UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 3

Grupo de Investigación de Web Semántica

Integración de datos y metadatos desde revistas científicas en Acceso Abierto

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

Autor:
Juan Gabriel León Herrera

Tutores:
MSc. Yusniel Hidalgo Delgado
Ing. Alejandro J. Mariño Molerio
MSc. Mailen Edith Escobar Pompa

La Habana, 20 junio de 2017
"Año 59 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaro ser el autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____	_____	_____
Juan Gabriel León Herrera	MSc. Yusniel Hidalgo Delgado	Ing. Alejandro Jesús Mariño Molerio
Autor	Tutor	Tutor

_____

MSc. Mailen Edith Escobar Poma

Tutor

DATOS DE CONTACTO

Síntesis del Tutor

El MSc. Yusniel Hidalgo Delgado se graduó con título de oro en la Universidad de Ciencias Informáticas en el año 2010. En su primer año de adiestramiento ocupó diversos roles dentro del proyecto de desarrollo del ERP Cubano. Actualmente se desempeña como profesor asistente del departamento docente de técnicas de programación de la Facultad 3. Coordinador del grupo de investigación de Web Semántica de la UCI. Es miembro de la Asociación Cubana de Reconocimiento de Patrones, de la Sociedad Cubana de Matemática y Computación y de la International Association for Pattern Recognition.

A mis queridos padres, por habérmelo dado todo.

A mi hermana por existir y ser mi ejemplo a seguir.

A M., que aunque estés lejos me diste la fuerza para alcanzar este momento.

Juan Gabriel León Herrera

AGRADECIMIENTOS

Primero que todo agradezco a Dios por permitirme llegar hasta aquí y cumplir con uno de mis sueños.

Agradezco a la Revolución Cubana, por brindarme la posibilidad de estudiar y superarme en este hermoso y magnífico proyecto que es hoy la Universidad de las Ciencias Informáticas.

Agradezco al Comandante en Jefe Fidel Castro Ruz por haber creado esta universidad y aunque no esté con nosotros físicamente puede contar con nosotros.

A mi madre Genma Herrera Curbelo, por ser la mejor madre del mundo. Sin ti el camino que he transitado hubiese sido bien difícil, con su entera compañía y su continuo apoyo todo resultó muy fácil.

A mi padre Juan René León Armas, por ser el mejor padre del mundo.

A mi hermanita Inés Danielly León Herrera, por ser ejemplo para mí y apoyarme en todos los momentos que lo necesite.

A M. desde que te conocí me has dado fuerza para alcanzar este momento tan especial para mí y siempre has estado a mi lado aunque no físicamente.

A todos mis tíos y tías, principalmente a mi tía Aida que no se encuentra ya entre nosotros, por su apoyo en todo momento.

A toda mi familia en general por estar siempre ahí cada vez que lo necesitaba.

A mis tutores MSc. Yusniel Hidalgo Delgado que desde China me supo guiar, a Ing. Alejandro Jesús Marino Molerio por su excelente, impecable y certera guía en la

AGRADECIMIENTOS

realización de este trabajo y también a la MSc. Mailen Edith Escobar Pompa por todo su apoyo.

Al Consejo de Dirección, por su guía y formación.

A los profesores que ayudaron de una forma u otra en mi formación como profesional.

A todos mis amigos de la FEU, Yoandri, Arlenis, Arlene, Yoelvis, Bienvenido, Alvaro, Julio, Eddy, Javier, Rosalia, Rolando, Yamila, Denet, Claudia, Dory, Sandra y tantos otros con los que viví tan buenos y malos momentos.

A todos los amigos con los que compartí durante estos años.

A los colegas del grupo, que en muchísimas ocasiones necesite de ellos y siempre estuvieron ahí.

Juan Gabriel León Herrera

RESUMEN

En la Universidad de las Ciencias Informáticas, se encuentra en ejecución un proyecto de investigación con el objetivo de construir una biblioteca digital semántica que integre datos y metadatos de revistas científicas y actas de congresos. En el marco de este proyecto, fueron desarrolladas dos herramientas de software: MetHarto y PDFCrawler. La primera de ellas es un recolector de metadatos bibliográficos utilizando el protocolo OAI-PMH. La segunda, extrae y almacena localmente artículos científicos en formato PDF existentes en cualquier revista científica en Acceso Abierto. No obstante, ambas herramientas aún no han sido integradas. Esta situación constituye un problema de integración de datos y metadatos, al no contar con un proceso estandarizado para este propósito. Esto significa que actualmente, no es posible establecer un vínculo entre los metadatos obtenidos por MetHarto y los artículos en formato PDF extraídos por PDFCrawler.

Palabras claves: herramienta; metadatos bibliográficos; revistas científicas; pdf.

ABSTRACT

At the University of Computer Science, a research project is underway to build a digital semantic library that integrates data and metadata from scientific journals and conference proceedings. Within the framework of this project, two software tools were developed: MetHarto and PDFCrawler. The first one is a collector of bibliographic metadata using the OAI-PMH protocol. The second locally extracts and stores scientific articles in PDF format existing in any scientific journal in Open Access. However, both tools have not yet been integrated. This situation constitutes a problem of integration of data and metadata, as there is no standardized process for this purpose. This means that it is not currently possible to establish a link between MetHarto metadata and PDFCrawler extracted PDF articles.

Keywords: *bibliographic metadata, scientific magazines, tool, pdf.*

ÍNDICE

INTRODUCCIÓN 1
CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA 6
1.1. Introducción 6
1.2. Análisis bibliométrico y documental 6
1.3. Marco teórico 7
1.4. Estado del arte 12
1.5. Conclusiones parciales 20
CAPÍTULO 2. DESCRIPCIÓN DE LA PROPUESTA 21
2.1. Introducción 21
2.2. Enfoque propuesto 21
2.3. Herramientas utilizadas 22
2.4. Entorno de desarrollo 25
2.5. Requisitos 27
2.6. Arquitectura del componente 29
2.7. Estándares de código 30
2.8. Planificación de pruebas 31
2.9. Conclusiones parciales 33
CAPÍTULO 3. VALIDACIÓN DE LA PROPUESTA 34
3.1. Pruebas de software 34
3.2. Caso de estudio 38
3.3. Diseño experimental 38
3.4. Análisis de los resultados 40
3.5. Conclusiones parciales 42
CONCLUSIONES GENERALES 43
RECOMENDACIONES 44
REFERENCIAS BIBLIOGRÁFICAS 45

ÍNDICE DE TABLAS

Tabla 1 Resumen de la revisión bibliográfica consultada.

Tabla 2 Herramientas para la extracción de metadatos bibliográficos desde documentos PDF.

Tabla 3: Resultados (A100: Primera evaluación con 100 artículos, B100: Segunda evaluación con 100 artículos, B1153: Segunda evaluación con 1153 artículos), (Lipinski, et al. 2013)

Tabla 4 Valores de Distancia de Jaro para diferentes cadenas correspondientes a nombres y apellidos

Tabla 5 Valores de Distancia de Jaro-Winkler para diferentes cadenas

Tabla 6 Distancias de Levenshtein para diferentes cadenas correspondientes a nombres y apellidos

Tabla 7: Requisitos funcionales.

Tabla 8 Relación de los requisitos no funcionales de software.

Tabla 9 Relación de los requisitos no funcionales de hardware.

Tabla 10 Caminos independientes identificados en el grafo de flujo del método loadConfigFile().

Tabla 11 Caso de prueba para el camino básico #1

Tabla 12 Caso de prueba para el camino básico #2

Tabla 13 Diseño experimental propuesto.

Tabla 14 Análisis de resultados del experimento.

ÍNDICE DE FIGURAS

Figura 1 Actualidad de la bibliografía consultada (Fuente: elaboración propia).

Figura 2 Interacción de los componentes principales del proyecto “Extracción, publicación y consumo de metadatos bibliográficos como datos enlazados”.

Figura 3 Flujo de un crawler secuencial básico.

Figura 4 Esquema de funcionamiento de MetHarto (Hidalgo et al. 2013).

Figura 5 Arquitectura de la propuesta de solución (Fuente: Elaboración propia).

Figura 6: Funcionalidad encargada de cargar el archivo de configuración (Fuente: Elaboración propia).

Figura 7 Grafo de flujo asociado al método loadConfigFile() (Fuente: Elaboración propia).

Figura 8: Gráfica comparativa de los resultados obtenidos al realizar el experimento. (Fuente: Elaboración propia)

Figura 9: Gráfica comparativa de escalabilidad de los resultados obtenidos al realizar el experimento. (Fuente: Elaboración propia)

INTRODUCCIÓN

La World Wide Web (WWW por sus siglas en inglés) o red informática mundial es un sistema de distribución de documentos de hipertexto o hipermedios interconectados y accesibles vía Internet. Con un navegador web, un usuario visualiza sitios web compuestos de páginas web que pueden contener texto, imágenes, vídeos u otros contenidos multimedia, y navega a través de esas páginas usando hiperenlaces.

La Web, desde su creación hasta la actualidad, ha tenido un proceso de evolución y desarrollo con el objetivo de aumentar la calidad con la que se publican los contenidos y ampliar sus funcionalidades y aplicaciones. Anteriormente el usuario era un mero lector y su capacidad de interactuar con los contenidos ofrecidos por la red era prácticamente nula. En estos momentos se ha convertido en un lector-escritor de contenidos como lo plantean (RUIZ y MONTERROSO 2009) en su definición de la web 2.0, por lo que su papel ha ganado en influencia en las nuevas creaciones que se vierten a la red.

El crecimiento que ha experimentado la web en estos últimos años es innegable, ya que el usuario no solo interactúa con la red como receptor de conocimiento sino también como generador del mismo. Ejemplo de esto son los científicos y académicos, que publican los resultados de sus investigaciones en revistas científicas provocando que toda la información generada en la web sea imposible de consultar manualmente.

La brecha entre lo que el usuario demanda en una búsqueda por internet y los recursos recuperados se debe, principalmente, a que el sistema necesita trabajar con representaciones tanto de los documentos como de las necesidades de información de los usuarios para poder operar con ellos en el ámbito de uno de los diferentes modelos teóricos que abordan el problema de la recuperación de información (básicamente, el modelo booleano, el vectorial y el probabilístico). Por un lado, el contenido semántico de los documentos se suele representar mediante índices de frecuencias de aparición de términos, mientras que las necesidades de información suelen ser expresadas por los propios usuarios utilizando una serie de términos de búsqueda en muchos casos, ambiguos o inapropiados. Este problema adquiere una mayor dimensión cuando el usuario presenta un alto grado de especialización y requiere recursos muy específicos, como es el caso de los investigadores y su necesidad de acceder a la información más actual que se ha publicado en su área de interés.

En esta transformación de la forma de acceso a los recursos informativos no se han visto ajenas las bibliotecas, que al igual que la propia web, deben hacer frente al reto de gestionar eficazmente el

gran volumen de documentos que almacenan para facilitar a sus usuarios un acceso sencillo y ágil a recursos que satisfagan sus necesidades de información. En las bibliotecas, tradicionalmente se han propuesto diferentes soluciones como los servicios de difusión selectiva de información (DSI) y los boletines de novedades o los boletines de sumarios, pero en la actualidad se encuentran en la necesidad de aportar otras soluciones para adaptar sus servicios a las nuevas plataformas digitales de trabajo.

El progresivo uso de las tecnologías de la información y la comunicación (TIC) en las bibliotecas, la universalización del uso de Internet y la diversificación de los recursos que se pueden hacer accesibles desde la red, han provocado que las bibliotecas inicien un proceso de reinvención que implica una profunda revisión de sus técnicas y sus metodologías de trabajo y de los servicios que prestan para, de esta forma, poder satisfacer mejor las demandas cada vez más exigentes y específicas de sus usuarios. Como fruto de este proceso de transformación y adaptación, surgen las bibliotecas digitales, una extensión lógica de las bibliotecas físicas que albergan colecciones de recursos en formato electrónico (bien en su origen, bien tras ser sometidos a procesos de digitalización), y que tiene asociada una serie de servicios para facilitar el acceso a estos recursos a diferentes comunidades de usuarios utilizando para ello diversas tecnologías.

Las bibliotecas digitales son sistemas híbridos que heredan muchas de las virtudes y los inconvenientes de las bibliotecas físicas, ya que se ha producido un trasvase a la web de gran parte de sus herramientas y sus servicios, aunque adaptados a la nueva plataforma. Las diferencias se manifiestan en el sentido de que las bibliotecas físicas se basaban en procedimientos mecánicos y estas nuevas ya utilizan procesos automatizados y, lo más importante, se puede tener acceso a todas las fuentes documentales a distancia a través de internet. Otra diferencia importante consiste en que a las bibliotecas digitales se puede ingresar a cualquier hora los 365 días del año. También cumple con las normas ISO y con las leyes referentes a derecho de autor, de patentes y marcas.

Las bibliotecas digitales constituyen hoy en día uno de los principales nodos de acceso a la información en la web, y este papel se ve reforzado en el nuevo marco de la enseñanza superior como elemento dinamizador de la investigación y la docencia. No obstante, como grandes repositorios de información en continuo crecimiento, se hizo necesario dotar a las bibliotecas digitales de mecanismos que sean capaces de ofrecer la información que requieren los usuarios de la manera más eficiente posible y satisfacer así cada vez más las necesidades de información.

Entre los mecanismos aplicados a las bibliotecas digitales están las tecnologías de la **web semántica**[1]. Con la incorporación de estas metodologías se intenta proporcionar a los usuarios el acceso a bibliotecas digitales que integren diversas fuentes de datos, proporcionando servicios de valor añadido. Las bibliotecas digitales albergan actas de congresos, que constituyen recopilaciones de las ponencias y comunicaciones de congresos, simposios, seminarios, memorias de eventos científicos, entre otros. Usualmente dichas actas son almacenadas en formato Portable Document Format (PDF, por sus siglas en inglés) que aparece como una alternativa para la creación de documentos. Estas bibliotecas pueden proporcionar acceso al texto completo de las contribuciones, en adición a los metadatos de cada una de ellas (Delgado 2015). La calidad de los metadatos bibliográficos es un elemento crucial que afecta significativamente la visibilidad, el descubrimiento y la reutilización de los recursos descritos en el contexto de los datos enlazados.

En la Universidad de las Ciencias Informáticas el Grupo de Investigación de la Web Semántica se encuentra desarrollando el proyecto de investigación bajo el nombre **"Extracción, publicación y consumo de metadatos bibliográficos como datos enlazados"**, el cual está dividido en cuatro etapas, siendo la primera de ellas la extracción.

La etapa de extracción consiste en recopilar los artículos científicos en formato PDF desde revistas científicas en acceso abierto, así como los metadatos bibliográficos desde los mismos repositorios. Los metadatos obtenidos en este proceso son el título, los autores, el resumen y la fuente del artículo, pertenecientes a la portada de los documentos científicos.

Para la etapa de extracción fueron desarrolladas dos herramientas de software: MetHarto y PDFCrawler. La primera de ellas es un recolector de metadatos bibliográficos utilizando el protocolo OAI-PMH. La segunda, extrae y almacena en un repositorio artículos científicos en formato PDF existentes en cualquier revista científica en Acceso Abierto. No obstante, ambas herramientas aún no han sido integradas efectivamente. Esta situación constituye un problema de integración de datos y metadatos, al no contar con un proceso estandarizado para este propósito, por lo que actualmente no es posible establecer un vínculo entre los metadatos obtenidos por MetHarto y los artículos en formato PDF extraídos por PDFCrawler.

Atendiendo a la problemática anterior, se propone el siguiente **problema a resolver**: ¿Cómo integrar los datos y metadatos obtenidos desde revistas científicas en acceso abierto?

1 (BERNERS-LEE, HENDLER y LASSILA 2001) promotor del concepto de Web Semántica plantea "*La Web Semántica no pretende sustituir la Web actual, sino que es una extensión de la misma en la que la información tiene un significado bien definido, posibilitando a los humanos y las computadoras trabajar en cooperación*".

A partir de lo anterior se define como **Objeto de estudio** de la investigación: Proceso de desarrollo de software.

Dentro del objeto de estudio se encuentra el **Campo de acción** de la misma: Proceso de desarrollo de software orientado al consumo de datos.

Para resolver el problema se identifica el siguiente **objetivo general**: Desarrollar un componente de software que integre los datos y metadatos obtenidos a partir de revistas científicas en acceso abierto.

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

1. Elaborar el marco teórico y el estado del arte del objeto de estudio de la investigación mediante el análisis bibliográfico documental para identificar tendencias y adoptar posiciones al respecto.

2. Diseñar un componente de software que integre datos y metadatos obtenidos a partir de revistas científicas en acceso abierto.

3. Implementar un componente de software que integre datos y metadatos obtenidos a partir de revistas científicas en acceso abierto.

4. Validar los resultados obtenidos con la utilización del componente de software desarrollado mediante la realización de un diseño experimental.

Para el logro de los objetivos de la investigación, se aplicaron varios métodos de investigación científica.

A continuación, se detallan cada uno de ellos.

Con la aplicación el método **analítico-sintético** se conocieron las principales aproximaciones existentes en la literatura sobre los procesos de desarrollo de software para la administración de información, integración de datos y metadatos bibliográficos, así como los principales conceptos que guardan relación con el objeto de estudio de la investigación y las relaciones existentes entre ellos.

Para la confección la investigación se utilizó el método **Inductivo-Deductivo**, el cual permitió generalizar los procesos comunes que describen las diferentes guías metodológicas estudiadas. Este método se basa en los procedimientos inducción y deducción, donde el primero permite arribar a conclusiones generales a partir del estudio de las aproximaciones existentes sobre el objeto de la

investigación y el segundo permite a partir de un razonamiento lógico sobre lo anterior inferir nuevos conocimientos que lleven al planteamiento de una posible solución para la investigación.

Métodos Empíricos

Entre los métodos empíricos existentes se seleccionó para su utilización el de **medición**, el cual fue utilizado en la recopilación de datos para la validación de la solución propuesta.

El presente trabajo de diploma consta de tres capítulos, a continuación, un resumen de cada uno de ellos:

En el **Capítulo 1**: se realiza un análisis de la literatura consultada, se definen los principales conceptos asociados al área del conocimiento en cuestión. Como elemento modular del capítulo se hace una exposición sobre algunas de las herramientas que se utilizan para recolectar metadatos bibliográficos desde documentos en formato PDF, así como de algoritmos que se utilicen para comprar cadenas.

El **Capítulo 2**: se presenta un componente de software que permite integrar datos y metadatos bibliográficos. Para garantizar lo anterior se realiza una descripción general de la propuesta de solución. Además, se define el entorno de desarrollo a utilizar para la implementación y se hace una propuesta de arquitectura para el componente, entre otros aspectos.

El **Capítulo 3**: tiene como elemento central la validación de la solución propuesta a partir de las pruebas de software. Se explica el diseño experimental utilizado para demostrar si se resuelve o no el problema descrito a partir de un conjunto de datos. La propuesta de solución es validada a partir del diseño experimental, el cual determina si se integran los documentos en formatos PDF y los metadatos bibliográficos.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En este capítulo se enuncian los conceptos fundamentales relacionados con la integración de datos y metadatos bibliográficos sobre bibliotecas digitales. Se realiza un análisis de la literatura para identificar los elementos que pueden formar parte de la propuesta de solución. Finalmente, se identifican conceptos relacionados con la web semántica y que son de vital importancia en el ciclo de vida de un proyecto de datos enlazados.

1.2. Análisis bibliométrico y documental

En la presente investigación el análisis bibliométrico documental se realiza con el objetivo de mostrar la novedad de la revisión bibliográfica realizada, enmarcado principalmente en la literatura científica publicada en los últimos 5 años. Se consultan varias fuentes bibliográficas, entre las que se encuentran bases de datos referenciadas como Springer[2], IEEE[3], ACM[4]. A continuación, se muestra en la Tabla 1 un resumen de la bibliografía consultada.

Tabla 1 Resumen de la revisión bibliográfica consultada.

Tipo de fuente consultada	Cantidad consultada	Cantidad publicada en los últimos cinco años [2013-2017]
Artículos en revistas científicas	12	4
Artículos en congresos/conferencias	4	3
Libros	6	2
Páginas web	8	5

[2] http://www.springerlink.com/
[3] http://ieeexplore.ieee.org/Xplore/dynhome.jsp
[4] http://dl.acm.org/

Otros documentos	5	4
TOTAL	**36**	**18**

La tabla anterior muestra que se consultaron un total de 36 trabajos científicos, de los cuales 18 fueron publicados en los últimos cinco años, lo cual representa un 50% de la bibliografía consultada (Ver Figura 1).

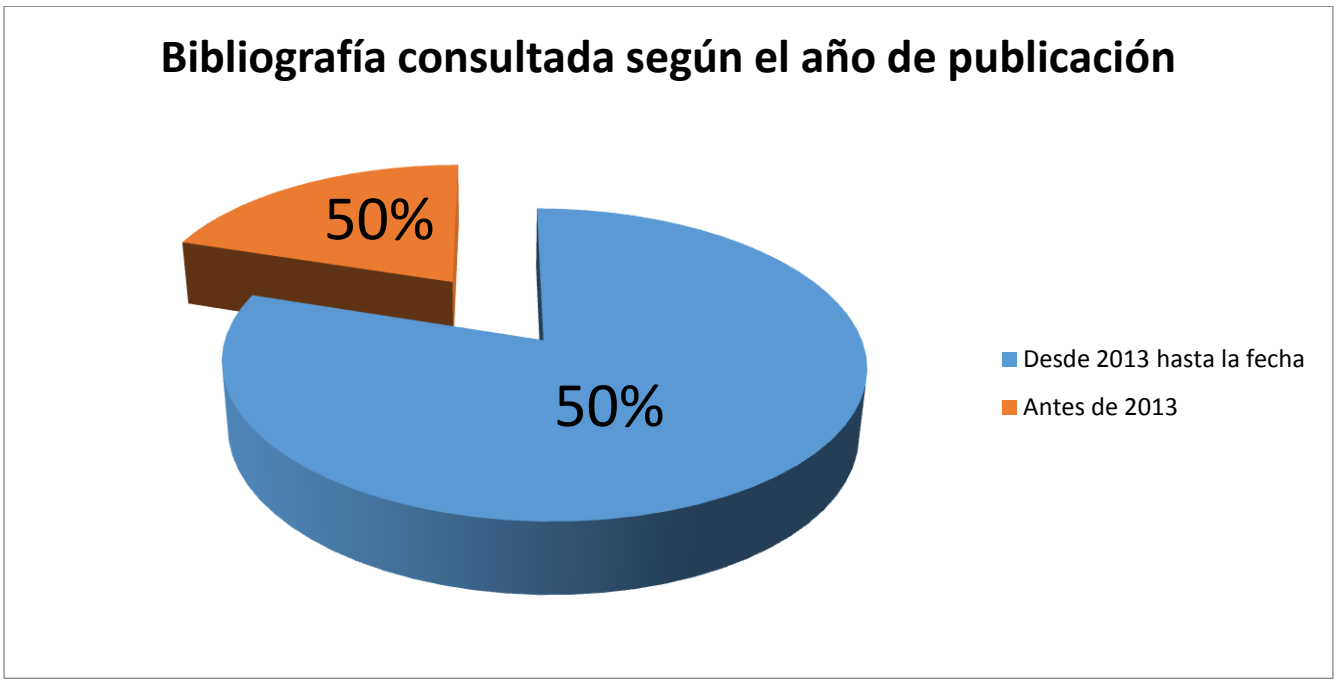

Figura 1 Actualidad de la bibliografía consultada (Fuente: elaboración propia).

1.3. Marco teórico

En este epígrafe se definen los conceptos fundamentales asociados al dominio del problema haciendo énfasis en sus características y aspectos más relevantes, posibilitando de esta forma crear una base teórica de los conocimientos necesarios para el desarrollo de la investigación.

Metadato

El concepto de metadato existe antes de la aparición de Internet, pero en los últimos años se ha popularizado mucho dada la necesidad de organizar la información en la web y de estandarizarla con

miras a la interoperabilidad de los sistemas de información. Aunque el término metadato se relacionó inicialmente con el campo de la bibliotecología, actualmente se ha extendido a los recursos digitales.

La primera acepción que se le dio (y actualmente la más extendida) fue la de dato sobre el dato, sin embargo, según (Tkaczyk et al. 2014), los metadatos son datos altamente estructurados que describen información, el contenido, la calidad, la condición y otras características de los datos. En otras palabras, es "Información sobre información".

Por otra parte, el término metadatos según (Initiative 2014), describe varios atributos de los objetos de información y les otorga significado, contexto y organización. En el mundo digital los metadatos permiten sustentar la navegación y la gestión de archivos.

Al analizar ambas definiciones, se puede apreciar que los dos autores coinciden en que los metadatos describen información sobre los datos, pero es (Tkaczyk et al. 2014) quien hace una disertación más amplia sobre el término metadatos, es por ello que se adopta esta definición, al ser la que más se ajusta a las características de la investigación.

Algunos ejemplos de metadatos son:

- Fecha y hora de creación del archivo.
- La dirección o la ubicación geográfica de dónde fue creado el archivo.
- El nombre de la persona, el nombre de la organización, el nombre de la computadora o dirección IP.
- Los nombres de cualquier persona que haya contribuido con el documento o los comentarios que insertaron.
- El tipo de cámara utilizada y sus configuraciones al momento de tomar la foto.
- Tipo de dispositivo de audio o video usado y las configuraciones establecidas al realizar una grabación.
- Marca, modelo y proveedor de servicios del teléfono inteligente.

PDF

Es un formato de almacenamiento para documentos digitales independiente de plataformas de software o hardware. Este formato es de tipo compuesto (imagen vectorial, mapa de bit y texto). Fue inicialmente desarrollado por la empresa Adobe System, oficialmente lanzado como un estándar abierto el 1 de julio de 2008 y publicado por la Organización Internacional de Estandarización (ISO) como ISO 32000-1.

El objetivo del formato PDF según (Adobe 2007) es permitir a los usuarios intercambiar y ver documentos electrónicos de forma fácil y confiable, independientemente del ambiente en el que fueron creados. En el núcleo de PDF se encuentra un modelo de imagen avanzada derivado de la descripción de la página del idioma PostScript®. Este modelo de imagen de PDF permite la descripción de texto y gráficos en un dispositivo independiente de la resolución. Para mejorar el rendimiento de la visualización interactiva, PDF define una estructura que es utilizada por la mayoría de los programas de lenguaje PostScript.

Formato de fichero PDF

Independientemente de cómo se haya creado los ficheros PDF, todos ellos comparten la misma estructura interna compuesta de cuatro partes:

- **Cabecera:** Información sobre la especificación del estándar PDF que se ha seguido en donde se indica, por ejemplo, la versión.
- **Cuerpo:** Descripción de los elementos usados en las páginas del fichero.
- **Tabla de referencias cruzadas:** Información de los elementos usados en las páginas del fichero.
- **Coda:** Indica donde encontrar la tabla de referencias cruzadas.

Cuando un fichero PDF es modificado y se añade nuevo contenido, éste tendrá nuevas secciones de cuerpo, tabla de referencias cruzadas y coda, pero al guardar este documento se puede optimizar para que las secciones duplicadas se fusionen en sólo una y se reorganice el fichero.

Tipos de archivos PDF

Según (BURHOE 2013) los tipos de archivos PDF son:

Archivos de sólo imágenes

Un archivo de sólo imágenes se presenta como una imagen de mapa de bits o una instantánea. Debido a que es una instantánea, cualquier texto no se puede buscar. Sin embargo, este formato es útil cuando las versiones digitales deben ser absolutamente fieles a las originales, como en los casos de facturas o documentos legales.

Documentos sin etiquetas

Los documentos se han creado sin etiquetas PDF. Las etiquetas PDF son similares a las etiquetas utilizadas en el código HTML para hacer la búsqueda web más accesible. El texto en documentos sin

etiqueta es a menudo difícil de leer. Cuando sucede esto, aparecerá un cuadro de diálogo, dando al usuario la opción de añadir etiquetas.

Documentos etiquetados

Los documentos etiquetados son fáciles de encontrar a través de una búsqueda web. Son fáciles de leer, optimizados para ser vistos en pantallas pequeñas y capaces de ser copiados.

Formularios electrónicos

Los formularios PDF electrónicos ofrecen al usuario la posibilidad de guardar los datos introducidos con el teclado o copiados de un archivo existente. Los formularios terminados pueden ser protegidos con contraseña y guardados. Además pueden ser publicados en Internet o enviados a través de correo electrónico.

Biblioteca digital

El concepto de biblioteca digital es empleado en la actualidad con diversas acepciones, siempre relacionadas con el acceso a documentos en formato electrónico por medio de redes de comunicación. En este sentido, bibliotecas digitales, virtuales o electrónicas son términos que se utilizan indiferentemente para referirse a realidades similares: sistemas de acceso a documentos electrónicos desde portales específicos.

Varias son las definiciones que se han aplicado a las bibliotecas digitales. Algunas defienden que las bibliotecas digitales son meramente bibliotecas electrónicas. La biblioteca electrónica se aquella que permite acceder a bancos de información en formato electrónico. Este tipo de bibliotecas incluye también los catálogos automatizados de bibliotecas tradicionales. Según esta definición la biblioteca electrónica intenta reproducir la producción impresa pero utilizando un medio diferente del soporte papel. Partiendo de esta realidad la biblioteca digital sigue los pasos de la biblioteca electrónica, pero evolucionando hacia la introducción de otros tipos de materiales, es decir, introduciendo elementos digitales.

Una definición de biblioteca digital dada por (DLP 2012), plantea que "biblioteca digital no es únicamente el equivalente de repositorios digitalizados con métodos de gestión de la información. Es más bien, un entorno donde se reúnen colecciones, servicios, y personal que favorece el ciclo completo de la creación, difusión, uso y preservación de los datos, para la información y el conocimiento".

Por otra parte (Pérez 2014) afirma que la biblioteca digital no intenta "copiar" la realidad impresa, sino que genera una nueva estructura de la información que hace que esta evolucione desde el concepto lineal del libro y los documentos tradicionales al concepto hipertextual, donde la información llega al usuario de formas muy variadas y provista de todo tipo de vínculos, los cuales permiten ampliar, concretar o explicar los contenidos de forma simultánea y diferente. El hipertexto incluye mucha más información no textual que el impreso, ya que incorpora elementos multidimensionales: voz, sonido, imagen, 3D, etc.

A partir de las definiciones descritas anteriormente se puede realizar las siguientes clasificaciones (Pérez 2014):

- Biblioteca clásica: contenidos en soportes físicos, acceso mediante referencias bibliográficas consignadas en los catálogos.
- Biblioteca electrónica: contenidos en soporte electrónico, acceso por medios físicos (CD-ROM), o electrónicos (acceso en línea).
- Biblioteca digital: contenidos en soportes electrónicos y digitales, y acceso en línea a través de redes telemáticas.
- Biblioteca virtual: contenidos en soporte electrónico y digital, y acceso en línea a través de redes telemáticas (como en las bibliotecas digitales).

Al realizar un análisis de las definiciones anteriores, los distintos autores coinciden en que las bibliotecas digitales son repositorios de objetos digitales, parcialmente organizados, que sirven a una comunidad de usuarios definida, los cuales tienen los derechos de autor presentes y gestionados, y disponen de mecanismos de preservación y conservación. Esta definición tiene en cuenta que estos repositorios constan de datos (el contenido) y metadatos (la información que describe los datos) e incorporan técnicas de búsqueda y recuperación de la información.

Integración de metadatos bibliográficos y PDF

Un escenario de integración hace posible la combinación de recursos de información existentes en diversas fuentes. Esto proporciona al usuario una vista unificada de dichos recursos y también puede actuar como una fuente de datos para diversas aplicaciones. Por esta razón al autor de la presente investigación define a la integración de metadatos bibliográficos y PDF como el proceso mediante el cual se hace coincidir los metadatos extraídos mediante el protocolo OAI-PMH con su respectivo documento en formato PDF.

1.4. Estado del arte

Al tenerse los metadatos bibliográficos y los documentos PDF en directorios diferentes se hace necesario el estudio de algunas herramientas que ayuden al proceso de integración. Es por ello que en el estado del arte de la investigación se realiza un análisis sobre un grupo de herramientas para la extracción automática de metadatos bibliográficos desde documentos PDF, y se analizan algoritmos de comparación de cadenas, los cuales son elementos que formarán parte de la propuesta de solución.

1.4.1 Herramientas para la extracción de metadatos bibliográficos

Existen varias herramientas dedicadas a la extracción de metadatos bibliográficos a partir de documentos científicos y técnicos en formato PDF, las cuales se exponen en la Tabla 2

Tabla 2 Herramientas para la extracción de metadatos bibliográficos desde documentos PDF.

Nombre de la herramienta	Método que utiliza
Docear's PDF Inspector*	Análisis del Estilo de la Información (SIA) [5]
GROBID	CRF[6]
Medeley Desktop	SVM, basado en web
ParsCit	CRF
PDFMeat*	Consultas de Google Scholar, pdftotext
SciPlore Xtract*	Análisis de XML

De las herramientas mencionadas en la tabla anterior se seleccionaron aquellas que utilizan técnicas de Aprendizaje Automático (AA). Estas son GROBID, Mendeley y ParsCit. Las herramientas señaladas con un asterisco no serán analizadas como parte del estado del arte de la investigación ya que no utilizan técnicas de AA. A continuación se analizan las herramientas seleccionadas.

GROBID [7]

[5] SIA, Style Information Analysis
[6] CRF, Conditional Random Fields
[7] GROBID,GeneReration Of Bibliographic Data

GROBID es un sistema para la extracción y generación automática de metadatos bibliográficos de documentos científicos y técnicos y el reconocimiento de la estructura del documento (López y Romary 2015). Esta herramienta posee una licencia de software libre, y fue desarrollado utilizando el lenguaje de programación Java. Puede ser empleada como una aplicación web o integrada a otros sistemas.

Los metadatos bibliográficos que puede extraer son: los autores, el título, el resumen, las palabras claves, el contenido del artículo, la información sobre la revista en la que fue publicado el artículo y las referencias bibliográficas. Los tipos de documentos utilizados para la extracción de los metadatos en esta herramienta son documentos científicos y técnicos, documentos académicos, manuales técnicos y patentes, siempre en formato PDF. El procesado de los documentos se realiza en cinco pasos, los tres primeros son aplicados al conjunto de documentos utilizados para el entrenamiento:

1. Análisis de la estructura del documento.
2. Selección de los términos candidatos.
3. Análisis de características.
4. Aplicación del modelo de AA para la evaluación de cada término candidato independientemente.
5. Analizar nuevamente para obtener relaciones entre los términos candidatos que no fueron capturadas por el modelo de AA.

Uno de sus objetivos es la conversión de los documentos científicos en formato PDF a documentos en formato TEI[8]. Para ello es necesario primeramente reconocer la estructura del documento PDF y luego extraer los metadatos (López y Romary 2015). En el análisis de las secciones del documento se utiliza el método de aprendizaje automático CRF. GROBID centra su atención en el procesado de las secciones: encabezado (título, resumen), introducción, la sección de títulos, las conclusiones y las referencias bibliográficas, porque en estas secciones los autores introducen los elementos principales y los lectores suelen prestar más atención también a estas partes del documento.

Puede ser utilizado en las bibliotecas digitales como un módulo para el análisis y procesado de documentos de texto. También en las bibliotecas digitales para la obtención de información a partir del procesado de los documentos, para generar y sugerir citas bibliográficas a los usuarios (López 2015)

[8] TEI, Text Encoding Initiative

Mendeley Desktop

Mendeley posee una licencia de software libre. Puede utilizarse a través de un sitio web o una aplicación para PC y dispositivos Apple (IPhone e IPad) para el almacenamiento y manejo de documentos PDF. Permite tener los documentos almacenados en la nube y también compartirlos con otras personas como una red social. La aplicación automáticamente organiza los artículos por categorías (autor, título, revista, fecha, entre otros) en una base de datos para luego realizar filtrados. Proporciona el manejo de referencias bibliográficas, la selección o creación de estilos de citas textuales y la creación automática de bibliografía (Lo Russo et al. 2013). Permite agregar artículos a la base de datos desde diferentes fuentes, bases de datos online, desde la propia computadora o de otras bibliotecas digitales (Lo Russo et al. 2013).

ParsCit

Es una herramienta de código abierto para el análisis de referencias bibliográficas. ParsCit realiza el análisis examinando cada una de las referencias e identificando cada campo que las componen. Los campos extraídos pueden ser utilizados por otros autores. Consta de dos procesos fundamentales para la extracción de las referencias, el pre-procesado y el post-procesado (Councill et al. 2015).

En el pre-procesado, ParsCit utiliza métodos heurísticos para convertir el documento en formato PDF a texto plano, empleando UTF-8 (Councill et al. 2015). Luego, en el post-procesado utiliza CRF++, implementación del método de aprendizaje automático CRF, para obtener cada uno de los tokens que componen la referencia (Granitzer et al. 2015). La herramienta puede ser utilizada tanto como un servicio web o como una aplicación independiente.

1.4.1.1. **Comparación entre las herramientas**

Con el propósito de conocer que aplicación tiene un mejor desempeño en el proceso de extracción de metadatos bibliográficos se toma como referencia la comparación hecha por (Lipinski, et al. 2013). Para llevar a cabo la comparación (Lipinski, et al. 2013) selecciona aleatoriamente una colección de 1153 artículos científicos en PDF, incluyendo sus metadatos, para compararlos con los extraídos por las herramientas estudiadas. Los metadatos seleccionados para el análisis fueron el título, el autor o autores, separando el nombre y los apellidos, el resumen y el año de publicación. Estos metadatos suelen ser los más utilizados en la realización de consultas.

Las herramientas deben cumplir el requisito de permitir la integración con otros proyectos de desarrollo (por ejemplo, una biblioteca digital), a través de una biblioteca de clases o ser una

aplicación independiente que permita cargar archivos PDF. A partir de aquí se realizan tres evaluaciones con dos configuraciones de pruebas según el número de artículos que se procesan, cien en la primera y 1153 en la segunda. Los resultados obtenidos para las herramientas seleccionadas se muestran en la tabla siguiente:

Tabla 3: Resultados (A100: Primera evaluación con 100 artículos, B100: Segunda evaluación con 100 artículos, B1153: Segunda evaluación con 1153 artículos), (Lipinski, et al. 2013)

	Título			**Autores**			**Apellidos del autor(es)**		**Resumen**			**Año**	
	A_{100}	B_{100}	B_{1153}	A_{100}	B_{100}	B_{1153}	B_{100}	B_{1153}	A_{100}	B_{100}	B_{1153}	B_{100}	B_{1153}
GROBID	N/A	0.92	0.92	N/A	0.83	0.83	0.90	0.91	N/A	0.75	0.74	0.64	0.69
Mendeley Desktop	N/A	0.84	0.82	N/A	0.72	0.70	0.78	0.77	N/A	N/A	N/A	0.23	0.26
ParsCit	0.59	0.52	0.54	0.47	0.29	0.31	0.36	0.37	0.49	0.31	0.26	0.06	0.07

Los valores representados en la tabla corresponden a la evaluación del desempeño que tuvo cada una de las herramientas en la extracción de los metadatos seleccionados. El valor uno indica que el metadato extraído coincide con los datos referenciados, cero que el metadato fue extraído incorrectamente. De las aplicaciones analizadas GROBID tuvo el mejor desempeño; 0.92 para títulos, 0.83 para los autores, 0.90 para el apellido de los autores, 0.74 para el resumen y 0.69 para el año de publicación.

El desempeño de GROBID indica que los metadatos extraídos tuvieron un mayor nivel de coincidencia con los metadatos que se tomaron como referencia para la comparación. Tiene ventajas sobre las otras herramientas, ya que al trabajar directamente con grandes cantidades de documentos es poca la información que se pierde.

En (Granitrzer et al. 2015) se comparan las herramientas Mendeley y ParsCit, obteniendo Mendeley una mejor evaluación. Además: indican que el método SVM[9] es mejor que el método CRF, pero con los resultados obtenidos en la comparación se puede concluir que la implementación de CRF que utiliza GROBID es mejor que el SVM de Mendeley y GROBID tiene un mejor desempeño en la extracción de metadatos que Mendeley.

[9] SVM, Support Vector Machine

1.4.2 Algoritmos para la similitud de cadenas

La similitud de cadenas consiste en que si tienes dos cadenas A y B, identificar qué tan parecidas o diferentes son. De acuerdo a (Christen 2012), una función de comparación aproximada dada debe cumplir con las siguientes propiedades:

a) El resultado de comparar una entidad consigo misma deberá arrojar un valor de similitud de 1.
b) El resultado de comparar dos entidades "completamente diferentes" deberá arrojar un valor de similitud de 0.
c) El resultado de comparar dos entidades "aproximadamente similares" entre sí deberá arrojar un valor de similitud entre 0 y 1.

En general existen diferentes algoritmos para medir la similitud de cadenas, entre los más conocidos están Jaro, Jaro-Winkler, Ratcliff/Obershelp y Levenstein.

Ratcliff/Obershelp en SuperBASIC[10]

El algoritmo Ratcliff/Obershelp consiste en una estrategia para calcular la similitud entre dos cadenas de caracteres. Para este cálculo se emplean el número de caracteres coincidentes dividido por el número total de caracteres en las dos cadenas. La búsqueda de las coincidencias se basa en ir hallando la mayor sub-secuencia de caracteres en común entre las dos cadenas de forma recursiva.

El algoritmo está diseñado para comparar dos cadenas y devolver un porcentaje que represente la similitud o lo cerca que están la una de la otra. Un resultado de 67 significa que las dos cadenas tienen un porcentaje de igualdad del 67%.

Se podría usar para comparar una palabra mal escrita con una serie de posibles palabras correctas en el corrector ortográfico de un procesador de textos. La palabra con el mayor porcentaje de igualdad se tomaría como la ortografía correcta de la palabra. También se podría emplear esta rutina en algún juego conversacional en el que se quisiera ayudar al usuario en caso de que cometa errores ortográficos o tipográficos.

El algoritmo es un buen ejemplo de cómo utilizar la recursividad en SuperBASIC. Toma las dos palabras y extrae la mayor subcadena común coincidente entre ellas (llamada esta subcadena MSC). Para cada cadena, se toma la parte de la palabra a la izquierda y a la derecha de la MSC y con esta

[10] SuperBASIC: lenguaje de alto nivel utilizado en los microordenadores de la familia Sinclair QL a mediados de los años 1990.

sub-cadenas se vuelve a repetir el proceso comparando unas y otras. El proceso finaliza cuando la cadena completa es analizada.

Distancia de Jaro

La Distancia de Jaro es una métrica para la comparación aproximada de cadenas inventada por Matthew Jaro (Jaro 1989). La distancia de Jaro se calcula al contar el número de caracteres coincidentes que son comunes para ambas cadenas en una longitud igual a la mitad de la longitud de la cadena más larga. La métrica de Jaro también considera el número de transposiciones de caracteres presentes en las cadenas a comparar (Cohen et al. 2003).

Una transposición se define de la siguiente manera. Considere a $\mathbf{s_1}$ como una cadena formada por los caracteres comunes a ambas cadenas en el orden que aparecen en $\mathbf{s_1}$. De la misma manera, considere a $\mathbf{s_2}$ como una cadena formada por los caracteres comunes a ambas cadenas en el orden que aparecen en $\mathbf{s_2}$. De esa manera, existe una transposición cuando para las cadenas $\mathbf{s_1}$ y $\mathbf{s_2}$, el carácter observado en el índice **i** es diferente.

Si no hay caracteres coincidentes dentro de la longitud permitida, la Distancia de Jaro es igual a cero. En caso contrario, la Distancia de Jaro está dada por la siguiente fórmula (Christen 2012):

$$sim_{jaro}(S_1, S_2) = \text{⅓} + (c/|S_1| + c/|S_2| + c\text{-}t/|c|)$$

Dónde:

|s1| es la longitud de la cadena $_1$

|s2| es la longitud de la cadena $_2$

c es el número de caracteres coincidentes en dentro de la longitud permitida (aquella menor a la longitud de la cadena más larga dividida entre dos)

t es el número de transposiciones en las cadenas dividido entre dos.

La siguiente tabla muestra valores de Distancia de Jaro para diferentes cadenas, junto con los valores de sus parámetros (Christen 2012):

Tabla 4 Valores de Distancia de Jaro para diferentes cadenas correspondientes a nombres y apellidos

Candena 1	**Cadena 2**	**C**	**T**	**Distancia de Jaro**
Shackleford	Shackelford	11	0	0,9697
Nichleson	Nichulson	8	0	0,9259
Jones	Johnson	4	0	0,7905
Massey	Massie	5	0	0,8889
Jeraldine	Geraldine	8	0	0,9259
Michelle	Michael	6	0	0,8690

Distancia de Jaro-Winkler

La Distancia de Jaro-Winkler fue publicada por William E. Winkler, en 1990 (Winkler 1990). Esta métrica se basa en la Distancia de Jaro y le suma otros parámetros para mejorar la comparación de las cadenas.

La Distancia de Winkler se calcula de acuerdo a la siguiente fórmula (Christen 2012):

$sim_{winkler}(S_1, S_2) = sim_{jaro}(S_1, S_2)+(1.0-som_{jaro}(S_1, S_2))\ P/10$

Dónde:

sim_{jaro} es la Distancia de Jaro

p es el número de caracteres coincidentes al inicio de la cadena. El valor permitido de p es de uno a cuatro.

La siguiente tabla muestra valores de Distancia de Winkler para diferentes cadenas, junto con los valores de sus parámetros (Christen 2012):

Tabla 5 Valores de Distancia de Jaro-Winkler para diferentes cadenas

Candena 1	**Cadena 2**	**C**	**T**	**p**	**Distancia de Jaro**
Shackleford	Shackelford	11	1	4	0,9818
Nichleson	Nichulson	8	0	4	0,9556
Jones	Johnson	4	0	2	0,8324
Massey	Massie	5	0	4	0,9333
Jeraldine	Geraldine	8	0	0	0,9259
Michelle	Michael	6	0	4	0,9214

Comparación por Distancia de Levenshtein

La Distancia de Levenstein se define como qué tan distantes son dos cadenas entre sí de acuerdo al número de operaciones de edición que se tendrían que realizar sobre una cadena para que fuese exactamente igual a la otra (Navarro 2001).

En la Distancia de Levenshtein las operaciones permitidas de edición son la inserción (se añade un carácter extra), la modificación (se cambia un carácter por otro) y el borrado (se elimina un carácter de la cadena). A estas operaciones se les asigna un costo para el cálculo de la Distancia de Levenshtein; lo más usual es que a todas se les asigne un costo de uno.

Tabla 6 Distancias de Levenshtein para diferentes cadenas correspondientes a nombres y apellidos

Candena 1	**Cadena 2**	**Distancia Levenshtein**
Shackleford	Shackelford	2
Nichleson	Nichulson	2
Jones	Johnson	4
Massey	Massie	5

Jeraldine	Geraldine	1
Michelle	Michael	3

Luego del análisis de estos algoritmos para la comparación de cadenas se puede concluir que **Ratcliff/Obershelp** es bueno para encontrar la similitud de cadenas en el lenguaje SuperBASIC, **Levenstein** aunque es uno de los algoritmos más conocidos el resultado al comparar las cadenas no es normalizado, el número que arroja depende del largo de las cadenas, mientras **Jaro** y **Jaro-Winkler** son normalizados, sin importar el largo, si las cadenas son completamente iguales, el resultado es 1. Es por ello que se selecciona Jaro-Winkler para utilizarlo en esta investigación.

1.5. Conclusiones parciales

La revisión de la bibliografía evidenció cuales son las técnicas existentes para la recolección de metadatos desde documentos PDF, así como de los principales algoritmos para calcular la similitud de cadenas. Con el análisis de los principales conceptos del marco teórico y sus relaciones, se logró profundizar en el problema planteado por la investigación.

CAPÍTULO 2. DESCRIPCIÓN DE LA PROPUESTA

2.1. Introducción

En el presente capítulo se describe la solución que se propone como resultado de este trabajo. Para ello primeramente se hace un análisis sobre su diseño y la arquitectura. Se especifican los principales productos de trabajo generados, así como la propuesta arquitectónica del componente implementado que integra los metadatos bibliográficos con los documentos en formato PDF en la biblioteca digital. Además, se realiza un estudio sobre las herramientas y tecnologías empleadas en el proceso de implementación de la solución.

2.2. Enfoque propuesto

El proyecto “Extracción, publicación y consumo de metadatos bibliográficos como datos enlazados” tiene como objetivo construir una Biblioteca Digital Semántica basada en datos enlazados. Este es el primer proyecto de su tipo en Cuba por lo que no se cuenta con colecciones de artículos en formato PDF. El proyecto cuenta con cuatro actividades fundamentales, que según (Delgado 2015) son definidas de la siguiente manera: (1) Extracción de datos, (2) Pre-procesamiento de datos, (3) Publicación de datos y finalmente (4) Consumo de datos. Ver Figura 2.

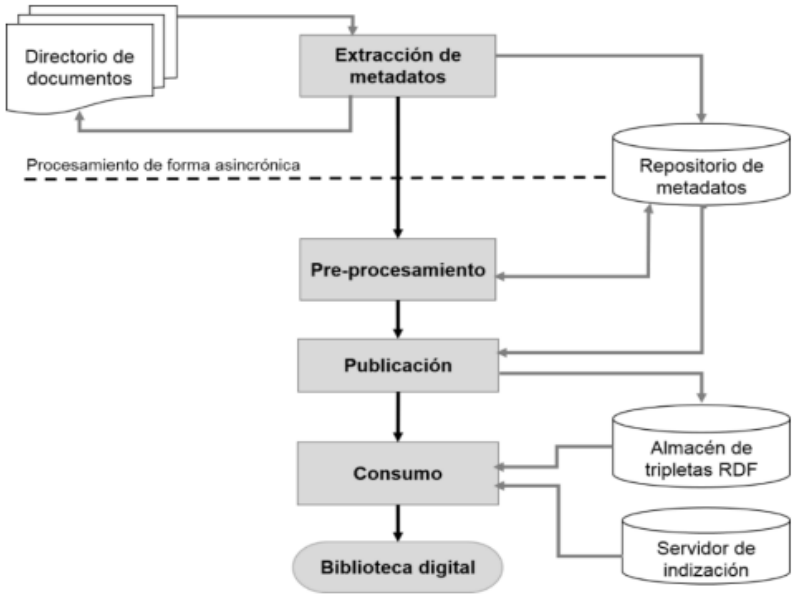

Figura 2 Interacción de los componentes principales del proyecto “Extracción, publicación y consumo de metadatos bibliográficos como datos enlazados”.

Lo primero es extraer y almacenar los metadatos provenientes de fuentes de datos heterogéneas. En esta etapa, es necesario analizar varios aspectos de cada fuente de datos, tales como: la interoperabilidad de datos, la calidad de los datos, los esquemas de datos, la frecuencia de actualización y la sostenibilidad en el tiempo. El resultado para esa actividad es una base de datos relacional intermedia con los metadatos extraídos.

En un segundo momento son analizados los metadatos provenientes de la tarea anterior, puesto que la calidad de los datos de la biblioteca es un punto crucial que afecta significativamente la visibilidad y el descubrimiento de los recursos descritos. El propósito que se tiene es limpiar y normalizar algunos campos de metadatos mejorando su calidad. Esta actividad incluye la transformación de datos, tales como fechas, volúmenes y números de revistas; así mismo, la detección de registros duplicados y la desambiguación de autores y afiliaciones. Consecuentemente se obtiene la base de datos con los metadatos limpiados y normalizados.

Como tercera instancia es importante determinar la(s) ontología(s) a utilizar para el modelado de datos de la biblioteca. Se recomienda en este contexto reutilizar lo más posible los vocabularios disponibles. Como salida a este proceso se obtiene un modelo ontológico. Con el modelo ontológico y el esquema de base de datos, lo próximo es la realización de tres importantes tareas: (1) la transformación, (2) el enlazado y (3) la publicación de dichos datos. Como consecuencia de acometer estas tareas se generan grafos RDF que son almacenados en un almacén de tripletas RDF. Finalmente, una vez los datos listos para su consumo, es posible obtener mejores resultados mediante búsquedas textuales y facetadas, así como de manera más rápida mediante el empleo de índices para las consultas realizadas por los usuarios.

La solución que se propone como resultado de este trabajo es el desarrollo del componente **mrlink** con el propósito de realizar el siguiente proceso, incluido en la etapa: extracción de metadatos, el cual se indica a continuación:

- **Sincronizar los PDF y los metadatos bibliográficos extraídos**
  Esta actividad es el núcleo del componente, en ella se integran los PDF almacenados en un repositorio con los metadatos bibliográficos que están en una base de datos SQLite.

2.3. Herramientas utilizadas

A continuación se describen las herramientas que actualmente se utilizan en el proyecto “Extracción, publicación y consumo de metadatos bibliográficos como datos enlazados” para la extracción de PDF y la extracción de metadatos bibliográficos desde revistas científicas en acceso abierto.

SDL-Crawler

PDFCrawler según (León y Delgado 2014) es un crawler focalizado basado en **crawler4j** el cual proporciona una Application Program Interface (API por sus siglas en Ingles) y una biblioteca que facilitan el trabajo con el mecanismo implementado. Es capaz de recolectar documentos en formato PDF a partir de una URL inicial o lista de URL almacenándolos en una dirección destino que se le configura.

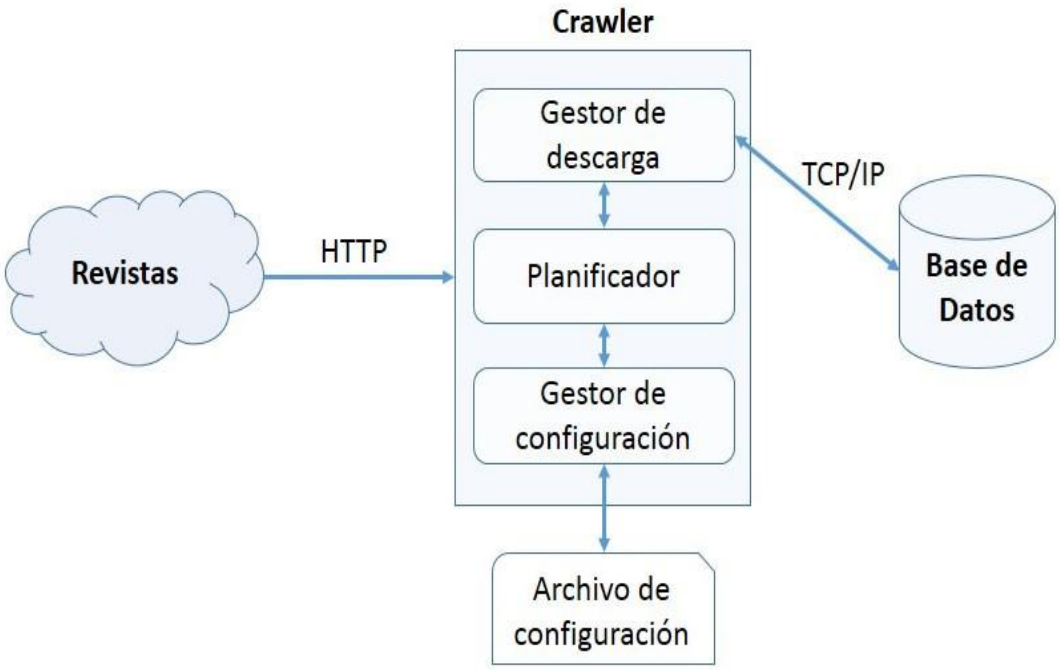

Figura 3 Flujo de un crawler secuencial básico.

En la Figura 3 se puede observar cómo es el funcionamiento del PDFCrawler. A partir de una dirección URL o un conjunto de direcciones URLs, el crawler procede a descargar los documentos en formato PDF guardándolos en un repositorio y aprovecha los enlaces de dichas páginas para realizar una nueva búsqueda con cada URL vinculada. A continuación, se explica la función de los tres componentes principales:

Gestor de descarga: examina el contenido de un sitio web, toma los documentos PDF y los almacena en una base de datos. A la vez busca en dicho sitios más enlaces o URL, los cuales son puestos en una cola de espera para después ser visitados y analizados.

Planificador: es el componente que se encarga de tomar los enlaces de la cola de espera para enviarlos al gestor de descarga y así realizar con él un nuevo proceso. Trabaja directamente con el Gestor de configuración.

Gestor de Configuración: es donde se guardan las configuraciones que se realizan en el archivo de configuración del crawler. El planificador realiza su función a partir de las configuraciones que estén en este componente.

MetHarto

MetHarto según (Hidalgo et al. 2013) es una herramienta que se utiliza para recolectar metadatos en lotes y de forma selectiva desde múltiples repositorios. Los metadatos recolectados son almacenados en una base de datos relacional MySQL o Posgre para su uso posterior (ver Figura 4). Posee una interfaz de línea de comando para realizar algunas tareas comunes.

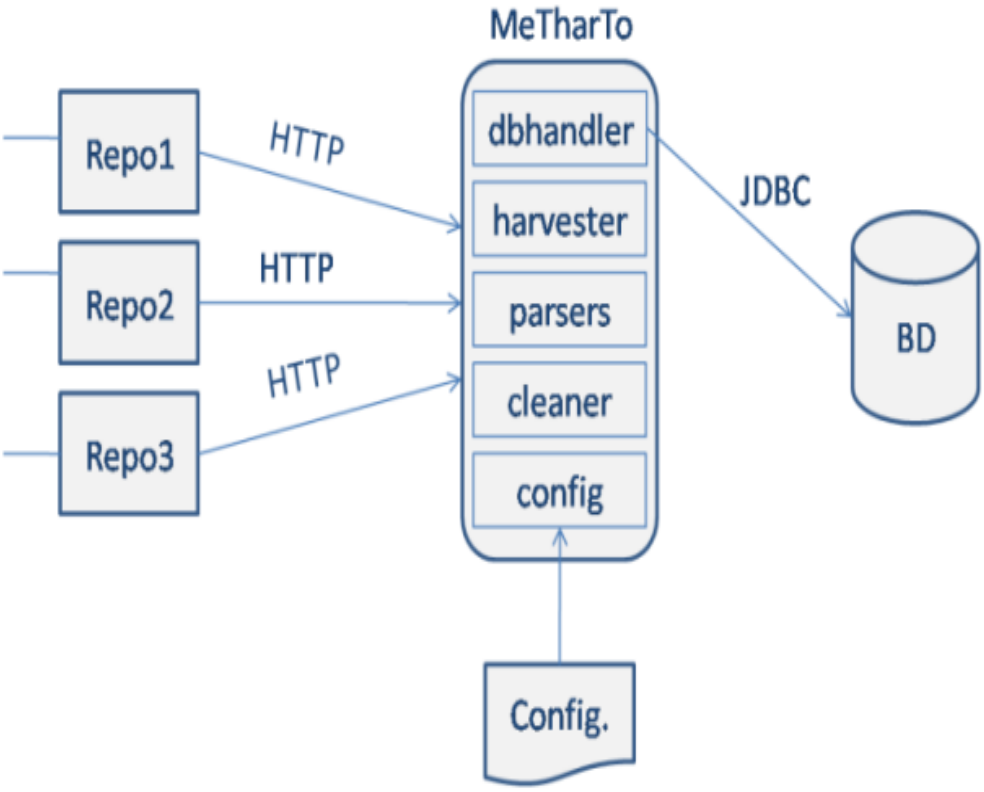

Figura 4 Esquema de funcionamiento de MetHarto (Hidalgo et al. 2013).

La herramienta MetHarto fue diseñada por paquetes. Cada paquete está compuesto por un conjunto de clases. Los principales según (Hidalgo et al. 2013) son:

config: contiene las clases necesarias para la gestión de las opciones de configuración de la herramienta, permitiendo que la misma sea configurable a diversos escenarios de ejecución. Entre las opciones de configuración principales se encuentran el nombre del servidor proxy, el puerto, el usuario y la contraseña.

dbhandler: permite gestionar, mediante la utilización de Hibernate, las conexiones a la base de datos relacional en donde se persisten los metadatos recolectados.

harvester: contiene el controlador principal de la herramienta desde el que se realizan las tareas principales. También posee la clase que gestiona la interfaz en línea de comandos de la aplicación.

parsers: contiene las clases necesarias para procesar los archivos XML utilizando Axiom[11].

cleaner: contiene un conjunto de clases encargadas de la limpieza de los metadatos obtenidos por la herramienta. Algunos de estos metadatos son: título, autores y fuente. Se diseñó un algoritmo con el objetivo de normalizar los nombres de los autores, haciendo transformaciones tales como: eliminar tildes y caracteres especiales, eliminar espacios en blanco al principio y al final, eliminar grado científico o categoría docente, entre otras.

2.4. Entorno de desarrollo

A lo largo de la investigación se utilizan un grupo de herramientas y tecnologías. A continuación, se realiza una breve descripción de cada una de ellas.

Sistema Gestor de Base de Datos

Un Sistema Gestor de Bases de Datos (SGBD) es una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Posee un lenguaje para la manipulación y consulta de los datos. Permite definir los datos a distintos niveles de abstracción.

Sqlite

SQLite según (Rommel 2007) es una herramienta de software libre, que permite almacenar información en dispositivos de una forma sencilla, eficaz, potente, rápida y en equipos con pocas capacidades de hardware, como puede ser una PDA[12] o un teléfono celular. SQLite implementa el estándar SQL92 y también agrega extensiones que facilitan su uso en cualquier ambiente de

[11] http://ws.apache.org/axiom/
[12] PDA, Personal Digital Assistant

desarrollo. Esto permite que SQLite soporte desde las consultas más básicas hasta las más complejas del lenguaje SQL, y lo más importante es que se puede usar tanto en dispositivos móviles como en sistemas de escritorio, sin necesidad de realizar procesos complejos de importación y exportación de datos, ya que existe compatibilidad al 100% entre las diversas plataformas disponibles, haciendo que la portabilidad entre dispositivos y plataformas sea transparente.

Algunas de las características principales de SQLite son:

- La base de datos completa se encuentra en un solo archivo.
- Puede funcionar enteramente en memoria, lo que la hace muy rápida.
- Tiene un footprint menor a 230KB.
- Es totalmente autocontenida (sin dependencias externas).
- Cuenta con librerías de acceso para muchos lenguajes de programación.
- Soporta texto en formato UTF-8 y UTF-16, así como datos numéricos de 64 bits.
- Soporta funciones SQL definidas por el usuario (UDF).
- El código fuente es de dominio público y se encuentra muy bien documentado.

Lenguaje de Programación Java

Se utiliza Java como lenguaje de programación. Este lenguaje es orientado a objeto siendo rápido, seguro y fiable. Desde portátiles hasta centros de datos, desde consolas para juegos hasta súper computadoras, desde teléfonos móviles hasta Internet, Java está en todas partes. La programación en Java tiene muchas similitudes con el lenguaje C y C++, así que si se tiene conocimiento de este lenguaje, el aprendizaje de la programación Java será de fácil comprensión por un programador que haya realizado programas en estos lenguajes.

Entorno de desarrollo Integrado

Un Entorno de Desarrollo Integrado, por sus siglas en inglés (IDE) es un programa compuesto por una serie de herramientas utilizadas por programadores para desarrollar aplicaciones.

Eclipse

Eclipse es un software cuyo principal objetivo es el desarrollo de otro software. Dispone de un editor de textos con un analizador sintáctico. La compilación es en tiempo real. Tiene pruebas unitarias con JUnit, control de versiones con CVS, integración con Ant, asistentes (*wizards*) para creación de

proyectos, clases, tests, etc., y refactorización. Asimismo, a través de "plugins" libremente disponibles es posible añadir control de versiones con Subversion e integración con Hibernate. En este trabajo se utiliza la versión 3.8, la cual es definida por el grupo de investigación de Web Semántica.

2.5. Requisitos

La especificación de requisitos del software es una descripción completa del comportamiento del sistema software a desarrollar. Incluye la descripción de todas las interacciones que se prevén que los usuarios tendrán con el software. También contiene requisitos no funcionales, que imponen restricciones de diseño o funcionamiento del sistema software (PYTEL et al. 2011).

2.5.1. Técnicas para la captura de requisitos

La obtención de requisitos es el proceso mediante el cual los interesados en un sistema de software descubren, revelan, articulan y entienden sus requisitos. Esta actividad debe ser muy efectiva ya que de su éxito dependerá que se satisfagan las necesidades del cliente (CHAVE 2007).

La técnica de captura de requisitos utilizada en la presente investigación es:

Entrevista

Esta técnica se aplicó con el objetivo de tener un mejor entendimiento del problema y de las necesidades del cliente. A partir de esta se definen los requisitos funcionales que debe cumplir la propuesta de solución. La entrevista se realizó a partir de la selección de un conjunto de preguntas que responden a lo anterior y fueron realizadas al Ing. Alejandro Jesús Mariño Melerio, 2do jefe del grupo de investigación de Web Semántica y que a su vez es el cliente.

La entrevista realizada brindó información que es fundamental para el desarrollo de la propuesta de solución. A partir de ella se definieron los requisitos funcionales de la solución, los cuales constituyen la base para lograr una correcta implementación y así cumplir con las necesidades y expectativas del cliente.

2.5.2. Requisitos funcionales

Los requisitos funcionales de un sistema son aquellos que especifican lo que este debe hacer, son declaraciones de los servicios que proveerá el sistema, de la manera en que éste reaccionará a entradas particulares. La tabla siguiente muestra el requisito funcional de la propuesta de solución:

Tabla 7: Requisitos funcionales.

No.	Requisito	Prioridad	Complejidad
1	Sincronizar PDF y metadatos.	Alta	Alta

2.5.3. Requisitos no funcionales

Los requerimientos no funcionales son condiciones que imponen restricciones en el diseño o la implementación. Son propiedades o cualidades que el producto debe tener. Los buenos requisitos deben ser medibles, comprobables y sin ambigüedades. Estos requerimientos son de gran significación en la aceptación del software, debido a que representan las ventajas más visibles al usuario y repercuten en el óptimo funcionamiento y mantenimiento del sistema (MORENO et al. 2013).

2.5.3.1. Software

Los requisitos no funcionales de software según (Giraldo 2007) es el software con que se debe disponer, una vez implementado el sistema. A continuación, son especificados los dos requisitos no funcionales de software con el que debe de cumplir el componente que se propone. Ver Tabla 8.

Tabla 8 Relación de los requisitos no funcionales de software.

No.	Requisitos no funcionales de software
1	Máquina virtual de Java v.1.7 o superior.
2	Gestor de Base de Datos SQLite

2.5.3.2. Hardware

Los requisitos no funcionales de hardware según (Giraldo 2007) son los elementos de hardware que se necesitan para que el software cumpla sus funcionalidades. A continuación, son especificados los cuatros requisitos no funcionales de hardware con el que debe de cumplir el componente que se propone. Ver Tabla 9.

Tabla 9 Relación de los requisitos no funcionales de hardware.

No.	Requisitos no funcionales de hardware
1	Microprocesador Intel core i5-2380P CPU 3.10Ghz.
2	1 GB de memoria RAM.

3	512 GB de disco duro.
4	Tarjeta de red

2.6. Arquitectura del componente

La arquitectura de software según (Reynoso y Kicillof 2004) se entenderá como “la organización fundamental de un sistema encarnada en sus componentes, las relaciones de los componentes con cada uno de los otros y con el entorno, y los principios que orientan su diseño y evolución”. Debe quedar claro que en esta definición el concepto de “componente” es genérico e informal, y no se refiere sólo a lo que técnicamente se concibe como tal en modelos de componente.

La propuesta de solución sigue una arquitectura centrada en flujo de datos: tuberías y filtros. Este estilo arquitectónico provee la arquitectura y los mecanismos para los sistemas que deben procesar flujos de datos. Cada etapa del procesamiento es encapsulada en un filtro. Los datos se transmiten a través de tubos entre filtros adyacentes. Se pueden obtener familias de sistemas relacionados recombinando, eliminando y agregando filtros (Reynoso y Kicillof 2004).

Se sugiere aplicar este estilo en los siguientes casos (Reynoso y Kicillof 2004):

- Procesamiento de señales.
- Procesamiento de imágenes o sonido.
- Compiladores.
- Procesamiento de cadenas.
- Sistemas con poca o nula interacción con el usuario cuyo flujo de datos se entienda o perciba como continuo.
- También se menciona la posibilidad de aplicarlo para el cálculo de la Transformada de Fourier Discreta (Rápida), algoritmos de búsqueda de paralelo de simulación científica.

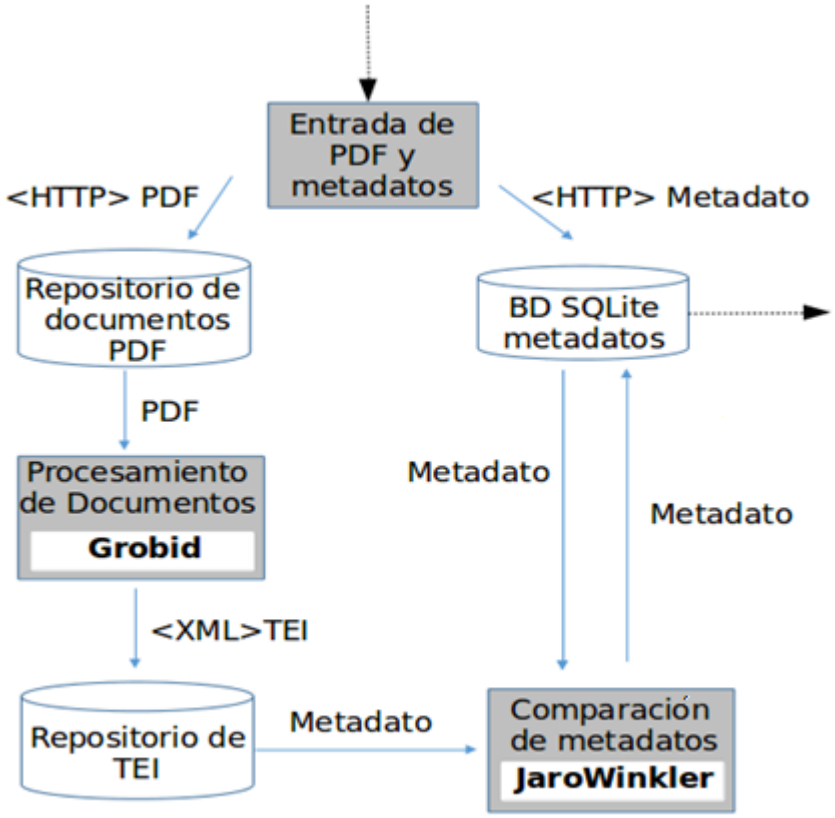

Figura 5 Arquitectura de la propuesta de solución (Fuente: Elaboración propia).

En la Figura 5 se muestra el diseño arquitectónico que sigue la propuesta de solución. El proceso comienza cuando el usuario especifica los datos que le son solicitados por el sistema, díganse los metadatos bibliográficos y los documentos científicos en formato PDF que serán procesados. Una vez que está especificada la colección de documentos son procesados por la herramienta Grobid, la cual recibe los documentos en formato PDF, los procesa y extrae sus metadatos bibliográficos convirtiéndolos en formato TEI. Luego de tener los metadatos devueltos por la herramienta Grobid se almacenan en el repositorio de TEI y comienza la actividad comparación. Se comparan los metadatos de los archivos TEI con los que están en la base de datos de metadatos utilizando Jaro-Winkler que es un algoritmo para comparar cadenas. Si la comparación es mayor a 0.9 que es el umbral definido, entonces coinciden y se actualiza la base de datos de metadatos especificando de qué PDF son los metadatos.

Estándares de código

Un estándar de código se basa en la estructura y apariencia física de un programa con el fin de facilitar la lectura, comprensión, mantenimiento del código, reutilización a lo largo del proceso de

desarrollo de un software y no en la lógica del programa. Un estándar de programación no solo busca definir la nomenclatura de las variables, objetos, métodos y funciones, sino que también tiene que ver con el orden y legibilidad del código escrito (Guerrouj 2013). Partiendo de lo dicho anteriormente, se definen 3 partes principales dentro de un estándar de programación:

2.1.1. Nomenclatura de las clases

Los nombres de las clases siempre comienzan con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación UpperCamelCase, la cual define que la primera letra de cada una de las palabras es mayúscula y con solo leerlo se reconoce el propósito de la misma.

Ejemplo: SQLiteConnection. En este caso el nombre de la clase está compuesto por dos palabras iniciadas cada una con letra mayúscula.

2.1.2. Nomenclatura de las funcionalidades y atributos

El nombre a emplear para las funciones y los atributos se escriben con la inicial del identificador en minúscula, en caso de que sea un nombre compuesto se empleará notación lowerCamelCase.

Ejemplo de función: getRecords(). El nombre de este método está compuesto por dos palabras, debido a esto es que se escribe la primera con minúscula y la segunda con mayúscula.

Ejemplo de atributo: connection. El nombre del atributo está compuesto por una sola palabra, debido a esto es que se escribe con minúscula, si fuera un nombre compuesto por más de una palabra se procede a aplicar la notación antes mencionada.

2.1.3. Nomenclatura de los comentarios

Los comentarios deben ser lo bastante claros y precisos de forma tal que se entienda el propósito de lo que se está desarrollando. En caso de ser una función complicada se debe comentar para lograr una mejor comprensión del código.

2.7. Planificación de pruebas

El proceso de pruebas permite aumentar la calidad del sistema reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. En el caso específico del componente implementado, solo se utilizarán las pruebas internas y las pruebas de aceptación, debido a que las pruebas de liberación son 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 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 (Sánchez 2014). Es aplicable a componentes representados en el modelo de implementación para verificar que los flujos de control y de datos están cubiertos, y que ellos funcionen como se espera. El objetivo de las pruebas internas es el aislamiento de partes del código y la demostración de que estas partes no contienen errores.

La prueba de unidad siempre está orientada a caja blanca, que serán las utilizadas por el equipo de desarrollo. Para ello se comprobarán los caminos lógicos del software proponiendo casos de prueba que examinen que están correctas todas las condiciones y/o bucles para determinar si el estado real coincide con el esperado o afirmado. Esto genera gran cantidad de caminos posibles por lo que hay que dedicar esfuerzos a la determinación de las condiciones de prueba que se van a verificar. Este tipo de prueba será aplicada a la estructura procedimental (código fuente) de las funcionalidades que implementa cada historia de usuario, a través de la técnica del camino básico.

Pruebas de aceptación

Estas pruebas las realiza el cliente. En este caso el cliente es el Grupo de Investigación de Web Semántica. Las pruebas de aceptación, no se realizan durante el desarrollo, pues no se podrían presentar al cliente; sino que se ejecutan sobre el producto terminado e integrado o bien una versión del producto o una iteración funcional pactada previamente con el cliente. 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.

La experiencia muestra que aún después del más cuidadoso proceso de pruebas por parte del desarrollador, quedan una serie de errores que aparecen cuando el cliente comienza a usarlo, por lo que las pruebas de aceptación tienen un mayor grado de importancia que las pruebas unitarias (RUIZ et al. 2011)(PRESSMAN 2005).

2.8. Conclusiones parciales

La definición del requisito funcional permitió tener una mejor comprensión de las necesidades del cliente, así como determinar cuáles eran las principales actividades que eran necesarias implementar para dar cumplimiento al requisito.

El patrón arquitectónico seleccionado para representar la arquitectura de la propuesta de solución ilustra cómo se realiza la integración de los documentos PDF con los metadatos bibliográficos.

CAPÍTULO 3. VALIDACIÓN DE LA PROPUESTA

En el capítulo anterior fueron definidos los tipos y técnicas de pruebas para su empleo posterior. Este capítulo tiene como objetivo validar la propuesta de solución aplicando dichas pruebas. Se desarrolla un caso de estudio en un contexto real de utilización empleando artículos en formato PDF y metadatos bibliográficos extraídos de revistas científicas en acceso abierto. Adicionalmente, se realiza un pre-experimento para evaluar los tiempos de respuesta de la aplicación al realizar las diferentes tareas de integración de metadatos bibliográficos con sus respectivos documentos en formato PDF y de esta forma validar el componente propuesto. Además, se describen en detalle los principales resultados obtenidos con el caso de estudio y el pre-experimento desarrollado. Por último, se relacionan aspectos representativos resultantes de la validación de la propuesta de solución y se emiten las consideraciones generales de la sección.

3.1. Pruebas de software

El principal objetivo de las pruebas de software es obtener un conjunto de instrumentos que permitan descubrir los defectos del software. Para llevar a cabo este objetivo, se usan dos tipos de prueba: pruebas de caja blanca y pruebas de aceptación.

3.1.1 Pruebas de caja blanca

Las pruebas de caja blanca fueron aplicadas haciendo uso de la técnica del camino básico, como se menciona en el capítulo anterior, con el objetivo de evaluar la complejidad lógica de un diseño procedimental y usar esta medida como guía para la definición de un conjunto básico de caminos de ejecución (Rogers 2000). Esta prueba permite garantizar que en los casos de prueba obtenidos a través del camino básico se ejecute cada sentencia del programa por lo menos una vez.

Para este caso, se expone su aplicación sobre la funcionalidad encargada de cargar el archivo de configuración, el método loadConfigFile() el cual se encuentra contenido en la clase BmrlinkConfigManager, (ver Figura 6). Se identificaron los bloques de ejecución para este método atendiendo a las dependencias procedimentales, con el objetivo de evaluar la complejidad lógica y usar esta medida como guía para la definición de un conjunto básico de caminos de ejecución.

```
public Properties loadConfigFile() throws IOException {
        Properties loadedProperties = new Properties();                    //(1) -Inicio
        File file = new File(this.getPath());                              //(2)
        if (file.exists()) {                                               //(3)
            loadedProperties.load(new FileInputStream(file));              //(4)
        } else {
            GenerateDefaultConfigFile();                                   //(5)
            loadedProperties.load(new FileInputStream(file));              //(6)
        }
        return loadedProperties;                                           //(7) -Fin
    }
```

Figura 6: Funcionalidad encargada de cargar el archivo de configuración (Fuente: Elaboración propia).

Después de este paso, es necesario representar el grafo de flujo asociado al código antes presentado a través de nodos, aristas y regiones, ver Figura 7.

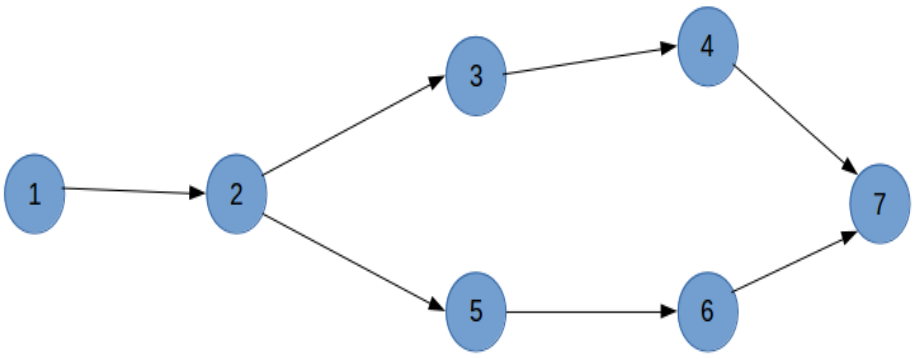

Figura 7 Grafo de flujo asociado al método loadConfigFile() (Fuente: Elaboración propia).

Una vez construido el grafo de flujo asociado al procedimiento se determina la complejidad ciclomática, la cual es una métrica de software útil pues proporciona una medición cuantitativa de la complejidad lógica de un programa. El valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y da un límite superior para el número de pruebas que se deben realizar.

Para calcular dicha complejidad existen tres vías de solución, las cuales se enuncian a continuación:

V(G) = (a − n) + 2 **(I)** **V(G)** = p + 1 **(II)** **V(G)** = r **(III)**

Siendo **a** la cantidad total de aristas, **n** la cantidad total de nodos, **p** la cantidad total de nodos predicados (nodos de los cuales parten dos o más aristas) y **r** la cantidad total de regiones (Rogers 2000). Se aplican las tres fórmulas para afirmar un resultado seguro y confiable.

V(G) = (7 − 7) + 2 **=** 2 **(I)** **V(G)** = 1 + 1 = 2 **(II)** **V(G)** = 2 **(III)**

La evaluación de las fórmulas I, II y III arroja un valor de complejidad ciclomática igual a 2. Esta cifra representa la cantidad de caminos independientes que existen en el grafo de flujo construido para el método loadConfigFile(). Este valor constituye el número mínimo de casos de prueba a crear para el procedimiento tratado. A continuación se muestra la Tabla 10 con cada uno de los caminos independientes que fueron identificados en el grafo de flujo.

Tabla 10 Caminos independientes identificados en el grafo de flujo del método loadConfigFile().

Caminos independientes	
Camino Básico: #1	1-2-3-4-7
Camino Básico: #2	1-2-5-6-7

Luego de tener elaborado el grafo de flujo e identificados los caminos a recorrer, se preparan los casos de prueba que garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa. Se escogen los datos de manera que las condiciones de los nodos predicados estén adecuadamente establecidas, con el fin de comprobar cada camino. A continuación, se especifican los casos de prueba.

Tabla 11 Caso de prueba para el camino básico #1

Código: CPCB-01	
Descripción:	Los datos de entrada cumplirán los siguientes requisitos: -Variable file debe de estar vacía.
Condición de ejecución:	No debe de existir el archivo de configuración.
Entrada:	File:"null”
Resultados esperados:	Se crea el archivo de configuración y se carga el archivo de configuración.
Evaluación de la prueba: Satisfactoria	

Tabla 12 Caso de prueba para el camino básico #2

Código: CPCB-02	
Descripción:	Los datos de entrada cumplirán los siguientes requisitos: -Variable file contiene un archivo de configuración.
Condición de ejecución:	Existe el archivo de configuración,
Entrada:	File:”config/mrlink.conf”
Resultados esperados:	Se carga el archivo de configuración.
Evaluación de la prueba: Satisfactoria	

3.1.2 Pruebas de aceptación con el cliente

Para realizar las pruebas de aceptación se utilizó la técnica de validación con el cliente. El cliente es quien valida si la aplicación está lista para ser utilizada por los usuarios finales. Para la validación del software se utilizaron los casos de pruebas definidos para las pruebas funcionales, además de un conjunto de datos de prueba que son las entradas a cada uno de los casos de prueba.

3.2. Caso de estudio

Con el objetivo de validar la solución al problema de investigación se diseña un caso de estudio. Se utiliza para ello una colección con 75 PDF los cuales están almacenados a priori en un directorio local, siendo provenientes de “la Revista de Enfermedades no Transmisibles”, “la Revista Cubana de Urología” y “la Revista Médico Científica”. Para el caso de estudio se cuenta con un equipo de cómputo con las siguientes prestaciones:

- Tipo de CPU: Intel Core i5 5200U (5ta generación
- Memoria del sistema: 4GB RAM DDR3 SDRAM

Se proponen los siguientes escenarios para la evaluación:

- Realizar la integración de metadatos bibliográficos y PDF sin el empleo de la propuesta de solución, es decir de manera manual.
- Realizar la integración de metadatos bibliográficos y PDF utilizando la herramienta CERMINE.
- Realizar la integración de metadatos bibliográficos y PDF utilizando la propuesta de solución como estímulo.

3.3. Diseño experimental

Para una mejor comprensión del contenido de este subepígrafe se hace necesario definir el término experimento. Una definición simple pero acertada es: un estudio que involucra la manipulación intencional de una acción para analizar sus posibles efectos (Grau et al. 2004). Los experimentos se dividen en tres grupos: (1) pre-experimentos, (2) cuasi-experimentos y (3) experimentos puros.

Los pre-experimentos se distinguen por no poseer un grupo de control o patrón para realizar la comparación. La principal característica de los cuasi-experimentos es que la asignación de los participantes a los grupos no se hace de forma aleatoria ni por emparejamiento. Los experimentos puros difieren de los pre-experimentos y los cuasi-experimentos en el control sobre la situación experimental; en ellos se debe manipular una o más variables independientes, medir el efecto de la

variable independiente sobre la variable dependiente y controlar la validez interna de la situación experimental.

De acuerdo a la clasificación anterior de los experimentos en la investigación se emplea un pre-experimento, dado que se precisa el resultado de una observación inicial que será comparada en otro momento con la aplicación de un estímulo. Se definen cuatro tareas a realizar, enumeradas seguidamente:

1. Integrar 15 documentos en formato PDF con sus metadatos bibliográficos.
2. Integrar 30 documentos en formato PDF con sus metadatos bibliográficos.
3. Integrar 50 documentos en formato PDF con sus metadatos bibliográficos.
4. Integrar 75 documentos en formato PDF con sus metadatos bibliográficos.

A continuación, se muestra la Tabla 13 con el diseño experimental propuesto.

Tabla 13 Diseño experimental propuesto.

Fuente de datos	Tareas	Observación simple	Estímulo 1	Observación con estímulo 1	Estímulo 2	Observación estímulo 2
G	T_1	OS_1	E_1	OE_{11}	E_2	OE_{21}
	T_2	OS_2		OE_{12}		OE_{22}
	T_3	OS_3		OE_{13}		OE_{23}
	T_4	OS_4		OE_{14}		OE_{24}

La simbología empleada en la tabla anterior es la siguiente:

- **G**: Colección de documentos en formato PDF como fuente de datos.
- **T i**: Tareas (procesar X cantidad de PDF) realizadas sobre G. El subíndice i representa el número de la consulta.
- **OS i**: Resultado de la observación luego de acometer T. El indicador es el tiempo en segundos que tarda el usuario en extraer los metadatos de manera manual.
- **E_1**: Tratamiento o estímulo. En este caso la aplicación del componente CIMINE.
- **OE_1 i**: Observación realizada tras aplicar E_1. El indicador es el tiempo en segundos que tarda la herramienta en integrar los metadatos y los PDF.
- **E_2**: Tratamiento o estímulo. En este caso la aplicación del componente propuesto.

- **OE_2 i**: Observación realizada tras aplicar E_2. El indicador es el tiempo en segundos que tarda la herramienta en integrar los metadatos y los PDF.

3.4. Análisis de los resultados

Tabla 14 Análisis de resultados del experimento.

Fuente de datos	Tareas	Observación simple	Estímulo 1	Observación con estímulo 1	Estímulo 2	Observación con estímulo 2
75 PDF	T_1: Procesar 15 PDF	OS_1: 00:12:15:00	CER	OE_{11}: 00:09:03:00	MRLINK	OE_{21}: 00:00:24:95
	T_2: Procesar 30 PDF	OS_2: 00:24:54:07		OE_{12}: 00:18:06:00		OE_{22}: 00:00:37:16
	T_3: Procesar 50 PDF	OS_3: 00:40:04:46		OE_{13}: 00:30:01:00		OE_{23}: 00:00:38:92
	T_4: Procesar 75 PDF	OS_4: 01:00:16:32		OE_{14}: 00:45:15:00		OE_{24}: 00:00:52:03

La Tabla 13 corresponde a la aplicación de la propuesta de solución como estímulo en el segundo y tercer escenario de prueba. Nótese que los tiempos de procesamiento de manera manual, en este caso la observación simple, son muy superiores a los tiempos obtenidos al aplicar el estímulo 1 y el estímulo 2 en este caso el componente para la integración de metadatos y PDF al mismo grupo de tareas. Con la aplicación de la observación con estimulo 2 se puede apreciar una reducción considerable de los tiempos de integración de los metadatos bibliográficos con sus documentos en formato PDF. En la Figura 8 se reflejan más claramente los resultados obtenidos con la realización del experimento.

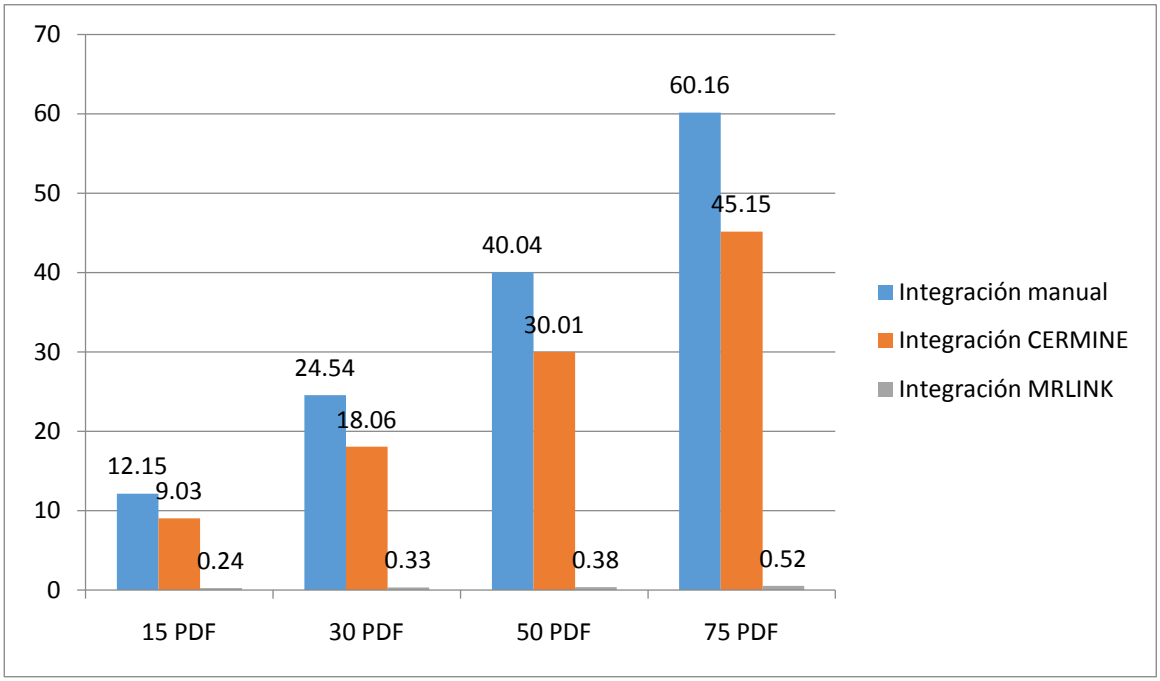

Figura 8: Gráfica comparativa de los resultados obtenidos al realizar el experimento. (Fuente: Elaboración propia)

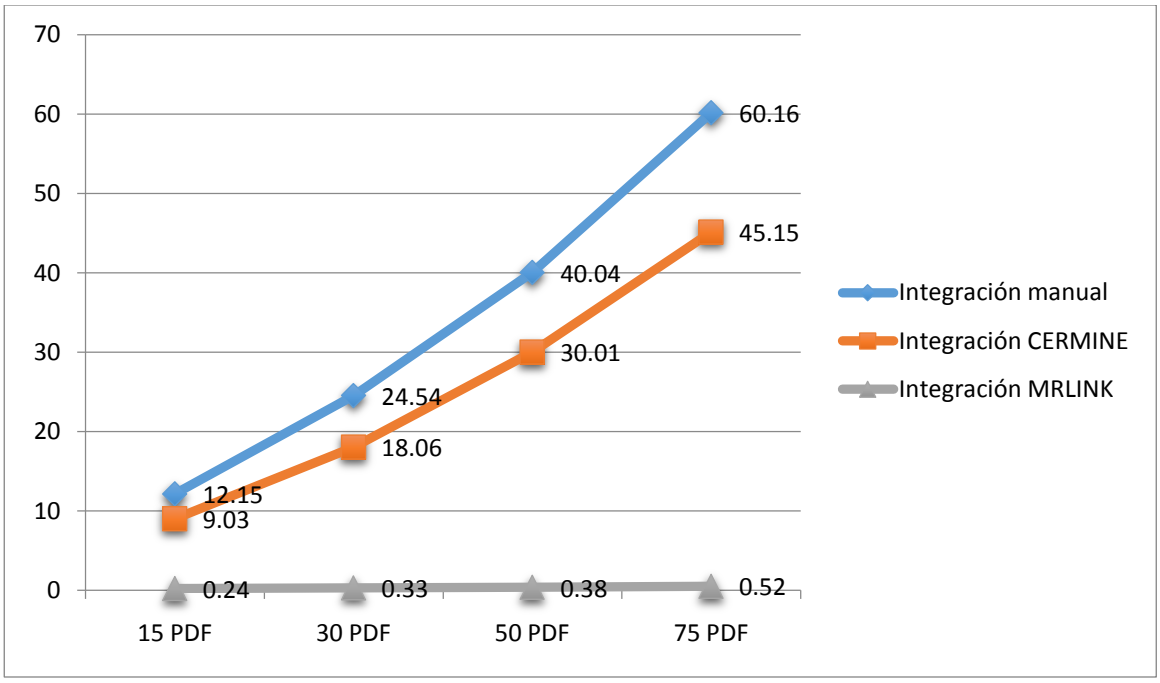

Figura 9: Gráfica comparativa de escalabilidad de los resultados obtenidos al realizar el experimento. (Fuente: Elaboración propia)

La gráfica 9 muestra los valores de escalabilidad de los 3 escenarios que son analizados en el pre-experimento. Nótese que la escalabilidad para el proceso manual es bien elevada, llegando a ser una función exponencial, mientras más grande es la cantidad de artículos en formato PDF a integrar mayor será el tiempo para su realización. Aún con el estímulo 1 el proceso sigue siendo alto con una escalabilidad muy parecida al primer escenario, aunque con menor tiempo. Con la aplicación del estímulo 2 se puede observar que los resultados son prácticamente una función lineal, que aunque aumente la cantidad de artículos en formato PDF a integrar, el tiempo aumentará pero en menor medida.

3.5. Conclusiones parciales

En este capítulo se diseñó un caso de estudio y se propuso un diseño experimental con el propósito de validar la propuesta de solución presentada en el capítulo anterior. Tras aplicar las pruebas de software definidas y el diseño experimental descrito, se concluye lo siguiente:

- Las pruebas de caja blanca permitieron comprobar que los casos de prueba obtenidos a través del camino básico ejecuten cada sentencia del componente por lo menos una vez.
- El diseño de un caso de estudio permitió realizar un pre-experimento con el objetivo de validar la propuesta de solución presentada en el capitulo anterior.
- El análisis del tiempo demostró que el componente desarrollado contribuye a la reducción del tiempo empleado en la realización de la actividad de integración de datos y metadatos bibliográficos.

CONCLUSIONES GENERALES

- La definición del marco teórico permitió una mayor comprensión del contexto donde se desarrolla la investigación, definiéndose las tecnologías adecuadas que se usaran para desarrollar el componente propuesto.
- La arquitectura propuesta permitió estructurar el componente, sirviendo de base para las actividades de implementación.
- Los resultados obtenidos en las pruebas de caja blanca demostraron que no existe código innecesario dentro del paquete implementado.
- La realización de un pre-experimento demostró que el componente desarrollado contribuye a la reducción del tiempo empleado en la realización de la actividad de integración de datos y metadatos bibliográficos.

RECOMENDACIONES

Se recomienda integrar el componente implementado en la Biblioteca Digital Semántica.

Actualmente el componente implementado solo está diseñado para procesar artículos científicos publicados en revistas y eventos. Se recomienda extender las funcionalidades del componente para integrar datos y metadatos de otros documentos científicos tales como Libros y Tesis.

REFERENCIAS BIBLIOGRÁFICAS

Adobe Systems Incorporated, 2007, “Document management -Portable document format -Part 1: PDF 1.7"., (PDF 32000-1:2008)» . p. 6. [en línea] [Consulta: 28 mayo 2017]

BERNERS-LEE, T., HENDLER, J. y LASSILA, O., 2001. The semantic web. Scientific american, vol. 284, no. 5, pp. 28–37.

BURHOE, B., 2013, Tipos de archivos PDF, eHow en Espanol, [en línea] [Consulta: 25 mayo 2017]

REYNOSO, C. y KICILLOF, N., 2004. Estilos y Patrones en la Estrategia de Arquitectura de Microsoft,Universidad de Buenos Aires, Version 1.0. [en línea] [Consulta: 30 mayo 2017]

CHAVE, M.A., 2007. La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. [en línea], vol. VI, no. 10. ISSN 1409-4746. Disponible en: http://revistas.ucr.ac.cr/index.php/intersedes/article/view/790.

Christen, P., 2012 . Data matching. Data-Centric Systems and Appl, Springer.pp. 101

Christen, P., 2012. Data matching. Data-Centric Systems and Appl, Springer.pp.110.

COHEN, W., RAVIKUMAR, P. y FIENBERG, S., 2003, A comparison of string metrics for marching names and records. In KDD Workshop on Dta Cleaning and Object Consolidation (Vol. 3, pp. 73-78).

COUNCILL, I.G., LEE GILES, C. y KAN, M.-Y., 2015. ParsCit: An open-source CRF reference string parsing package.

DELGADO, Y.H., 2015. Marco de trabajo basado en los datos enlazados para la interoperabilidad semántica en el protocolo OAI-PMH. [en línea], [Consulta: 29 marzo 2017]. Disponible en: http://eprints.rclis.org/28755/

DELGADO, Y.H., RODRIGUEZ, R., ORTIZ, E., ALONSO, L.E., 2013. HERRAMIENTA PARA LA RECOLECCIÓN DE METADATOS BIBLIOGRÁFICOS MEDIANTE EL PROTOCOLO OAI-PMH, Informatica 2013.

DIGITAL LIBRARIES INITIATIVE, 2012 Biblioteca Digital. [en línea], [Consulta: 01 abril 2017]. http://www.dli2.nsf.gov

GARCIA, Ariel Rodriguez, 2013, El aprovechamiento de los metadatos en las bibliotecas. e-Ciencias de la Información. 2013. Vol. 0, no. 0, p. 1-13.

GIRALDO, O.P., 2007. Ingenieria de requisitos. Volumen, 13.

GRANITZER, M., MAYA, H. y ROBERT, K., 2015. A Comparison of Metadata Extraction Techniques for Crowdsourced Bibliographic Metadata Management.

GRAU, R., CORREA, C. and ROJAS, M., 2004, Metodología de la Investigación (Segunda Edición ed.). Ibagué: El POIRA Editores SA. 2004

GUERROUJ, Latifa, 2013, Normalizing source code vocabulary to support program comprehension and software quality. In : Proceedings of the 2013 International Conference on Software Engineering. IEEE Press. 2013. p. 1385–1388.

INITIATIVE, Dublin Core Metadata and OTHERS, 2014, Dublin core metadata initiative. . 2014.

JARO, M.A., 1989, Advances in record linkage methodology as applied to the 1985 census of Tampa Florida, Journal of the American Statistical Association. 84 (406): 414-20.

LIPINSKI, M., KEVIN, Y., JOERAN, B. y BELA, G., 2013. Evaluation of Header Metadata Extraction Approaches and Tools for Scientific PDF Documents. , pp. 385-386.

Leon, J.G.H., Delgado, Y.H., 2015. Crawler focalizado para la extracción de documentos PDF desde revistas científicas, Serie Científica de la Universidad de las Ciencias Informáticas; Vol 8, No 1 (Year 2015).

LÓPEZ, P. y ROMARY, L., 2015. HUMB: Automatic Key Term Extraction from Scientific Articles in GROBID. ,

LÓPEZ, P., 2015. GROBID: Combining Automatic Bibliographic Data Recognition and Term Extraction For Scholarship Publications. ,

LO RUSSO, G., SPOLVERI, F., CIANCIO, F. y MORI, A., 2013. Mendeley: An Easy Way to Manage, Share, and Synchronize Papers and Citations. Plastic and Reconstructive Surgery, pp. 946-947. DOI 10.1097/PRS.0b013e31828bd400

MORENO, Juan Carlos y MARCISZACK, Marcelo Martín, 2013, La Usabilidad Desde La Perspectiva De La Validación de Requerimientos No Funcionales Para Aplicaciones Web. Argentina, ISSN. 2013. P. 2346–9927.

NAVARRO, G., 2001. A guided tour to approximate string matching. ACM computing surveys. (CSUR), 33(1), pp.37.

Pérez, D., 2014, La biblioteca digital. Universitat Oberta de Catalunya [en línea], [Consulta: 08 abril 2017]

PRESSMAN, R.S., 2005. Ingeniería de Software, Sexta Edición, Ed. S.l.: Mc Graw Hill, Mexico.

PYTEL, P., UHALDE, C., RAMÓN, H., CASTELLO, H., TOMASELLO, M., POLLO-CATTANEO, M., BRITOS, P. y GARCÍA MARTÍNEZ, R., 2011. INGENIERÍA DE REQUISITOS BASADA EN TÉCNICAS DE IN GENIERÍA DEL CONOCIMIENTO. XIII Workshop de Investigadores en Ciencias de la Computación. S.l.: s.n., pp. 426-429. ISBN 978-950-673-892-1.

ROGERS, Pressman, 2000, Ingeniería de Software Quinta Edición (un enfoque práctico), McGrawHill. Interamericana de España.

ROMMEL, F., 2007, SQLite: La Base de Datos Embebida, SG #17 Septiembre-Octubre 2007 [Consulta: 31 mayo 2017] Disponible en: https://sg.com.mx/revista/17/sqlite-la-base-datos-emb.

RUIZ, F. J., MONTERROSO, I. E. S., 2009. WEB 2.0: UN NUEVO ENTORNO DE APRENDIZAJE EN LA RED. DIM,13, 7.

RUIZ, J.H., ALMANZA, L.Á. y PONS, N.L., 2011. Comparación y tendencias entre metodologías ágiles y formales. Metodología utilizada en el Centro de Informatización para la Gestión de Entidades. Serie Científica [en línea], vol. 4, no. 10. [Consulta: 31 mayo 2017]. Disponible en: https://publicaciones.uci.cu/index.php/SC/article/view/484/0.

SÁNCHEZ, Tamara Rodríguez, 2014, Metodología de desarrollo para la actividad productiva de la UCI. 2014.

SICILIA, Miguel-Angel, 2014, Handbook of metadata, semantics and ontologies. World Scientific.

TKACZYK, D., SZOSTEK, P., DENDEK, P.J., FEDORYSZAK, M. and BOLIKOWSKI, L., 2014, CERMINE – Automatic Extraction of Metadata and References from Scientific Literature. In : 2014 11th IAPR International Workshop on Document Analysis Systems (DAS). April 2014. p. 217–221.