una búsqueda avanzada en el catálogo en línea y termina cuando el usuario obtiene un listado con los resultados de la búsqueda.	
Complejidad	Media	
Prioridad	Alta	
Precondiciones	El usuario debe proporcionar varios valores para realizar la búsqueda	
Postcondiciones	El sistema muestra los resultados de la búsqueda	
Flujo de eventos		
Flujo básico <Consultar catálogo (Avanzado)>		
	Actor	**Sistema**
1.	Selecciona un criterio de búsqueda. (“Cualquiera”, por defecto)	

2.	Proporciona un valor para realizar la búsqueda	
3.	Escoge un operador booleano (“AND”, por defecto)	
4.	Selecciona otro criterio de búsqueda. (“Cualquiera”, por defecto)	
5.	Proporciona otro valor para realizar la búsqueda	
6.	Escoge un operador booleano (“AND”, por defecto)	
7.	Si desea agregar más condiciones para realizar la búsqueda, da clic en el botón “+”. Comienza de nuevo el paso 4	
8.		Agrega opciones de búsqueda para establecer un nuevo criterio de búsqueda, un nuevo valor y un operador booleano.
9.	Si desea eliminar una condición de búsqueda, da clic en el botón “-”.	
10.		Elimina la condición de búsqueda elegida, quitando el criterio y el valor de búsqueda así como el operador booleano.
11.	Da clic en el botón “Buscar”	
12.		Verifica que el los valores para realizar la búsqueda no estén vacíos
13.		Si hay un valor para buscar vacío, muestra un mensaje informándolo para que lo corrija
14.		Si los valores de búsqueda no son vacíos, se procede a buscar.
15.		Si no encuentra materiales que coincidan con las condiciones de búsqueda proporcionadas se muestra un mensaje informándolo para que establezca una nueva búsqueda y comienza de nuevo el paso 3.
16.		Si encuentra materiales que coincidan con las de búsqueda proporcionado se muestra un listado con los mismos.
17.	Obtiene un listado con los resultados de búsqueda	

Tabla 6: CU_4 Gestionar Selección

Caso de Uso		**Gestionar selección**
Descripción		Permite al actor agregar, visualizar, eliminar un registro de la selección
Actor		Consultor del catálogo, Usuario de la biblioteca
Resumen		El caso de uso inicia cuando el usuario desea agregar, visualizar, eliminar un registro de la selección, ya sea de una lista pública o privada.
Complejidad		Media
Prioridad		Media
Precondiciones		El actor debe haber creado una lista pública o privada y haber realizado una búsqueda en el catálogo en línea
Postcondiciones		El sistema muestra las listas con los registros seleccionados
Flujo de eventos		
Flujo básico <Gestionar selección>		
	Actor	**Sistema**
1.	Escoge Agregar, Visualizar o eliminar un registro de las listas	
2.		Si el actor decide Agregar, ver Sección 1: Agregar a Lista
3.		Si el actor decide visualizar una lista, ver Sección 2: Visualizar lista
4.		Si el actor decide eliminar de una lista, ver Sección 3: Eliminar de lista
Sección 1: Agregar a Listas		
Flujo Alterno <Agregar a Listas>		
1.	Da clic en el botón “Agregar a lista” que aparece en cada material que desee guardar en listas	
2.		Muestra las listas para que el usuario escoja en que lista desea guardar el material
3.	Selecciona la lista en la que guardará el material	

4.		Agrega el material a la lista seleccionada
Sección 2: Visualizar Lista		
Flujo Alterno <Visualizar Lista>		
1.	Dar clic en “Listas” en la barra de navegación	
2.		Muestra un modal con las listas existentes, ya sean públicas o privadas
3.	Da clic sobre una lista para ver su contenido	
4.		Muestra un listado con los materiales que se encuentran en la lista seleccionada por el usuario
Sección 2: Eliminar de Lista		
Flujo Alterno < Eliminar de Lista >		
1.	Da clic en la lista en la cual quiere eliminar elementos	
2.		Muestra un listado con los materiales que se encuentran en la lista seleccionada por el usuario
3.	Escoge el elemento que desee eliminar y da clic en el botón “-”	
4.		Elimina el elemento seleccionado de la lista y muestra los elementos restantes en la lista.

Tabla 7: CU_8 Gestionar lista de selección

Caso de Uso	**Gestionar lista de selección**
Descripción	Permite al actor agregar, visualizar, eliminar un registro de la lista de selección
Actor	Consultor del catálogo, Usuario de la biblioteca
Resumen	El caso de uso inicia cuando el usuario desea agregar, visualizar, eliminar un registro de la lista de selección.
Complejidad	Media
Prioridad	Media

Precondiciones	El actor debe haber realizado una búsqueda en el catálogo en línea	
Postcondiciones	El sistema muestra los elementos agregados a la lista de selección	
Flujo de eventos		
Flujo básico <Gestionar lista de selección>		
	Actor	**Sistema**
1.	Escoge Agregar, Visualizar o eliminar un registro de las listas	
2.		Si el actor decide Agregar, ver Sección 1: Agregar a lista de selección
3.		Si el actor decide visualizar una lista, ver Sección 2: Visualizar lista de selección
4.		Si el actor decide eliminar de una lista, ver Sección 3: Eliminar de la lista de selección
Sección 1: Agregar a lista de elección		
Flujo Alterno <Agregar a lista de selección>		
1.	Da clic en el botón “Agregar a selección” que aparece en cada material, en el listado de los resultados de la búsqueda	
2.		Agrega el elemento a la lista y muestra un mensaje informándolo al usuario
Sección 2: Visualizar lista de selección		
Flujo Alterno <Visualizar lista de selección>		
1.	Dar clic en “Selección” en la barra de navegación	
2.		Muestra un modal con los elementos agregados a la lista de selección.
3.		Si la lista de elementos está vacía muestra un mensaje informándolo.
4.	Da clic en “Cerrar”	
5.		Cierra el modal con los elementos de la lista de selección
Sección 2: Eliminar de la lista de selección		
Flujo Alterno <Eliminar de la lista de selección>		

1.	Visualiza la lista de selección	
2.		Muestra un listado con los materiales que se encuentran en la lista de selección.
3.	Escoge el elemento que desee eliminar de la lista de selección y da clic en el botón "-"	
4.		Elimina el elemento seleccionado de la lista y muestra los elementos restantes en la misma.
5.	Da clic en "Cerrar"	Cierra el modal que contiene los elementos de la lista de selección

Tabla 8: CU_10 Administrar Listas

Caso de Uso	**Administrar listas**	
Descripción	Permite al usuario crear listas de selección públicas o privadas. El usuario solo podrá editar y/o eliminar las listas creadas por él.	
Actor	Usuario de la biblioteca	
Resumen	El caso de uso inicia cuando el actor desea crear listas de selección, sean públicas o privadas para almacenar registros bibliográficos.	
Complejidad	Media	
Prioridad	Media	
Precondiciones	El actor debe haber iniciado sesión en el catálogo en línea	
Postcondiciones	El sistema muestra las listas de selección del usuario y las públicas	
Flujo de eventos		
Flujo básico <Administrar listas>		
	Actor	**Sistema**
1.	Da clic en "Listas" en la barra de navegación para acceder a las listas.	
2.		Si no hay listas creadas, se muestra un mensaje informándolo al actor
3.		Si el actor ha creado listas anteriormente, se muestra un listado con las mismas.
4.	Escoge crear o eliminar una lista	
5.		Si el actor decide crear una lista, ver Sección 1: Crear Listas

6.		Si el actor decide eliminar una lista, ver Sección 2: Eliminar lista
Sección 1: Crear Listas		
Flujo Alterno <Crear Listas>		
1.	Selecciona la opción Crear	
2.		Muestra un formulario para crear una lista de selección
3.	Entra el nombre de la lista	
4.	Escoge el tipo, si va a ser pública o privada	
5.	Da clic en el botón “Crear”	
6.		Comprueba que el campo del nombre no este vacío.
7.		Si el campo del nombre está vacío, muestra un mensaje informándolo y se repite desde el paso 3.
8.		Si el campo del nombre no está vacío, se crea la lista.
9.	Da clic en el botón “Cancelar”	
10.		Se cancela la operación y se cierra el formulario
Sección 2: Eliminar Lista		
Flujo Alterno <Eliminar Lista>		
1.	Selecciona la opción Eliminar, sobre una lista.	
2.		Comprueba que la lista a eliminar ha sido creada por el mismo usuario
3.		Si la lista fue creada por el mismo usuario, muestra un mensaje para confirmar y eliminar la lista.
4.		Si la lista no fue creada por el usuario, le muestra un mensaje informándole al mismo que no puede eliminar esa lista.

2.4 Arquitectura del software

La arquitectura de software es la estructura de un sistema o bien la forma en que va a estar organizado este; incluye los elementos que lo conforman, sus propiedades visibles desde el exterior y las relaciones que existen entre ellos (Pressman 1988).

Cuando se habla de arquitectura de software, se hace alusión a la especificación de la estructura del sistema, entendida como la organización de componentes y relaciones entre ellos; los requerimientos que debe satisfacer el mismo y las restricciones a las que está sujeto. La arquitectura está compuesta por propiedades no funcionales del sistema y su impacto sobre la calidad del mismo; las reglas y decisiones de diseño que gobiernan esta estructura y los argumentos que justifican las decisiones tomadas. Es la estructura del sistema e incluye los componentes fundamentales del mismo. Proporciona un marco de referencia para guiar de manera más organizada la construcción del software entre los analistas, diseñadores, programadores y demás miembros del equipo de desarrollo (Pressman 2010).

Los estilos arquitectónicos expresan la arquitectura del software en el sentido más formal y teórico. La familia de estilos utilizada en la investigación es la Arquitectura en Capas, como se muestra a continuación:

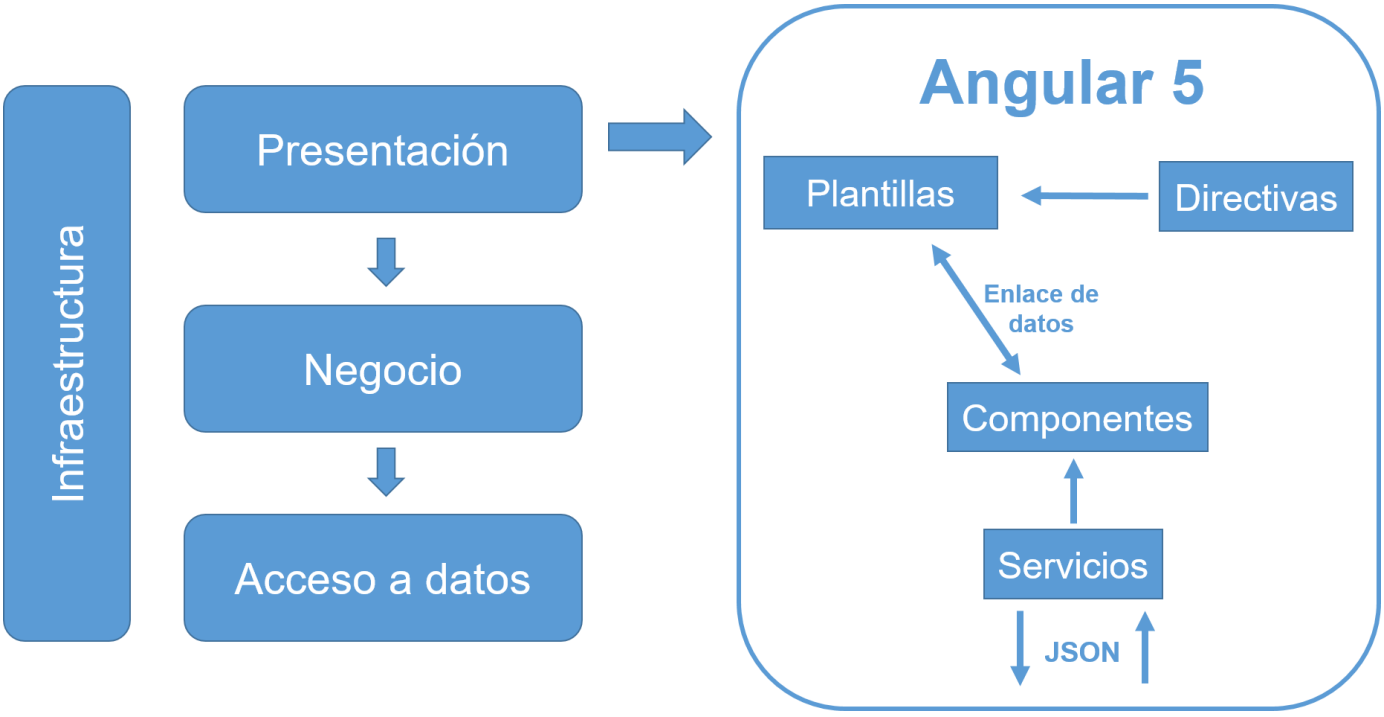

Figura 2 Arquitectura. Fuente: Elaboración propia

A continuación, se describen las capas principales de un patrón de arquitectura por capas (España y Fernando 2016):

Capa de presentación: Referente a la interacción entre el usuario y el software. Puede ser tan simple como un menú basado en líneas de comando o tan complejo como una aplicación basada en formas. Su principal responsabilidad es mostrar información al usuario, interpretar los comandos de este y realizar algunas validaciones simples de los datos ingresados.

Capa de negocio: También denominada Lógica de Dominio, esta capa contiene la funcionalidad que implementa la aplicación. Involucra cálculos basados en la información dada por el usuario y datos almacenados y validaciones. Controla la ejecución de la capa de acceso a datos y servicios externos. Se puede diseñar la lógica de la capa de negocios para uso directo por parte de componentes de presentación o su encapsulamiento como servicio y llamada a través de una interfaz de servicios que coordina la conversación con los clientes del servicio o invoca cualquier flujo o componente de negocio.

Capa de acceso a datos: Esta capa contiene la lógica de comunicación con otros sistemas que llevan a cabo tareas por la aplicación. Estos pueden ser monitores transaccionales, otras aplicaciones, sistemas de mensajerías. Para el caso de aplicaciones empresariales, generalmente está representado por una base de datos, que es responsable por el almacenamiento persistente de información.

Capa de infraestructura: Esta capa es adyacente a todas las demás. Comprende todos aquellos servicios susceptibles de ser adquiridos desde cualquiera de las capas lógicas del sistema. Está constituida por los nodos de red que realizan la conmutación y encaminamiento de paquetes.

En la capa de presentación de pueden identificar algunos de los bloques de construcción principales de la arquitectura de Angular:

Componentes: Los componentes controlan una sección de la vista de la aplicación. Definen la lógica de aplicación de un componente y la funcionalidad que cumple para la vista, dentro de una clase.

Plantillas: las plantillas definen la vista de un componente, éstas, contienen un fragmento de código HTML que le indica a Angular como renderizar el componente. Una plantilla se puede ver como HTML normal, solo con algunas diferencias.

Enlaces de datos: es un mecanismo para coordinar partes de una plantilla con partes de un componente. Agrega un marcado de enlace de datos a la plantilla HTML para decirle a Angular como conectar ambos lados.

Directivas: las directivas son básicamente funciones que son invocadas cuando el DOM (*Document Object Model*) es compilado por el framework de Angular. Existen dos tipos, las directivas estructurales que alteran el diseño, añadiendo, eliminando y reemplazando elementos en el DOM, y las directivas de atributo que alteran la apariencia o el comportamiento de un elemento existente.

Servicios: los servicios abarcan una amplia categoría que incluye cualquier valor, función o característica que la aplicación necesite.

2.5 Patrones de diseño

Los patrones de diseño figuran una estructura recurrente de componentes que se comunican entre sí para resolver un problema general de diseño que se repite en sistemas orientados a objetos. En él se describe el problema, la solución, cuándo se aplica la solución y las consecuencias que el uso del mismo pueda traer (Guerrero, Suárez y Gutiérrez 2013).

Patrones GoF

Los patrones definidos por GoF (por sus siglas en inglés, *Gang of Four*) agrupan los patrones de acuerdo a su propósito: creación, estructura y comportamiento. (Guerrero, Suárez y Gutiérrez 2013)

El patrón GoF que se utilizará en la propuesta de solución es:

Composite (Objeto compuesto): sirve para construir objetos complejos a partir de otros más simples y similares entre sí, gracias a la composición recursiva. Por ejemplo, se definen componentes para gestionar las plantillas HTML, al crear una interfaz, se dividen en varios componentes haciendo más sencillo el control de la vista.

Patrones GRASP

Los patrones GRASP (por sus siglas en inglés*, General Responsibility Assignment Software Patterns*) describen los principios fundamentales de la asignación de responsabilidades a objetos. El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos (Grosso 2011).

Los patrones GRASP a utilizar en la propuesta son:

Bajo acoplamiento: Permite reutilizar las funcionalidades de las distintas clases con un nivel de dependencia mínima. Este patrón se evidencia en todas las aplicaciones realizadas con Angular, pues cada componente de la aplicación tiene un propósito clave, que puede modificarse sin tener que afectar la implementación de otros componentes. Por ejemplo, se puede modificar una interfaz, cambiando solo la plantilla de uno de sus componentes sin tener que modificar el resto de los componentes.

Alta cohesión: Asigna responsabilidades de manera tal que la cohesión siga siendo alta, o sea que las funcionalidades de las clases estén altamente relacionadas de forma tal que exista una colaboración entre ellas para compartir el esfuerzo y no caiga todo el peso sobre una única clase. Usar este patrón simplifica el mantenimiento y favorece el bajo acoplamiento. Este patrón se utiliza en todas las clases y componentes.

2.6 Conclusiones parciales

Una vez culminadas las actividades de planificación y diseño de la propuesta de solución se obtuvo una visión más clara y detallada del sistema que se desea desarrollar. Los requerimientos funcionales y no funcionales obtenidos a partir del proceso de identificación de requisitos, constituyeron elementos claves en la construcción de la propuesta de solución. La utilización de los patrones de diseño permitió identificar aspectos importantes de la estructura del diseño del sistema web propuesto, lo que garantizó una mayor organización. Se generaron los artefactos de diseño acorde a la metodología seleccionada, constituyendo la base para la implementación del software.

Capítulo 3: Implementación y pruebas

Introducción

En el presente capítulo, se mostrarán los artefactos correspondientes a las etapas de implementación y prueba del sistema. De acuerdo a la metodología que se utiliza, se especifican, de los tipos de pruebas que esta plantea, cuáles serán empleadas para comprobar el funcionamiento del sistema.

3.1 Diagrama de despliegue

Un diagrama de despliegue modela la arquitectura de un sistema en tiempo de ejecución; muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del software se trazan en esos nodos. Las relaciones que existen entre nodos representan los protocolos de comunicación que se utilizan para acceder a cada uno.

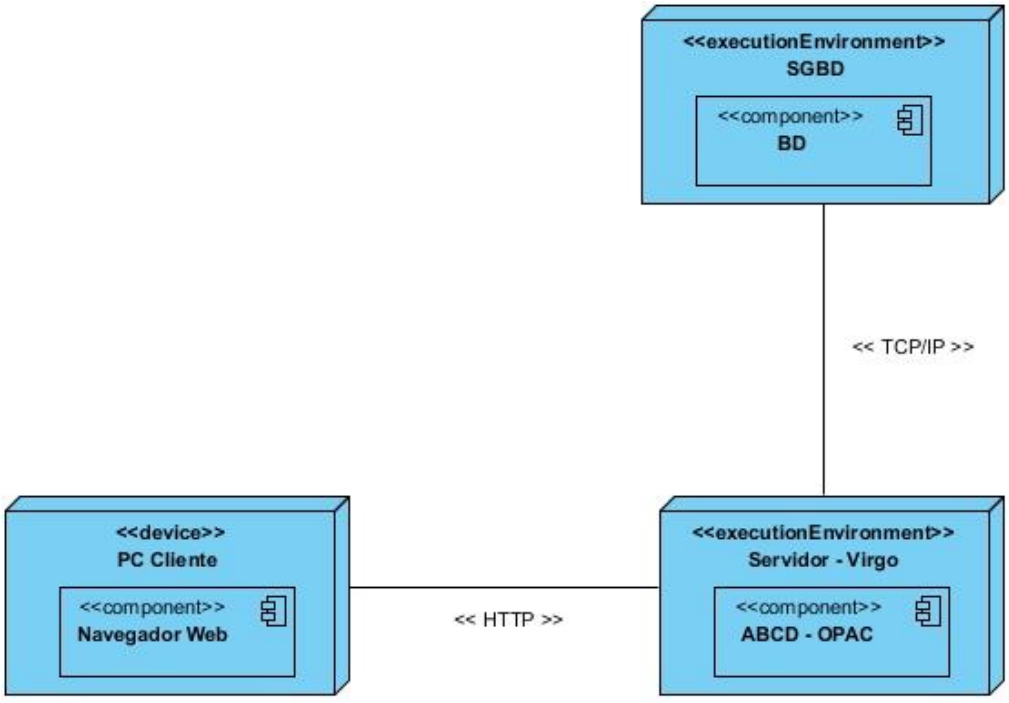

Figura 3 Diagrama de despliegue. Fuente: Elaboración propia

Se encuentra el nodo PC Cliente como contenedor del componente de software Navegador Web, el nodo Servidor – Virgo, donde se ejecuta el sistema ABCD, que contiene el módulo OPAC. También se encuentra el nodo SGBD, el cual contiene la base de datos.

El nodo PC Cliente representa las estaciones de trabajo de los usuarios que se conectan al sistema, las mismas realizan peticiones al Servidor de aplicaciones Virgo v3.6.3 a través del protocolo HTTP por el puerto 8080 el cual se conecta al servidor de bases de datos donde se almacena la base de datos del sistema, mediante el protocolo TCP/IP.

3.2 Estándares de codificación

Un estándar de codificación completo comprende todos los aspectos de la generación del código fuente de un software. Es preciso definir una serie de pautas que lleven por objetivo uniformar la estructura del código de manera tal que este sea lo más legible posible, como si un único programador hubiese escrito todo el código de una sola vez.
A continuación, se muestran los estándares de codificación que se utilizaron en la implementación:

3.2.1 Declaración de variables y constantes

La declaración de las variables, para una mayor comprensión del código, debe estar en el formato LowerCamelCase, donde el nombre comienza con letra minúscula, en caso de ser un nombre compuesto, la segunda palabra comenzará con letra inicial mayúscula.
Si se declaran variables que no van a cambiar su valor durante la ejecución de la aplicación, deberán declararse con *const.*
Ejemplo:

```
let autor: string = ‘Bob’;
let orderBy: string = ‘titulo’;
const urlId = `${this.url}/${id}`;
```

3.2.2 Métodos

Los nombres de los métodos deben estar escritos en el formato *LowerCamelCase. El cuerpo del método se delimitará con llaves {},* la de apertura en la misma línea donde se declara el método separado por un espacio, la llave de cierre debe estar en una línea diferente y no debe existir en esa línea ningún otro código que la propia llave de cierre.

Ejemplo:
```
orderByOption() {
    const orderBy = this.orderBy;
    this.result.sort((a, b) => a[orderBy].localeCompare(b[orderBy]));
  }
```

3.2.3 Clases e interfaces

Los nombres de las clases deben estar escritos en *UpperCamelCase*. La indentación de las mismas debe ser de una tabulación con respecto al margen izquierdo.

Ejemplo:
```
export interface Material {
                    id: string;
                    titulo: string;
                    autor: string;
                    fecha: string;
                    ubicacion: string;
                    tipo: string;
        }
     export class ResultadosService {
  constructor (private http: HttpClient) {}
}
```

3.3 Pruebas de software

Es el conjunto de actividades que se planean con anticipación y se realizan de manera sistemática. Por tanto es necesario definir un conjunto de pasos en que se puedan incluir técnicas y métodos específicos del diseño de casos de prueba. Las pruebas son el último bastión para la evaluación de la calidad y de manera más pragmática, el descubrimiento de errores (Pressman, 2011).

Las pruebas de software son un elemento crítico para la garantía de calidad del software, y representan una revisión final de las especificaciones del diseño y de la codificación. Son utilizadas para identificar posibles fallos de implementación, calidad o usabilidad de un programa.

La metodología que guía el proceso de desarrollo de la presente investigación propone realizar pruebas internas, de liberación y de aceptación. Sin embargo, para la solución propuesta solo se formalizarán las pruebas internas, y las de aceptación, puesto que se encargan de verificar el resultado de la implementación al probar cada construcción (Rodríguez, 2015). La otra prueba que propone la metodología requiere de una entidad certificadora y una aprobación por parte del cliente, elementos que no se contemplan dentro de los requerimientos del presente trabajo.

3.3.1 Estrategia de prueba

La estrategia de prueba es lo primero que se debe realizar antes de las pruebas de un *software*, ya que mediante esta queda plasmado los niveles de prueba a tratar, así como los tipos de pruebas que se deben llevar a cabo por cada nivel, los métodos de pruebas aplicar y las técnicas a utilizar por el método.

Una estrategia de prueba de *software* proporciona una guía que describe los pasos que deben realizarse como parte de la prueba, cuando se planean y se llevan a cabo dichos pasos, y cuanto esfuerzo, tiempo y recursos se requieran. Por tanto, cualquier estrategia de prueba debe incorporar la planificación de la prueba, el diseño de casos de prueba, la ejecución de la prueba y la recolección y evaluación de los resultados. Una estrategia de prueba de software debe ser suficientemente flexible para promover un uso personalizado de la prueba. Al mismo tiempo, debe ser suficientemente rígida para alentar la planificación razonable y el seguimiento de la gestión conforme avanza el proyecto.

3.3.2 Niveles de prueba

Las pruebas se aplican a diferentes tipos de destinos, en fases o niveles diferentes de esfuerzo de trabajo. Estos niveles suelen distinguirlos los roles que están más capacitados para diseñar y dirigir las pruebas, donde las técnicas son más adecuadas para realizar la prueba en cada nivel. Es importante asegurarse de hay un equilibrio entre los diferentes esfuerzos de trabajo. Existen varios niveles de pruebas como: pruebas de desarrollador, independiente, unidad, integración, sistema y aceptación, donde cada nivel contiene una técnica de prueba específica según los atributos de calidad que se deseen verificar.

A continuación, se muestran los niveles de pruebas que se aplicarán para comprobar que el sistema da respuesta a los requisitos funcionales definidos anteriormente:

- **Nivel de sistema:** en este nivel las pruebas tienen como propósito ejercitar profundamente el sistema para verificar que se han integrado todos los elementos del sistema (hardware, software) y que realizan las funciones adecuadas. Los casos de prueba a diseñar en este nivel de pruebas, deben cubrir los aspectos funcionales y no funcionales del sistema.
- **Nivel de integración:** en este nivel se prueba los componentes combinados para ejecutar un caso de uso. Además, se realiza la prueba para descubrir errores en las especificaciones de las interfaces de las clases. Se centra principalmente en probar la comunicación entre los componentes de un mismo sistema, comunicación entre sistemas o entre *hardware* y *software*.
- **Nivel de aceptación:** en este nivel se evidencian las pruebas realizadas por el usuario en un entorno muy similar al de producción para demostrar que el sistema cumple las especificaciones funcionales y requisitos. Son las últimas pruebas realizadas donde el cliente prueba el software y verifica que cumpla con sus expectativas, estas son fundamentales por lo cual deben incluirse obligatoriamente en el plan de pruebas de *software*.

3.3.3 Tipos de pruebas

Las pruebas funcionales son un proceso de control de calidad que consiste en asegurar el cumplimiento de un sistema con requerimientos funcionales. El objetivo principal de las pruebas funcionales es analizar el producto terminado y determinar si hace todo lo que debería hacer y si lo hace correctamente.

Las pruebas de unidad descomponen las funciones del programa en comportamientos comprobables discretos que se pueden probar como unidades individuales. Están destinadas a verificar las unidades más pequeñas del software. Se aplican a las funcionalidades para verificar que los flujos de control y datos están cubiertos y funcionan tal como se espera.

Las pruebas de integración son aquellas que se realizan en el ámbito del desarrollo de software una vez se han realizado las pruebas de unidad, se comprueba que todos los elementos que componen el sistema funcionan juntos correctamente, esta es su principal función, probar la comunicación entre los diferentes componentes.

Las pruebas de carga son el tipo más sencillo de pruebas de rendimiento. Una prueba de carga se realiza generalmente para observar el comportamiento de una aplicación bajo una cantidad de peticiones esperada. Esta carga puede ser el número esperado de usuarios concurrentes utilizando la aplicación y que realizan un número específico de transacciones durante el tiempo que dura la carga. Esta prueba puede mostrar los tiempos de respuesta de todas las transacciones importantes de la aplicación.

3.3.4 Métodos de pruebas

Se decide utilizar el método de caja negra, estas permiten obtener un conjunto de condiciones de entrada que ejerciten los requisitos funcionales, en ella se verifican las funcionalidades sin tomar en cuenta la estructura interna del código.

Las pruebas de caja negra son un enfoque complementario a las demás pruebas que intenta descubrir diferentes tipos de errores, entre los que se encuentran:

- Funciones incorrectas
- Errores de interfaz
- Errores de rendimiento.
- Errores de inicialización y terminación

Para el desarrollo de las pruebas de caja negra hay que tener en cuenta las diferentes técnicas que existen para este propósito (Terrera 2017):

- **Partición de equivalencias:** Consiste en clasificar las entradas de datos del sistema que presentan un comportamiento similar, por lo cual serán procesados de la misma forma. Se pueden definir particiones tanto para datos válidos como no válidos.
- **Análisis de valores borde:** Parte del principio que el comportamiento al borde de una partición de datos tiene mayores probabilidades de presentar errores. Los valores máximos y mínimos de una partición son los valores bordes.
- **Tablas de decisión:** Las tablas de decisión son una herramienta útil para documentar reglas de negocio de alta complejidad que el sistema debe cumplir, se crean a partir de la especificación funcional
- **Transición entre estados:** Una tabla de estados, muestra las relaciones entre los estados y las entradas de datos. Puede ayudar a identificar posibles transacciones inválidas.

- **Pruebas de casos de uso:** Los casos de uso describen las interacciones entre actores que producen un resultado que agrega algún valor. A partir de estos se pueden derivar casos de prueba.

La técnica que se va a utilizar es la de partición de equivalencia, ya que esta divide el dominio de entrada de un programa en clases de datos, a partir de los cuales se derivan los casos de pruebas.

3.3.5 Diseño de casos de prueba

Un caso de prueba es un conjunto de condiciones o variables bajo las cuales se determinará si una aplicación o un sistema de *software* es parcial o completamente satisfactoria.

Los casos de prueba incluyen las funciones que el programa es capaz de realizar. Se deben tener en cuenta el uso de todo tipo de datos de entrada/salida, cada comportamiento esperado, todos los elementos de diseño, y cada clase de defecto.

A continuación se muestran los casos de prueba para los casos de uso Autenticar usuario (ver Tabla 7) y Consultar Catálogo (ver Tabla 8) asi como la descripción de las variables (ver Tabla 9):

Tabla 9. Caso de Prueba Autenticar usuario.

Escenarios	Descripción	Variable 1	Variable 2	Respuesta del sistema	Flujo central
EC 1.1 Autenticar usuario correctamente	Permite al usuario acceder al catálogo en línea, proporcionando los datos correctamente	V	V	El usuario accede al catálogo en línea correctamente	1- Dar click en Iniciar sesión. 2-Introducir los datos de acceso (usuario y contraseña). 3- Presionar Enter o dar click en Aceptar para iniciar sesión
EC 1.2 Autenticar usuario con datos incorrectos	El usuario introduce datos incorrectos en el formulario de iniciar sesión	V	I	Muestra un mensaje de error: "Usuario/Contraseña incorrectos, rectifique los datos"	1- Dar click en Iniciar sesión. 2-Introducir los datos incorrectos de acceso. 3- Presionar Enter o dar click en Aceptar.
		I	V		
		I	I		
EC 1.3 Autenticar usuario con campos vacíos	El usuario deja campos vacíos en el formulario de iniciar sesión	V	I	Muestra un mensaje de error: "Existen campos obligatorios vacíos, complete el formulario"	1- Dar click en Iniciar sesión. 2-Introducir los datos incorrectos de acceso. 3- Presionar Enter
		I	V		
		I	I		

Fuente: Elaboración propia

Tabla 10. Caso de Prueba Consultar catálogo

Escenario	Descripción	Variable 1	Variable 2	Respuesta del sistema	Flujo central
EC 2.1 Consultar catálogo con datos correctos	Permite al usuario realizar búsquedas simples en el catálogo	N/A	V	El sistema muestra el listado con los elementos encontrados	1- Seleccionar un criterio de búsqueda 2-Introducir un valor para realizar la búsqueda 3- Presionar Enter o dar click en Buscar
EC 2.2 Consultar catálogo con campo vacío	El usuario deja un campo vacío en el formulario de búsqueda	N/A	I	Muestra un mensaje de error: "Debe proporcionar un valor para buscar, complete el formulario"	1- Presionar Enter o dar click en Aceptar

Fuente: Elaboración propia.

Tabla 11. Descripción de las variables de entrada

No.	Nombre de campo	Clasificación	Valor nulo	Descripción de las variables
1	usuario	campos de texto	no	Nombre de usuario para acceder al catálogo en línea
2	contraseña	campos de texto	no	Contraseña para acceder al catálogo en línea
3	valor de búsqueda	campos de texto	no	Valor que proporciona el usuario para buscar
4	criterio de búsqueda	campos de selección	no	Criterio de búsqueda que selecciona el usuario. (Su valor por defecto es Cualquier campo)

Fuente: Elaboración propia.

3.3.6 Resultados de las pruebas

Se realizaron pruebas unitarias al sistema usando el *test runner* Karma.js y Jasmine que permite escribir las pruebas, al ejecutar Karma.js no se tiene control en cuanto a la cantidad de iteraciones, pero se detectaron un total de nueve no conformidades, destacando incorrectas declaraciones de variables, parámetros incorrectos y comparación entre variables de diferentes tipos. Todas las no conformidades fueron resueltas.

Para validar el correcto funcionamiento del sistema se realizaron las pruebas de caja negra a través de los casos de prueba basados en los casos de uso. Este proceso permitió verificar el cumplimiento de los

requisitos funcionales del componente, donde los resultados de las pruebas que no fueron satisfactorios pasaron a ser no conformidades.

A continuación, se muestra la tabla (ver Tabla 10) con los datos de las no conformidades encontradas en cada iteración. En la primera iteración se identificaron once no conformidades, de ellas, dos de validación, una de ortografía, cinco funcionales y tres de la interfaz. En la segunda iteración se identificaron dos no conformidades, ambas funcionales y fueron resueltas. En la tercera iteración no se encontraron no conformidades. Todas las no conformidades fueron resueltas en su totalidad.

Tabla 12. Resultados de las pruebas de funcionalidad

No. De iteraciones	Total de no conformidades	Sin resolver	Resueltas
Primera Iteración	11	0	11
Segunda Iteración	2	0	2
Tercera Iteración	0	0	0

Fuente: Elaboración propia

Se realizaron pruebas de carga para observar el comportamiento de una aplicación bajo una cantidad de peticiones esperada. Se utilizó *Artillery*, una herramienta implementada en NodeJS que a través de la declaración de un fichero de configuración permite definir, entre otras cosas, el número de usuarios y peticiones que se van a simular y registrar los resultados de latencia y duración para el mínimo, el máximo, la mediana, el percentil 95 y el percentil 99.

Se utilizó la siguiente línea de comandos para simular 200 usuarios, haciendo 20 peticiones cada uno:
artillery quick --count 200 -n 20 http://localhost:4040/materiales/
y se obtuvieron los siguientes resultados:

```
All virtual users finished
Summary report @ 21:19:34(-0400) 2018-06-8
  Scenarios launched:  200
  Scenarios completed: 200
  Requests completed:  4000
  RPS sent: 611.62
```

```
Request latency:
  min: 4
  max: 250.4
  median: 84
  p95: 171
  p99: 203.9
Codes:
  200: 4000
```

Lo que significa:

Scenarios launched: es la cantidad de usuarios virtuales creados en los 10 segundos anteriores

Scenarios completed: es la cantidad de usuarios virtuales que completaron sus escenarios en los 10 segundos anteriores

Requests completed: es la cantidad de solicitudes y respuestas HTTP enviadas

RPS sent: es la cantidad promedio de solicitudes por segundo completadas en los 10 segundos anteriores (o durante toda la prueba)

Request latency: La latencia de solicitud y la duración del escenario se expresan en milisegundos (mínimo, máximo y mediana), y los valores p95 y p99 son los valores percentiles 95 y 99 (un valor de latencia de solicitud p99 de 203.9ms significa que 99 de 100 solicitudes tardaron 203.9ms o menos en completarse).

Codes: representa los códigos devueltos y la cantidad de solicitudes.

3.4 Conclusiones del capítulo

A lo largo del proceso de implementación el uso de los estándares de codificación definidos contribuyó a obtener una adecuada uniformidad y legibilidad en el código fuente. Se obtuvo el diagrama de despliegue, el cual contribuyó a comprender la arquitectura de un sistema en tiempo de ejecución. Se conformaron los casos de pruebas para realizar las mismas a través del método de caja negra, aplicando la técnica de partición equivalente. A partir de las pruebas unitarias se obtuvieron y corrigieron las no conformidades referentes al código fuente. Al culminar el proceso de prueba se encontraron un total de veintiuna no conformidades, las cuales fueron resueltas, lo cual posibilitó cumplir con los requisitos funcionales del sistema.

Conclusiones

La investigación desarrollada y los resultados obtenidos permiten arribar a las siguientes conclusiones:

- Se analizaron los principales conceptos relacionados con el objeto de estudio y el campo de acción, lo que permitió entender el proceso de recuperación de los registros bibliográficos en los SIGB, obteniendo la base teórica necesaria para el desarrollo de la solución
- A partir del análisis y diseño del catálogo en línea se obtuvieron los artefactos que exige la metodología que guía el proceso de desarrollo del mismo.
- Como resultado de la implementación se obtuvo un catálogo en línea con un diseño adaptativo que permite la búsqueda facetada, dando cumplimiento al objetivo general de la investigación.
- La ejecución de las pruebas de software permitió resolver las no conformidades encontradas, lo que garantiza el correcto funcionamiento del catálogo en línea.

Referencias Bibliográficas

ABADAL, E. y CODINA, L., 2005. Recuperación de información. *Datos Documentales: Características, funciones y método*, pp. 29-92.

AJA, L., 2012. Gestión de información, gestión del conocimiento y gestión de la calidad en las organizaciones. *Acimed*, vol. 10.

ARIAS, J.J. y CALVACHE, C., 2016. Systematic review about the integration of model-driven software development and agile methodologies. *Informador Técnico*, vol. 80, no. 1, pp. 87.

ARIOLA, O. y BUTRÓN, K., 2008. Sistemas integrales para la automatización de bibliotecas basados en software libre. *Acimed*, vol. 18, no. 6.

ARRIOLA, Ó. y MONTES DE OCA, E., 2014. Sistemas Integrales de Automatización de Bibliotecas : una descripción suscinta. *Bibliotecas y Archivos*, vol. 1, no. 3, pp. 47-76.

ASCENCIO, G., 2015. ABCD Wiki. [en línea]. Disponible en: http://abcdwiki.net/wiki/es/index.php.

BROWN, K. y ARIAS, A., 2015. *Diseño e implementación del módulo Catálogo en Línea de Acceso Público del sistema ABCD 3.0*. S.l.: Universidad de las Ciencias Informáticas, La Habana.

CHÁFER, S., 2015. Panorámica general del uso de Sistemas Integrados de Gestión Bibliotecaria en España. *Métodos de información*, vol. 6, no. 10, pp. 43-56.

CÓRDOBA, G., 2015. *Implementación de un sistema integrado de gestión bibliotecaria (sigb) en la biblioteca Juan Alberto Aragón Bateman de la fundación universitaria Konrad Lorenz de la ciudad de Bogotá.* S.l.: Universidad del Quindío.

DE WOLFF, I. y HAGE, J., 2017. Refining types using type guards in TypeScript. *Proceedings of the 2017 ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation*. S.l.: ACM, pp. 111-122.

DÍAZ, I., SÁNCHEZ, J. y PASTOR, O., 2005. *Metamorfosis: Un marco para el análisis de requisitos funcionales.* 2005. S.l.: s.n.

ECKSTEIN, R., 2007. Java se application design with mvc. *Sun Microsystems Inc*,

ESPAÑA, S. y FERNANDO, H., 2016. *Documentación y análisis de los principales frameworks de arquitectura de software en aplicaciones empresariales*. S.l.: Facultad de Informática.

FLORES, F., 2011. *El Software De Código Abierto: Una Alternativa Para La Gestión Integral De La Biblioteca*. S.l.: Escuela Nacional de Biblioteconomía y Archivonomía.

GARCIA MELERO, L.A. y GARCIA CAMARERO, E., 1988. Automatización de bibliotecas. *Boletín de la ANABAD*. S.l.: Anabad, pp. 393-410. ISBN 84-7635-351-0.

GOOGLE, 2016. Angular. [en línea]. [Consulta: 11 enero 2018]. Disponible en: https://angular.io/.

GROSSO, A., 2011. Patrones GRASP. *Prácticas de Software*.

GUERRERO, C. y MARTÍN, D., 2016. Recopilar requisitos. ,

GUERRERO, C., SUÁREZ, J. y GUTIÉRREZ, L., 2013. Patrones de Diseño GOF (The Gang of Four) en el contexto de Procesos de Desarrollo de Aplicaciones Orientadas a la Web. *Información tecnológica*, vol. 24, no. 3, pp. 103-114.

HERNÁNDEZ, A. y GARCÍA, M.A., 2002. Intranets y preservación digital, algo más que tecnología. *Boletín del Instituto Andaluz del Patrimonio Histórico*, no. 38, pp. 237-243.

JIMÉNEZ, A. y GUTIÉRREZ, M., 2010. Open source software as an alternative in developing IT systems on the native language of the population. The case of OpebBiblio nahuatl version. , pp. 2-7.

LAFFERTY, J. y CHENGXIANG, Z., 2017. Document language models, query models, and risk minimization for information retrieval. *ACM SIGIR Forum*, vol. 51, no. 2, pp. 251-259.

LARMAN, C., 2003. *UML y Patrones*. S.l.: s.n.

MADHUSUDHAN, M. y SINGH, V., 2016. Integrated library management systems: Comparative analysis of Koha, Libsys, NewGenLib, and Virtua. *The Electronic Library*, vol. 34, no. 2, pp. 223-249.

MARTÍN, C., 2008. SIGB: Catálogos y gestión de autoridades. Diseño y prestaciones de OPACs. *Recuperado de http://eprints. rclis. org/13188/1/sigb. pdf*,

MASON, J., 2006. *Armas, gérmenes y acero: breve historia de la humanidad en los últimos trece mil años*. Barcelona: Debate Editorial. ISBN 84-8306-667-X.

MIRALLES, M., 2009. El catálogo: un recurso en expansión. *Anales de Documentación* [en línea]. [Consulta: 7 diciembre 2017]. Disponible en: http://www.redalyc.org/articulo.oa?id=63511932004.

NODE.JS FOUNDATION, 2018. Node.js. [en línea]. [Consulta: 12 diciembre 2017]. Disponible en: https://nodejs.org/en/.

OLIVERT, P., 2015. Análisis de sistemas integrados de gestión bibliotecaria de acceso abierto: examen de tres casos concretos. *Métodos de información*, vol. 6, no. 10, pp. 29-42.

PARLIAMENTARY OFFICE OF SCIENCE AND TECHNOLOGY, 2005. Open source software. . London:

PRESSMAN, R., 1988. *Ingeniería del software*. S.l.: McGraw Hill. ISBN 9786071503145.

PRESSMAN, R., 2010. *Ingeniería del software. Un enfoque práctico*. Edición 7. S.l.: MCGRAW-HILL. ISBN 9786071503145.

RAE, 2017. Real Academia de la Lengua Española. [en línea]. Disponible en: http://dle.rae.es/.

RIOS, A.B., 2003. *La estructura conceptual del registro bibliográfico: análisis de la funcionalidad de las «Reglas de Catalogación» españolas y del formato IBERMARC bibliográfico.* S.l.: Universidad de Salamanca.

RIVERO, J.L. y GALARZA, J., 2015. La gestión de la información y el conocimiento: una oportunidad para las instituciones de Educación Superior. *Revista Universidad y Sociedad*, vol. 7, no. 2, pp. 16-22.

RODRÍGUEZ, T., 2015. Metodología de desarrollo para la Actividad productiva de la UCI. . La Habana, Cuba:

SARMA, G., 2016. OPAC Module in Open Source Library Management Software: A Comparative Study.

DESIDOC Journal of Library & Information Technology, vol. 36, no. 1.

SENSO, J.A., 2011. Manual de uso de Open Journal Systems. ,

SOMERVILLE, I., 2007. *Software Engineering*. 8. S.l.: Addison-Wesley.

SOMMERVILLE, I., ARAKAKI, R. y MELNIKOFF, S., 2008. *Engenharia de software*. S.l.: Pearson Prentice Hall.

TORRES, L., 2015. La gestión de información y la gestión del conocimiento. *Revista Archivo Médico de Camagüey*, vol. 19, no. 2, pp. 96-98.

VIDAL, M.J. y ARAÑA, A.B., 2012. Gestión de la información y el conocimiento. *Educación Médica Superior*, vol. 26, no. 3, pp. 474-484.

VILLÉN-RUEDA, L., 2006. Indexing and subject retrieval in Spanish library opacs: two decades of evaluation? *El profesional de la Información*, vol. 15, no. 2, pp. 87-98.

WHITE, B., 2012. La función que desempeñan las bibliotecas para garantizar el acceso a los conocimientos. *Revista de la OPMI*, no. 4.

Bibliografía

ALONSO, H. & SOLER, D., 2010. *Desarrollo del Módulo de Reportes del Sistema Integrado de Gestión Bibliotecaria Koha para la Biblioteca Nacional de Cuba «José Martí».* S.l.: Universidad de las Ciencias Informáticas (UCI).

ALVAREZ, E.B., ALVAREZ, M. del C.V. y VIDOTTI, S.A.B.G., 2015. La red de bibliotecas de la Universidad de La Habana: camino a una nueva filosofía de trabajo. *Revista Interamericana de Bibliotecología*, vol. 38, no. 2, pp. 147-158. ISSN 0120-0976. DOI 10.17533/udea.rib.v38n2a06.

ARIAS, J.J. y CALVACHE, C., 2016. Systematic review about the integration of model-driven software development and agile methodologies. *Informador Técnico*, vol. 80, no. 1, pp. 87.

ARIOLA, O. y BUTRÓN, K., 2008. Sistemas integrales para la automatización de bibliotecas basados en software libre. *Acimed*, vol. 18, no. 6.

ARRIOLA, Ó. y MONTES DE OCA, E., 2014. Sistemas Integrales de Automatización de Bibliotecas : una descripción suscinta. *Bibliotecas y Archivos*, vol. 1, no. 3, pp. 47-76.

Biblioteca de la Universidad de Sevilla. *Normas de préstamo* [en línea], 2014. Disponible en: http://bib.us.es/.

BLANCO ENCINOSA, L.J., 2011. *La informática en la dirección de empresas*. La Habana: Editorial Félix Varela.

BREEDING, M., 2015. Informe sobre sistemas bibliotecarios. Hacer operativa la innovación. *El profesional de la Información*, vol. 24, no. 4, pp. 485-496.

BREEDING, M., 2017. Library system report 2017. Competing visions for tecnology, openness, and workflow. *El profesional de la Información*, vol. 26, no. 3, pp. 543-557.

CASTAÑEDA, E.R., 2012. *«Sistema para el Manejo de Órdenes de Servicio por los especialistas del Centro de Ingeniería Clínica y Electromedicina».* S.l.: Universidad de las Ciencias Informáticas (UCI).

CHACÓN, J.C.R., 2006. Aplicación de la metodología RUP para el desarrollo rápido de aplicaciones basado

en el estándar j2ee. ,

CHÁFER, S., 2015. Panorámica general del uso de Sistemas Integrados de Gestión Bibliotecaria en España. *Métodos de información*, vol. 6, no. 10, pp. 43-56.

COMER, D.E., [sin fecha]. *Redes globales de informacih con Internet y TCPAP Principios bhsicos , protocolos y arquitectura*. S.l.: s.n.

CÓRDOBA, G., 2015. *Implementación de un sistema integrado de gestión bibliotecaria (sigb) en la biblioteca Juan Alberto Aragón Bateman de la fundación universitaria Konrad Lorenz de la ciudad de Bogotá.* S.l.: Universidad del Quindío.

CRUZ, I. & FERNÁNDEZ, N., 2011. *Desarrollo del módulo de Reportes para el Sistema Integrado de Gestión de la Biblioteca del Archivo de la Nación «Fransisco de Miranda» en Venezuela*. S.l.: Universidad de las Ciencias Informáticas (UCI).

DE WOLFF, I. y HAGE, J., 2017. Refining types using type guards in TypeScript. *Proceedings of the 2017 ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation*. S.l.: ACM, pp. 111-122.

DÍAZ, I., SÁNCHEZ, J. y PASTOR, O., 2005. *Metamorfosis: Un marco para el análisis de requisitos funcionales.* 2005. S.l.: s.n.

ECKSTEIN, R., 2007. Java se application design with mvc. *Sun Microsystems Inc*,

ESPAÑA, S. y FERNANDO, H., 2016. *Documentación y análisis de los principales frameworks de arquitectura de software en aplicaciones empresariales*. S.l.: Facultad de Informática.

FEUS, P.Y., 2009. *Adaptación del Módulo de Circulación del Sistema Integrado de Gestión Bibliotecaria Koha a la Biblioteca Nacional de Cuba «José Martí».* S.l.: Universidad de las Ciencias Informáticas (UCI).

GOOGLE, 2016. Angular. [en línea]. [Consulta: 11 enero 2018]. Disponible en: https://angular.io/.

GROSS, K. & MARTÍNEZ, L., 2013. *Desarrollo del módulo para la gestión de notificaciones del Portal*

Educativo. S.l.: Universidad de las Ciencias Informáticas (UCI).

GRUPO DE TRABAJO DE CATÁLOGO COLECTIVO DE LAS BIBLIOTECAS DE LA ADMINISTRACIÓN GENERAL DEL ESTADO, 2010. Informe de evaluación del sistema integrado de gestión de bibliotecas Koha para las Bibliotecas de la Administración General del Estado. , pp. 88.

GRUPO DE TRABAJO DE CATÁLOGO COLECTIVO DE LAS BIBLIOTECAS DE LA ADMINISTRACIÓN GENERAL DEL ESTADO, 2017. Kobli : Sistema integrado de gestión bibliotecaria de fuentes abiertas. [en línea]. [Consulta: 10 enero 2018]. Disponible en: http://kobli.bage.es/.

GUTIÉRREZ, C. y PALMA, A., 2017. Elige tu SIGB. *SIGB* [en línea]. [Consulta: 13 diciembre 2017]. Disponible en: http://sigbescolar.webcindario.com/koha.html.

INTRODUCCIÓN, I., 2013. SISTEMAS INTEGRADOS DE AUTOMATIZACIÓN DE BIBLIOTECAS . SITUACIÓN ACTUAL Y TENDENCIAS DE FUTURO . III . LOS SISTEMAS INTEGRADOS DE GESTIÓN BIBLIOTECARIA (SIGB). , pp. 1-8.

MADHUSUDHAN, M. y SINGH, V., 2016. Integrated library management systems: Comparative analysis of Koha, Libsys, NewGenLib, and Virtua. *The Electronic Library*, vol. 34, no. 2, pp. 223-249.

NODE.JS FOUNDATION, 2018. Node.js. [en línea]. [Consulta: 12 diciembre 2017]. Disponible en: https://nodejs.org/en/.

OLIVERT, P., 2015. Análisis de sistemas integrados de gestión bibliotecaria de acceso abierto: examen de tres casos concretos. *Métodos de información*, vol. 6, no. 10, pp. 29-42.

PRESSMAN, R., 2010. *Ingeniería del software. Un enfoque práctico*. Edición 7. S.l.: MCGRAW-HILL. ISBN 9786071503145.

RIVERO, J.L. y GALARZA, J., 2015. La gestión de la información y el conocimiento: una oportunidad para las instituciones de Educación Superior. *Revista Universidad y Sociedad*, vol. 7, no. 2, pp. 16-22.

RODRÍGUEZ, T., 2015. Metodología de desarrollo para la Actividad productiva de la UCI. . La Habana, Cuba:

RUSSELL, STUART & NORVIG, P., 2005. *AI: A Modern Approach*. 3era. S.l.: s.n.

SANTANA, S., 2011. El modelo FRBR en el servicio de referencia y su implementación en los catálogos automatizados. *Acimed*, vol. 22, no. 3, pp. 189-203.

SARMA, G., 2016. OPAC Module in Open Source Library Management Software: A Comparative Study. *DESIDOC Journal of Library & Information Technology*, vol. 36, no. 1.

TORRES, L., 2015. La gestión de información y la gestión del conocimiento. *Revista Archivo Médico de Camagüey*, vol. 19, no. 2, pp. 96-98.

VEGA, M. y CASTRO, E., 2005. Recuperación por sistemas de clasificación en los OPACs de las bibliotecas universitarias españolas. *Boletín de la ANABAD*, vol. 49, no. 1, pp. 161-174.

VIDAL, M.J. y ARAÑA, A.B., 2012. Gestión de la información y el conocimiento. *Educación Médica Superior*, vol. 26, no. 3, pp. 474-484.

VILLÉN-RUEDA, L., 2006. Indexing and subject retrieval in Spanish library opacs: two decades of evaluation? *El profesional de la Información*, vol. 15, no. 2, pp. 87-98.

WHITE, B., 2012. La función que desempeñan las bibliotecas para garantizar el acceso a los conocimientos. *Revista de la OPMI*, no. 4.

Anexos

Anexo 1: Entrevista

Entrevista	
Objetivos	➢ Definir los requisitos funcionales y no funcionales del sistema. ➢ Definir la arquitectura del componente. ➢ Definir las características del catálogo en línea.
Personas entrevistadas	Ing. Alberto Alejandro Arias Benítez Ing. Leandro Tabares Martín
Lugar de la entrevista	Laboratorio 101, Docente Julio Antonio Mella, UCI .
Modo de realización	Diálogo
Aspectos a tener en cuenta	➢ Arquitectura del sistema ABCDv3.0. ➢ Características del componente a desarrollar.

Preguntas abordadas en las entrevistas:

1) ¿Qué arquitectura presenta el sistema ABCD v3.0?
2) ¿Cuáles son las características del componente que se desea desarrollar?
3) ¿Qué limitaciones presenta el actual catálogo en línea del sistema ABCD v3.0?
4) ¿Cuáles son los requisitos funcionales que debe cumplir el nuevo catálogo en línea?
5) ¿Qué metodología de desarrollo de software emplea el proyecto ABCD?

Anexo 2: Matriz de trazabilidad Requisitos_Casos de Prueba

Tabla 13: Matriz de trazabilidad Requisitos_Casos de Prueba

	CP_1: Realizar búsqueda	CP_2: Realizar búsqueda	CP_3: Ordenar resultados	CP_4: Gestionar selección	CP_5: Evaluar material	CP_6: Recomendar material	CP_7: Visualizar detalles de un material	CP_8: Gestionar lista de	CP_9: Administrar listas	CP_10: Autenticar usuario	CP_11: Gestionar	CP_12: Listar Reservas	CP_13: Gestionar	CP_14: Visualizar historial	CP_15. Filtrar resultados
RF_1: Realizar búsqueda simple	X														
RF_2: Realizar búsqueda avanzada		X													
RF_3: Ordenar resultados			X												
RF_4: Gestionar selección				X											
RF_5: Evaluar material					X										
RF_6: Recomendar material						X									
RF_7: Visualizar detalles de un material							X								
RF_8: Gestionar lista de selección								X							
RF_9: Administrar listas									X						
RF_10: Autenticar usuario										X					
RF_11: Gestionar reservación											X				
RF_12: Listar Reservas												X			
RF_13: Gestionar sugerencias													X		
RF_14: Visualizar historial del usuario														X	
RF_15. Filtrar resultados															X

Anexo 3: *Matriz de trazabilidad Requisitos_Requisitos*

Tabla 14: Matriz de trazabilidad Requisitos_Requisitos

	RF_1: Realizar búsqueda	RF_2: Realizar búsqueda avanzada	RF_3: Ordenar resultados	RF_4: Gestionar selección	RF_5: Evaluar material	RF_6: Recomendar material	RF_7: Visualizar detalles de un material	RF_8: Gestionar lista de	RF_9: Administrar listas	RF_10: Autenticar usuario	RF_11: Gestionar	RF_12: Listar Reservas	RF_13: Gestionar	RF_14: Visualizar historial del usuario	RF_15. Filtrar resultados
RF_1: Realizar búsqueda simple	X														
RF_2: Realizar búsqueda avanzada		X													
RF_3: Ordenar resultados	X	X	X												
RF_4: Gestionar selección				X											
RF_5: Evaluar material					X										
RF_6: Recomendar material						X									
RF_7: Visualizar detalles de un material							X								
RF_8: Gestionar lista de selección								X							
RF_9: Administrar listas									X						
RF_10: Autenticar usuario										X					
RF_11: Gestionar reservación											X				
RF_12: Listar Reservas												X			
RF_13: Gestionar sugerencias													X		
RF_14: Visualizar historial del usuario														X	
RF_15. Filtrar resultados															X