UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 3

Grupo de Investigación de Web Semántica

Componente para la extracción de metadatos bibliográficos a partir de corpus textuales en formato PDF

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

Autores:
Paul Núñez García
Osbel Zorrilla Rivera

Tutores:
MSc. Yusniel Hidalgo Delgado
Ing. Ernesto Ortiz Muñoz

La Habana, junio de 2016
"Año 58 de la Revolución"

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

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

_____	_____
Osbel Zorrilla Rivera	**Paul Nuñez García**
Autor	Autor
_____	_____
MSc. Yusniel Hidalgo Delgado	**Ing. Ernesto Ortiz Muñoz**
Tutor	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 desempeñó 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. Es 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 padres, que siempre han estado a mi lado apoyándome en todo y que siempre trabajaron para que yo pudiera terminar mis estudios en esta universidad.

A mi amada esposa que a lo largo de estos cinco años ha estado junto a mí, hombro con hombro dándome aliento para seguir adelante.

A mi pequeño Caleb que, aunque hoy es un bebé y no puede entender estas palabras, sé que un día lo hará y podrá ver cuanto lo amo, gracias por él Dios.

A mi abuela Olga que, aunque no se encuentre físicamente en este mundo, gran parte de lo que soy hoy se lo debo a ella.

A mi abuelo Jesús que tanto se preocupa por mi cuando estoy lejos de casa.

A mi querido hermano Osley, eres lo máximo bro, gracias por estar siempre ahí para mí.

A todos mis amigos de la universidad y a todos los profesores que me ayudaron a convertirme en un profesional de las Ciencias Informáticas.

A todos aquellos que siempre confiaron en mí.

Osbel.

A mi madre, mi vida, mi ángel guardián.

A mi padre, mi guía, mi orgullo.

A mi familia, mi apoyo.

A mis amigos.

Paul.

AGRADECIMIENTOS

Siempre dije que mi primer agradecimiento iba a ser para mi Dios, quiero agradecerle porque todo se lo debo a él, gracias Dios porque sin ti nada de lo que hoy es mi vida hubiese sido posible, gracias por estar a mi lado en medio de las pruebas durante estos angostos cinco años.

A mi mamá que siempre estuvo a mi lado y que me enseñó que de los malos momentos siempre se sale y que aunque los tiempos sean malos ella siempre iba a estar junto a mí. Gracias también a mí querido padre que tanto ha trabajado durante su vida para que yo pudiera tener lo que él nunca tuvo y para que fuese alguien en la vida.

A mi amada esposa Yainet que tantas cosas hemos tenido que soportar juntos, gracias por estos 5 años tan lindos de tu vida que me has dedicado, gracias por comprenderme siempre y por ir conmigo hasta el fin del mundo si fuera necesario, te amo.

A mi amado hermano Osley por estar a mi lado no en los momentos buenos, sino en los momentos cuando más lo necesite, al igual que su esposa, gracias Yede por cuidar tan bien a ese flaco que tanto quiero.

A todos mis amigos que me han acompañado durante esta etapa de mi vida y en especial a la gente del grupo Web Semántica, a Luis Manuel, Mariño, Moreira y a mi compañero de tesis Paul, gracias por todo mi hermano, aunque tú siempre supiste quien era el jefe.

A mi tutor que tanto lo mortificamos y tanto él nos mortificó a nosotros, gracias Yusniel porque a pesar de todo fuiste un buen tutor y supiste formarnos como verdaderos ingenieros.

A todo aquel que puso su granito de arena para hacer este sueño realidad.

Osbel.

AGRADECIMIENTOS

Gracias a mi madre por todo su amor, por su comprensión por su confianza y enseñarme que todo en la vida es posible solo hay que proponérselo.

Gracias a mi papa por impulsarme a convertirme en lo que soy, por su apoyo, por estar siempre ahí.

Gracias a mi hermano Ale por ayudarme a crecer, por ser mi mejor amigo y por enseñarme más cosas de las que en algún momento se pueda imaginar.

Gracias a mis abuelos que, aunque ya no pertenezcan a este mundo han sido motor impulsor de este futuro profesional que decidí llevar a cabo.

Gracias a mis tíos por ser ese hombro que en ocasiones he necesitado para apoyarme y seguir adelante.

Gracias a los viejos amigos irremplazables, gracias a ellos entendí que la verdadera amistad trasciende más allá del tiempo.

Gracias a todos los nuevos amigos que he tenido la dicha de conocer estos cinco años, con ellos todo fue más fácil.

Gracias al grupo de Investigación de Web Semántica por ayudarme a crecer como profesional, en especial a Alejandro, a Luis Manuel, Juan Carlos.

Gracias a mi compañero de tesis porque a pesar de los tropiezos y lo difícil que fue el camino, supo mantenerse fuerte y logramos seguir adelante.

Gracias a mi tutor Yusniel por enseñarme que lo que vales es lo que puedes llegar a aprender y la facilidad con la que te adaptas a los cambios que esta profesión regala tan frecuentemente.

Paul

RESUMEN

Los metadatos existen en numerosos contextos y estos se pueden reconocer por sus tipos, formas, características y usos. La extracción y almacenamiento de metadatos desde documentos en formato PDF han ganado en aplicabilidad al igual que la publicación de metadatos bibliográficos siguiendo los principios de los datos enlazados. El proceso de extracción de metadatos se vuelve complejo debido a que esta actividad generalmente requiere de personal altamente calificado. Este proceso se realiza manualmente en muchos casos, haciendo lento el proceso de digitalización y catalogación de los registros bibliográficos, por lo que es necesario contar con herramientas informáticas capaces de procesar, de manera semiautomática documentos en formato PDF, para posteriormente extraer sus correspondientes metadatos bibliográficos. El desarrollo de este tipo de herramienta constituye una solución viable para transformar un documento PDF o grupo de estos en metadatos, que luego pueden ser empleados por otros sistemas informáticos con fines específicos, tales como la búsqueda, recuperación de información y la clasificación de documentos. En esta investigación se propone un componente para la extracción de metadatos bibliográficos desde documentos en formato PDF, teniendo en cuenta los principales enfoques, técnicas y herramientas utilizadas para la extracción de metadatos en la actualidad. Con la implementación del componente se pretende dotar a los especialistas en bibliotecología de una herramienta de extracción de metadatos bibliográficos.

***Palabras claves*:** metadatos; extracción; aprendizaje automático; metadatos bibliográficos

ABSTRACT

There are metadata in many contexts. It can be recognized by types, forms, characteristics and uses. The extraction and storage of metadata from PDF documents has gained in applicability as the publication of bibliographic metadata following the principles of the linked data. The metadata extraction process becomes complex because this activity usually requires highly qualified personnel. The work of extraction is done manually by slow the process of digitizing and cataloging of bibliographic records, so it is necessary to have a software tool capable of processing, semi-automatically documents in PDF format, to later extract their corresponding bibliographic metadata. The development of such a viable tool to transform a PDF document or group of these metadata, which can then be used by other computer systems with specific, such as search, information retrieval and document classification purposes solution. In this research a component for extracting metadata from PDF documents taking into account the main approaches, techniques and tools used for extraction of metadata currently proposed. With the implementation of the component, it is to provide specialists in library science from an extraction tool.

Keywords: metadata, extraction, automatic learning, bibliographic metadata.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE TABLAS

ÍNDICE DE TABLAS

ÍNDICE DE TABLAS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Desde mediados del siglo XX la humanidad ha tratado de controlar, organizar, almacenar y utilizar de forma rápida y sencilla la información por medio de mejores sistemas de búsqueda y recuperación. Entre estos sistemas las computadoras jugaron un papel protagónico. El *Electronic Numerical Integrator And Computer*, Integrador y Computador Electrónico Numérico (ENIAC), primer ordenador digital universal totalmente electrónico, fue construido entre 1943 y 1946 en la Universidad de Pensilvania. Esta computadora era capaz de realizar varios cientos de multiplicaciones por minuto (Batista, Delgado, Bernardini 2015). Estas ideas sirvieron de base hacia una nueva concepción en la forma de almacenar, organizar y recuperar información con mayor facilidad, agilidad y accesibilidad. Estas cuestiones se concretarían años después con el advenimiento de las Tecnologías de la Información y las Comunicaciones (TIC), posibilitando que miles de usuarios accedan a ilimitados recursos de información (Buckland 2012).

En la actualidad, las TIC han alcanzado un vertiginoso desarrollo, convirtiéndose estas en el eslabón fundamental para alcanzar el éxito en diferentes esferas de la vida. Las nuevas tecnologías han promovido en millones de personas, hogares y oficinas, una comunicación electrónica mediante estándares universales y abiertos. Las empresas e instituciones generadoras de información, los archivos, las bibliotecas y los centros de documentación, por citar algunos ejemplos, se encuentran inmersas en un proceso de cambio de estrategias, debido a la relevancia que adquiere la tecnología en la gestión de sus procesos (Button, Harrington, Belan 2014). Con el desarrollo científico y tecnológico alcanzado por la humanidad, la forma de gestionar los documentos ha cambiado su visión en tal manera que el documento impreso está perdiendo protagonismo frente al documento electrónico. Esto se debe a que con la llegada del documento electrónico se han desarrollado también una gran variedad de herramientas informáticas que facilitan la gestión, organización y almacenamiento de dichos documentos (Jones, Sunderland, Sawicki, Little, Davis 2015). Entre estas herramientas se encuentran precisamente las Bibliotecas Digitales.

La concepción que se tiene de bibliotecas tradicionales es ofrecer documentos (libros, revistas, tesis, artículos, etc.) en soportes físicos a través de servicios de préstamos y consultas. Estas políticas limitaban de alguna manera el acceso a los documentos por la falta de ejemplares o la existencia de ejemplares desactualizados. No obstante, unido al incremento de los recursos informáticos, la Internet y el descenso de los costos para adquirir esos recursos y servicios relacionados, se ha potenciado en los últimos 20 años el diseño y la creación de las Bibliotecas Digitales (Hernández, Agenjo 2010). En este sentido, se inició la automatización de las bibliotecas tradicionales, con un crecimiento sostenido y en constante evolución.

Tales cambios se pueden constatar a partir de varias conferencias internacionales de referencia en el área entre las cuales se pueden nombrar: *Joint Conference on Digital Libraries* (JCDL), *Theory and Practice of Digital Libraries* (TPDL) antes conocida como *European Conference on Research and Advanced Technology for Digital Libraries* (ECDL), *International Conference on Asian Digital Libraries* (ICADL), IEEE/TCDL (*Technical Committee on Digital Libraries*). En los trabajos presentados en estas conferencias, en las publicaciones de revistas del área y en 128 revistas en *SCImago Journal & Country Rank* para febrero del 2013 en la categoría *"Library and Information Sciences*" (Scimago Journal & Country Rank 2013), se evidencia la diversidad de conceptos y descripciones que abarcan las palabras "bibliotecas digitales" o *"digital library".*

Según la bibliografía consultada, a partir de la década de los 90 comienzan a surgir las Bibliotecas Digitales, principalmente, gracias a proyectos de investigación financiados por agencias gubernamentales y organismos nacionales e internacionales. Otros proyectos específicos estuvieron a cargo de instituciones académicas, de investigación y de bibliotecas, individualmente o en colaboración entre las Universidades de Michigan, Stanford, Berkeley, Santa Bárbara, Illinois y Carnegie Mellon (Griffin 1998).

En Estados Unidos de América surge la *Digital Library Initiative* (DLI-1) integrada por la *National Science Foundation* (NSF), la *Defense Advanced Research Projects Agency* (DARPA) y la *National Aeronautics and Space Administration* (NASA). El objetivo de DLI-1 fue desarrollar e implementar modelos de bibliotecas digitales para almacenar y recuperar documentos científicos a través de las redes de comunicación. Luego de los logros obtenidos por la DLI-1, se anuncia un nuevo programa que fue llamado *Digital Libraries Initiative Phase* 2 (DLI-2), conformado por la *National Library of Medicine* (NLM), la *Library of Congress* (LC), la *Federal Bureau of Investigation* (FBI) y la *National Endowment for the Humanities* (NEH), además de los organismos que llevaron a cabo la DLI-1 (Griffin 1999). Los resultados de estas dos iniciativas permitieron la consolidación y estudio de nuevos estándares aplicados hoy en día en las Bibliotecas Digitales.

Las bibliotecas se clasifican en tres categorías conocidas: las bibliotecas convencionales, las bibliotecas digitales y las bibliotecas híbridas. Las bibliotecas convencionales operan a través de colecciones impresas, mientras que las bibliotecas digitales recogen, almacenan y comunican la información en formato digital o electrónico (Bergman, Afifi, Miyauchi 2014). La combinación de ambas bibliotecas, convencional y digital, son las bibliotecas híbridas, donde se almacena la información tanto en formatos impresos como en formatos digitales. Según (Singh, Sharma 2015), una biblioteca electrónica (también denominada como biblioteca digital (DL) o repositorio digital) no es más que una colección de objetos digitales que pueden incluir texto, material visual, material de audio y material de

vídeo, almacenado en formato electrónico, además de un conjunto de herramientas o recursos que permiten organizar, almacenar y recuperar los archivos y los medios contenidos en la colección de la biblioteca.

En las bibliotecas digitales también se recogen las actas de los congresos, que constituyen recopilaciones de las ponencias y comunicaciones de congresos, simposios, seminarios, memorias de eventos científicos, etc. Estas son editadas en general por la entidad organizadora y en ellas se dan a conocer por primera vez los resultados de muchos trabajos de investigación. Generalmente estas actas de congreso se almacenan en bibliotecas digitales en formato PDF (*Portable Document Format*), que aparece como una alternativa para la creación de documentos electrónicos.

El PDF es un formato de almacenamiento de documentos digitales independiente de las plataformas de software o hardware. Estos documentos pueden contener cualquier combinación de texto, elementos multimedia como vídeos o sonido, elementos de hipertexto como vínculos y marcadores, enlaces y miniaturas de páginas, lo que lo hace el formato más difundido en el internet para el intercambio de documentos electrónicos (Adobe Acrobat DC 2015). En Internet se manejan grandes volúmenes de información, la cual años atrás hubiese sido almacenada en formato duro, pero con el uso actual de la tecnología se tiene una gran cantidad de documentos digitales formando varios corpus en formato PDF. En este sentido, se han desarrollado también una gran variedad de herramientas informáticas que facilitan la gestión, organización y almacenamiento de dichos documentos dentro de las bibliotecas digitales. Los cambios en este ámbito han hecho de la gestión de metadatos una técnica a tener en cuenta para el trabajo con información de cualquier tipo.

Los metadatos en sí, no suponen algo completamente nuevo dentro del mundo bibliotecario. Según Howe (1993), el término fue acuñado por Jack Myers en la década de los 60 para describir conjuntos de datos. La primera acepción que se le atribuyó (y actualmente la más extendida) fue la de dato sobre el dato, ya que proporcionaban la información mínima necesaria para identificar un recurso. En este mismo trabajo se afirma que puede incluir información descriptiva sobre el contexto, calidad y condición o características del dato (Senso, Piñero 2003).

Desde que surgió la idea de los metadatos en las bibliotecas a mediados del pasado siglo, estos se empleaban únicamente como referentes al proceso automatizado y a la descripción de la información contenida en las bases de datos. Con los cambios en el uso y creación de la información que se origina con los recursos electrónicos, la actividad de los metadatos toma una importancia relevante en las comunidades bibliotecarias (Missier, Belhajjame, Cheney 2013). A pesar de que se ha dicho en la literatura especializada que los metadatos existen en numerosos contextos y estos se pueden

reconocer por sus tipos, formas, características y usos, aún existe incertidumbre sobre su desenvolvimiento en la tecnología de la información y la bibliotecología. Además, reflejan patrones que determinan las propiedades de los actuales paquetes de información que circulan en el entorno Web (García 2013).

La extracción y almacenamiento de metadatos desde documentos en formato PDF ha ganado en aplicabilidad al igual que la publicación de metadatos bibliográficos siguiendo los principios de los datos enlazados (Choudhury, Mitra, Kirk, Szep, Pellegrino, Jones, Giles 2013). La gestión de metadatos bibliográficos siempre ha estado sujeta a las autoridades centrales como las bibliotecas y los editores. Sin embargo, con el avance de las redes social de investigadores como Mendeley y herramientas de marcadores sociales como CiteULike, la gestión de metadatos es cada vez más descentralizada. Teniendo en cuenta que estos metadatos son publicados utilizando alguna de las serializaciones del formato RDF, un estándar de la W3C para la descripción de recursos en la Web, se han propuesto diversos algoritmos, técnicas y herramientas para la extracción de metadatos bibliográficos a partir de documentos académicos en formato PDF (Granitzer, Hristakeva, Knight, Jack, Kern 2012).

En el grupo de investigación de Web Semántica de la Universidad de las Ciencias Informáticas (UCI) se desarrolla el proyecto de investigación **Extracción, Publicación y Consumo de metadatos bibliográficos como datos enlazados.** El objetivo del proyecto es la extracción, publicación y consumo de metadatos bibliográficos siguiendo los principios de los datos enlazados. Una etapa importante del proyecto es la fase de extracción de metadatos bibliográficos a partir de documentos en formato PDF. Estos documentos provienen de artículos científicos publicados en revistas académicas y provienen además de memorias de eventos que son distribuidos en CD, donde todos los documentos pertenecientes a un mismo CD poseen la misma estructura y formato.

En esta fase, es imprescindible contar con una herramienta informática capaz de procesar, de manera semiautomática documentos en formato PDF para posteriormente extraer sus correspondientes metadatos bibliográficos. Esta actividad generalmente requiere de personal altamente calificado, el cual realiza esta labor manualmente, haciendo lento el proceso de digitalización y catalogación de los registros bibliográficos.

Teniendo en cuenta lo anteriormente planteado surge como problema a resolver: ¿Cómo ***extraer metadatos bibliográficos a partir de documentos en formato PDF*** de manera que se logre ***reducir el tiempo empleado en este proceso por parte de los especialistas en bibliotecología?***

Para ello se centra la investigación en el **objeto de estudio:** la extracción de metadatos bibliográficos enmarcado en el **campo de acción:** extracción de metadatos bibliográficos mediante técnicas de aprendizaje automático.

Para dar solución al problema planteado se define como **objetivo general**: desarrollar una herramienta informática que permita la extracción de metadatos bibliográficos a partir de corpus textuales en formato PDF, utilizando un enfoque semiautomático, que contribuya a la reducción del tiempo empleado en este proceso por parte de los especialistas en bibliotecología.

Por lo que se define como **idea a defender** que**:** si se desarrolla una herramienta informática capaz de extraer metadatos bibliográficos de documentos en formato PDF, entonces se contribuirá a la reducción del tiempo empleado en este proceso por parte de los especialistas en bibliotecología.

Para dar cumplimiento al objetivo general de la investigación se definen los siguientes **objetivos específicos**:

- 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.
- Diseñar un componente informático para la extracción de metadatos bibliográficos a partir de documentos en formato PDF utilizando un enfoque semiautomático.
- Implementar un componente informático para la extracción de metadatos bibliográficos a partir de documentos en formato PDF utilizando un enfoque semiautomático.
- Validar los resultados obtenidos con la utilización del componente informático desarrollado mediante la realización de un diseño experimental.

Para la realización de la investigación se emplearon los siguientes **métodos científicos:**

Métodos teóricos:

• Analítico-Sintético: se utiliza para el análisis de la literatura científica, materiales y temas relacionados con el objeto de estudio. Este análisis permitió realizar una síntesis de los elementos más importantes relacionados con estos temas, lo que facilitó la selección de las herramientas y tecnologías necesarias para el desarrollo de la propuesta de solución.

• Histórico-Lógico: se utiliza para desarrollar un estudio del estado del arte de la problemática y conocer cuáles son las herramientas o sistemas que existen actualmente en el mundo para la extracción de

metadatos bibliográficos desde documentos en formato PDF y cómo han venido comportándose en los últimos 5 años.

- Modelación e Inductivo-Deductivo: para la fundamentación y elaboración de la propuesta de solución, donde se pretende la modelación de la propuesta utilizando técnicas de visualización de la información, así como algunos artefactos de ingeniería que ayuden a comprender sus componentes y sus interrelaciones.

Métodos empíricos:

- Experimentación: se utiliza para la realización de los experimentos diseñados con el objetivo de validar la propuesta de solución.

- Medición: se utiliza en el cálculo de la efectividad de la propuesta de solución y de la calidad de las respuestas finales.

El presente Trabajo de Diploma posee la siguiente estructura:

En el **Capítulo 1** se realiza un análisis bibliométrico y documental de la bibliografía consultada para realizar el trabajo. Se analizan además los principales conceptos y definiciones asociados al dominio del problema, los cuales sirven de apoyo durante el desarrollo de la investigación. Como aspecto medular se ofrece un enfoque de los términos fundamentales relacionados con la extracción de metadatos. Por último, se hace un análisis del uso de la extracción de metadatos bibliográficos desde documentos en formato PDF en el resto del mundo.

En el **Capítulo 2** se presenta un componente de software que permite extraer metadatos bibliográficos desde documentos en formato PDF. Para garantizar lo anterior se realiza una descripción general de la propuesta de solución y se define la metodología de desarrollo de software a utilizar para obtener el producto final. 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.

En el **Capítulo 3** se desarrolla un caso de estudio en un contexto real de utilización del componente propuesto, empleando documentos en formato PDF que han sido publicados en revistas y eventos desarrollados en nuestro país. Adicionalmente, se realiza un pre-experimento para evaluar los tiempos de respuesta de la aplicación al realizar las diferentes tareas de extracción de metadatos solicitadas por los usuarios 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.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En este capítulo se enuncian los conceptos fundamentales sobre la extracción de metadatos bibliográficos a partir de documentos en formato PDF, los cuales permitirán un mejor entendimiento del presente trabajo. Por último, se muestra un análisis del estado del arte de las principales aproximaciones existentes en la literatura para la gestión de metadatos bibliográficos.

1.2 Análisis bibliométrico y documental

Para la realización de la presente investigación se llevó a cabo un estudio documental 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[1], IEEE[2], ACM[3] además se visitan varios sitios web oficiales de herramientas para la extracción de metadatos desde documentos en formato PDF para conocer cuáles son las tecnologías actualmente más eficientes y recomendadas para realizar el trabajo. A continuación, se muestra un gráfico resumen de la bibliografía consultada, ver Figura 1.

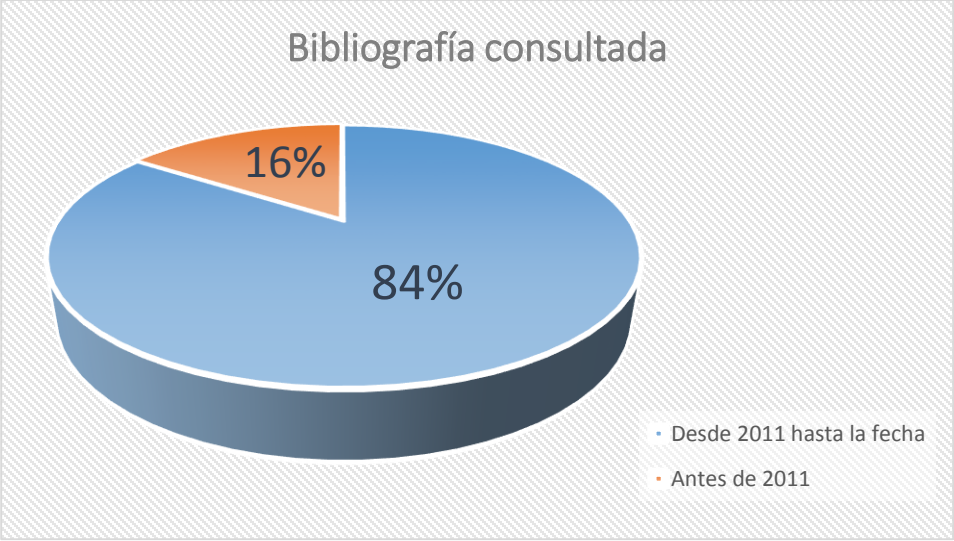


Figura 1: Análisis bibliométrico y documental.

La ilustración anterior muestra que se consultaron un total de 45 bibliografías distintas, de las cuales 38 de ellas fueron publicadas en los últimos 5 años lo cual representa un 84 % aproximadamente del

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

total de la bibliografía consultada, los 7 restantes que representan un 16% son anteriores a los últimos 5 años.

1.3 Marco teórico

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

1.3.1 Metadatos

El término **metadatos** se define a menudo como "datos sobre los datos". Esta definición básica no es muy informativa, sin embargo según (Tkaczyk, Szostek, Dendek, Fedoryszak, Bolikowski 2014), los metadatos son datos altamente estructurados que describen información, describen 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, others 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, Szostek, Dendek, Fedoryszak, Bolikowski 2014) quien hace una disertación más amplia sobre el término metadatos, debido a ello es que se adopta su definición como la que más se ajusta a las características de la investigación. En la actualidad, estos metadatos están contenidos en corpus electrónicos, los cuales emergen como una alternativa para el almacenamiento de grandes cantidades de información, convirtiéndose estos en contenedores de información y por consiguiente de metadatos con valiosa información.

1.3.2 Corpus textuales

Según el Diccionario de la Lengua Española de la Real Academia se define el **corpus** como*:* conjunto lo más extenso y ordenado posible de datos o textos científicos, literarios, etc., que pueden servir de base a una investigación (Sánchez 2006).

Por otra parte (Kennedy 2014) afirma que la definición de un corpus como una colección de textos en una base de datos electrónica puede acarrear muchas preguntas, ya que hay muchos tipos diferentes de corpus. Algunas definiciones del diccionario sugieren que los corpus necesariamente consisten en

colecciones estructuradas de texto compilados específicamente para el análisis lingüístico, que son grandes o que intentan ser representativos de una lengua en su conjunto.

Al realizar un análisis de las definiciones anteriores, ambos autores coinciden en que los corpus son colecciones de textos, pero en el caso de la definición propuesta por (Sánchez 2006) es la que más se ajusta a las características de la investigación. La creación de corpus textuales, aunque data de varios años, es hoy cuando forman parte de la solución al almacenamiento de grandes volúmenes de texto; estos corpus contienen metadatos que son fundamentales en su organización y elaboración. Con el vertiginoso desarrollo actual resulta de gran importancia la extracción de estos metadatos de la manera menos engorrosa posible, obteniéndose apoyo de las tecnologías existentes.

1.3.3 Extracción de metadatos.

Debido a la relevancia que han tomado los metadatos, se hace necesaria su extracción desde los documentos digitales. La extracción automática de metadatos consiste en obtener un conjunto de atributos o elementos que describan documentos digitales. Los metadatos extraídos se utilizan para la descripción e identificación de los materiales digitales (Zhang, Zhao 2013). Una vez que este proceso se realiza, estos elementos pueden ser depositados en una base de datos, repositorio o en un lugar donde se almacena información digital con el objetivo de preservarla, generalmente se utiliza el formato RDF (*Resource Description Framework)* y estos metadatos son organizados utilizando tripletas (Lösch, Bloehdorn, Rettinger 2012).

Los metadatos pueden ser clasificados en tres tipos:

Metadatos descriptivos: se utilizan para la descripción e identificación de la información contenida en el recurso. Contienen atributos físicos (medios, condición de las dimensiones) y atributos bibliográficos (título, autor/ creador, idioma, palabras claves).

Metadatos administrativos: se refieren a las características y propiedades del recurso, facilitando la gestión, procesamiento tecnológico y físico de las colecciones digitales, tanto a corto como a largo plazo. Incluyen información sobre la creación y el control de la calidad, la gestión de derechos, el control de acceso, la utilización y las condiciones de preservación.

Metadatos estructurales: proporcionan información sobre la estructura interna de los recursos electrónicos, tales como: página, sección, capítulo, partes, índices, tabla de contenidos, etc. y describen la relación entre los materiales. Facilitan la navegación y presentación de los recursos y relacionan las diferentes partes que lo componen (Testa, Ceriotto 2011).

De los tres tipos de metadatos existentes, en esta investigación se utilizan los metadatos descriptivos, específicamente sus atributos bibliográficos, ya que estos son los atributos que se desean extraer específicamente desde los corpus textuales en formato PDF.

1.4 Principales aproximaciones

Entre las principales aproximaciones para la extracción de metadatos se encuentran las basadas en la estructura del documento. La estructura del documento hace uso de un enfoque visual en sus páginas, como por ejemplo resaltar el título, el autor y la afiliación, elementos que generalmente aparecen en el encabezado.

1.4.1 Fuentes de metadatos

En la bibliografía consultada según (Sicilia 2014) aparecen fundamentalmente 5 estructuras a partir de las cuales se pueden extraer los metadatos. A continuación, se hace mención a cada una de ellas.

- **Estructura de formato:**

En esta categoría se encuentran documentos que tienen un formato electrónico definido, por ejemplo, en un documento en formato HTML[4] se pueden encontrar etiquetas como la etiqueta <TITLE>.

- **Estructura visual:**

Este tipo de documentos tienen definido una estructura visual, en esta clasificación se encuentran los documentos en formato PDF, especificando cómo está ubicado el texto en la página, esto puede ser utilizado para identificar las secciones del texto.

- **Disposición del documento:**

El documento puede estar estructurado siguiendo una disposición específica, por ejemplo, comenzar con el título, luego los autores y finalizar con las referencias bibliográficas.

- **Análisis de citaciones bibliográficas:**

Los documentos que están relacionadas entre sí a través de la vinculación de citas o el análisis de co-autoría pueden ser analizados a través de métodos bibliométricos.

[4] HTML: HyperText Markup Language

- **Estructura Lingüística:**

El documento se puede analizar lingüísticamente, permitiendo deducir el sentido de las partes de las oraciones, o relaciones entre los metadatos. Por ejemplo, las citas en el texto principal pueden estar contenidas dentro de la misma oración, lo que indica que las dos citas probablemente estén relacionadas de alguna manera. La relación puede ser positiva o negativa, dependiendo del texto que lo rodea, por lo que el análisis de la estructura lingüística depende del conocimiento del idioma del documento y posiblemente del conocimiento del dominio. Utilizando un análisis lingüístico se puede intentar extraer las palabras claves y las relaciones entre las citas.

En relación a la estructura visual, existen ventajas para HTML, como por ejemplo, el trabajo con etiquetas haciendo más fácil el reconocimiento de estructuras. En formatos como el PDF resulta más engorroso ya que especifica símbolos, palabras o ubicaciones de una página y el documento consta de una bolsa de símbolos o palabras en lugares específicos. La estructura del documento se puede inferir de las ubicaciones del símbolo. La desventaja fundamental de estos lenguajes de descripción es que existen múltiples maneras para presentar los textos (Sicilia 2014).

Una vez analizado las diferentes fuentes desde las cuales se pueden extraer metadatos, se determina que la fuente que más se ajusta a la investigación realizada es la **Estructura visual**, ya que como se mencionó anteriormente dentro de los documentos que entran en esta clasificación se encuentran precisamente los PDF, los cuales al especificar cómo está ubicado el texto en la página, permiten que esta característica pueda ser utilizada para identificar las secciones o bloques de texto dentro del documento.

1.4.2 Enfoques para la extracción de metadatos

Según (Sicilia 2014) los metadatos pueden ser extraídos utilizando diferentes enfoques. Estos son:

- **Máquina de soporte de vectores**

Una máquina de soporte de vectores (MSV) intenta encontrar un hiperplano de separación óptima a un máximo de dos clases separadas de muestras de entrenamiento. La función de decisión correspondiente para determinar dicho hiperplano se denomina un clasificador. La función del núcleo de una (MSV) puede ser un producto interno, gaussiano, polinomio, o cualquier otra función que obedece a condiciones del teorema de Mercer. Por otra parte, el teorema de Mercer constituye una representación de una función definida positiva simétrica en un cuadrante, como la suma de una secuencia convergente de las funciones del producto (Meyer, Wien 2015).

- **Campo Aleatorio Condicional (CRF)**

CRF es un marco etiquetado de secuencia estadística propuesto por (Lafferty, McCallum, Pereira 2001) para el etiquetado gramatical y el análisis sintáctico. CRF supera a otros modelos más populares, tales como los HMM (Modelo Oculto de Markov) y los modelos de máxima entropía, cuando la verdadera distribución de datos tiene dependencias de orden más altos que los modelos, lo cual es a menudo el caso en circunstancias prácticas. Por otra parte, los CRF han obtenido buenos resultados en muchos estudios en campos que van desde la bioinformática hasta el procesamiento del lenguaje natural (Ohta, Arauchi, Takasu, Adachi 2014).

- **Modelo Oculto de Markov**

El modelo oculto de Markov provee una solución a la aparición de términos ambiguos o que aparecen en distintas posiciones de un artículo, creando una máquina de estado o cadena de Markov, en la que, con el conocimiento previo, se especifican secuencias probables de *tokens* (palabras). Por ejemplo, una secuencia de probabilidades para un artículo científico comprende un número de *tokens* de título, seguido de una nueva línea, seguido de un número de fichas de autor, seguido de un carácter de nueva línea, seguido por un número de fichas de afiliación (Martínez 2014). Los Modelos Ocultos de Markov son el método de aprendizaje generativo más ampliamente utilizado para representar y extraer información de los datos secuenciales. Sin embargo, se basan en la suposición de que características del modelo que representan, no son independientes unas de otras. Por lo tanto, los Modelos Ocultos de Markov tienen dificultades para la explotación de las regularidades de un sistema real semiestructurado, por lo que se han introducido modelos basados en la entropía máxima y campos aleatorios condicionales para hacer frente al problema de características independientes (Maji 2015).

1.4.3 Técnicas para la extracción de metadatos

Existen diferentes enfoques para la extracción de metadatos, los cuales fueron identificados en la bibliografía consultada, según (Sicilia 2014) estos enfoques son:

- **Aplicación directa de la heurística observada**

En ciertos casos, la naturaleza semiestructurada de los datos hace que sea posible lograr una precisión razonable en la extracción de datos mediante la aplicación de un conjunto de heurísticas, reglas que abarcan las estructuras o los diseños más comunes. Esto es particularmente común en el caso de la extracción de metadatos de los documentos HTML / XHTML. Aquí cada elemento de texto está rodeado de marcado semi-semántico. La aspiración es que ninguna información sobre la disposición está

contenida en el documento XHTML en sí, pero sí dentro de un acompañamiento de hojas de estilo en cascada a las que hace referencia el documento. Sin embargo, en la práctica este no siempre es el caso y el marcado utilizado puede variar según del software utilizado para generar la información.

- **Concordancia de patrones**

Un patrón comprende una descripción de cómo construir una secuencia de caracteres y haciendo coincidir repetidamente el patrón sobre un texto. Estos patrones vienen a ser útiles para describir las entidades que se producen a menudo y se identifican de forma más fiable por un patrón que por cualquier otro método, por ejemplo, la dirección de correo que sigue una estructura para todo tipo de documento y se puede expresar en función de una expresión regular o en casos complejos a través de una gramática. La debilidad de los patrones es que es muy difícil construir un patrón que coincida de manera precisa, sin perder ningún elemento, por ejemplo, el patrón tiende a fallar cuando se especifica {nombre}@mail.com que es una forma de especificar en un artículo que varias personas tienen una dirección de correos en el mismo servidor, en este caso **mail.com**.

- **Clasificación**

Un clasificador consiste en clasificar las palabras encontradas en un texto basándose en probabilidades matemáticas que permitan decidir qué tipo de palabra se encuentra clasificando. Los clasificadores son útiles en la extracción de nombre de autor porque los nombres son típicamente palabras muy diferentes de las palabras comunes que se encuentran en el cuerpo principal de un artículo. Por ejemplo, "Ruud", "John", y "Canagarajah" es muy probable que sean nombres mientras que "siguiente", "clasificador" y "condena" es muy poco probable que sean nombres. Su principal debilidad es que existe una pequeña probabilidad, pero con frecuencia se produce, que un grupo de palabras podría pertenecer a cualquier clasificación. Por ejemplo, "A" es una palabra que se produce con mucha frecuencia en texto inglés, pero también podría ser una inicial de un nombre. Una de las diversas variantes de clasificadores es el clasificador bayesiano, el cual tiene una base matemática sencilla y fuerte que es completamente independiente del problema. Cuando es utilizado en el texto, funciona tan bien en francés como texto en chino, con la condición de que los datos dispongan de formación adecuada.

- **Ajuste del modelo**

En muchos casos existe un conocimiento previo acerca de los metadatos que se desean extraer, por lo que el ajuste del modelo se basa en aplicar la información previa existente como conocimiento del dominio para construir un conjunto de modelos para su uso en la extracción de los metadatos. Lo que

ocurre es que sería un proceso largo y difícil, por lo que existen técnicas de aprendizaje automático que a menudo se utilizan para resolver este problema.

- **Obtención de la estructura gramatical**

Es una técnica idealmente automatizada que fundamentalmente permite el análisis probabilístico. Un ejemplo se proporciona sobre la base de Modelos Ocultos de Markov**.**

1.5 Principales herramientas existentes para la extracción de metadatos

A continuación, se describen algunas de las herramientas encontradas en la literatura y que guardan relación con el objeto de estudio de la investigación.

1.5.1 Apache Tika

Apache Tika[5] es un conjunto de herramientas para detectar y extraer metadatos y texto estructurado del contenido de varios tipos de documentos usando librerías de *parser* existentes. El *kit* de herramientas Apache Tika detecta y extrae los metadatos y el texto de diferentes tipos de archivos (por ejemplo, PPT, XLS y PDF). Todos estos tipos de archivos se pueden analizar a través de una única interfaz, haciendo Tika útil para la indexación de documentos en los motores de búsqueda, análisis de contenido, traducción, y mucho más (Mattmann, Zitting 2011).

Tika es un proyecto de la *Apache Software Foundation* por lo que está escrito en Java, que es donde se obtiene la mayor parte de su flexibilidad y expresividad. Apache Tika 1.12 es la última versión existente del software al momento de escribir esta tesis e incluye varias mejoras que utilizan un mejor soporte para Java 1.7 que ayudan a extraer más contenido (Mattmann, Zitting 2011). La herramienta puede extraer metadatos en los siguientes formatos:

- HTML.
- XML y derivados.
- .doc, .xls, .ppt (formatos de documentos de Microsoft Office).
- .odt (Formato OpenDocument de OpenOffice).
- .pdf (Portable Document Format).
- .epub (formato para libros electrónicos).
- .rtf (Rich Text Format).
- Formatos comprimidos (ar, cpio, tar, zip, gzip, bzip2 y zip).

[5] https://tika.apache.org/index.html

- .txt (detectando el juego de caracteres).
- Audio (.mp3, .mid, .wav).
- Imagen (.jpeg).
- Video (.flv).
- Ficheros Java.

Tipo de Licencia:
Se distribuye bajo la Licencia Pública Apache (versión 2.0).

1.5.2 GROBID

Según (Lopez, Romary 2015), GROBID es una herramienta para analizar documentos técnicos y científicos, que se enfoca en la extracción automática de datos bibliográficos, como por ejemplo, encabezados, citas, y en la identificación de estructuras, sección de títulos y figuras. Para llevar a cabo la extracción de términos se basa en el Aprendizaje Automático, (Machine Learning, por sus siglas en inglés). 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 ML 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 ML.

Uno de los objetivos de GROBID es realizar conversiones fiables de documentos técnicos y científicos en formato PDF. Esta conversión implica primero el reconocimiento de las diferentes secciones del documento, a continuación, la extracción de todos los metadatos de cabecera y referencias. Después de la selección de un documento PDF, GROBID extrae los metadatos bibliográficos correspondientes a la información del encabezado (título, autores, resumen, etc.) y a cada referencia (título, autores, título de la revista, número, etc.). Las referencias están asociadas a sus respectivos contextos de citas. El resultado de la extracción de la citación puede ser exportado como un todo o por referencia atendiendo a los diferentes formatos existentes (Rendón-Miranda, Arana-Llanes, González-Serna, González-Franco 2014).

1.5.3 LA-PDFText

Según (Ramakrishnan, Patnia, Hovy, Burns 2012a), esta herramienta facilita la extracción precisa de texto de archivos PDF de los artículos de investigación para su uso en aplicaciones de minería de texto. Este sistema de código abierto extrae los bloques de texto de los artículos de investigación en formato PDF y los clasifica en unidades lógicas basadas en reglas que caracterizan a secciones específicas. El sistema LAPDFText se centra sólo en el contenido textual de los artículos de investigación. La versión actual es un sistema que cuenta con una interfaz en línea de comandos que extrae el texto mediante un proceso de tres etapas:

1. Identificación de los bloques de texto contiguo.
2. Clasificación de estos bloques en categorías retóricas.
3. Extracción del texto a partir de bloques agrupados por secciones.

El objetivo final de la LA-PDF Text es extraer con precisión el texto de cualquiera de las secciones en la secuencia correcta. Además, proporciona mecanismos para la salida del texto de estos PDF como formato XML utilizando DTD OpenAccess de PubMed Central (Ramakrishnan, Patnia, Hovy, Burns 2012).

Disponibilidad y Requerimientos

- Versión actual: 1.7
- Sistema operativo: MacOSX 10.6.7, Linux y Windows XP
- Lenguaje de programación: Java 1.6
- Licencia: Licencia Pública General de GNU

1.5.4 CERMINE

Según (Tkaczyk, Szostek, Fedoryszak, Dendek, Bolikowski 2015), CERMINE es una herramienta para la extracción de metadatos y las referencias bibliográficas directamente desde un archivo PDF, se basa en un flujo de trabajo modular compuesto de tres vías y una serie de pasos con entradas y salidas cuidadosamente definidas. En virtud de tal arquitectura de flujo de trabajo el paso individual se puede mantener por separado. Como resultado, es fácil de realizar la evaluación, mejorar o reemplazar una implementación de un paso sin cambiar otras partes del flujo de trabajo. Para la mayoría de las implementaciones se utilizan técnicas de aprendizaje automático supervisadas y no supervisadas, que aumentan la capacidad de mantenimiento del sistema, así como su capacidad de adaptación a nuevos diseños de documentos. CERMINE además es de código abierto y disponible para su uso como un

servicio web. La flexibilidad del sistema es concebida por su arquitectura modular y el uso de técnicas de aprendizaje automático.

Esta herramienta se basa en 3 etapas fundamentales, las cuales son:

1. Según la estructura del camino básico se toma un archivo PDF en la entrada y se produce una estructura jerárquica geométrica que representa el documento en formato TrueViz. La estructura se compone de páginas, zonas, líneas, palabras y caracteres, junto con sus coordenadas y dimensiones. El orden de lectura de todos los elementos se calcula. Cada zona está marcada con una de las cuatro categorías generales: metadatos, las citas, el cuerpo (publicaciones de texto, tablas) y otros (reconocimientos, declaraciones, números de página, entre otros).
2. Se realiza un análisis de la ubicación de los metadatos en la estructura jerárquica geométrica y se extrae un conjunto de metadatos del documento a partir de ellos.
3. Por último, analizando la trayectoria de las referencias en el documento, estas son igualmente extraídas.

Uno de los cambios más importantes desde la primera versión del flujo de trabajo es la introducción de dos caminos paralelos de ejecuciones. Por un lado, la extracción de metadatos y por otro la extracción de las referencias. Una vez que la estructura jerárquica geométrica se extrae del archivo de entrada y las categorías generales se asignan a todas las zonas, el tratamiento posterior puede ejecutarse en paralelo.

1.6 Análisis de herramientas

Según (Tkaczyk, Szostek, Fedoryszak, Dendek, Bolikowski 2015), se utilizó un subconjunto de 1.943 documentos para comparar el rendimiento de varios sistemas de extracción. Se evaluaron cuatro tareas:

1. Extraer cadenas autor de un artículo dado.
2. Extraer cadenas de afiliación de un determinado artículo.
3. Determinar las relaciones autor-afiliación en un artículo dado.
4. Determinar las relaciones autor-afiliación, siempre que los autores y afiliaciones fueron extraídos sin problemas.

Durante la evaluación se utilizó la versión por defecto de CERMINE, en el que los clasificadores de zona son entrenados en un subconjunto del conjunto de datos GROTOAP2 y el analizador de afiliación es entrenado con una base de datos de afiliaciones. Aparte de CERMINE, se evaluaron los siguientes

sistemas: **GROBID**, **ParsCit** y **PDFX**. Se utilizaron las versiones por defecto de las herramientas. PDFX fue ejecutado a través de su servicio web. La única excepción fue ParsCit, que analiza solo el contenido de texto de un documento, por lo tanto, en este caso, los archivos PDF se transformaron primero en texto usando la herramienta **pdftotext**. Los resultados se obtuvieron comparando las salidas proporcionadas por las herramientas con los datos comentados de archivos NLM.

Por cada documento y cada tarea se compararon las listas y grupos de objetos que fueron verdaderamente extraídos. Las cadenas de autor y afiliación se compararon utilizando la distancia coseno con un umbral. En el caso de las relaciones autor-afiliación, una relación estuvo marcada como correcta si ambos elementos coincidían, teniéndose la precisión individual para cada documento. Los valores globales de precisión se calcularon a partir de su valor medio en todos los documentos del conjunto de datos. Los resultados de la evaluación se muestran en la Figura 2. En ambas tareas de extracción de autor y de afiliación, CERMINE logra la mejor puntuación (89,2% y 84,3%, respectivamente). En el caso de la determinación de las relaciones autor-afiliación se compararon CERMINE y GROBID, debido a que las otras herramientas no extraen esta información. En ambas tareas CERMINE consigue mejores resultados (63,1% y 77,4% en las tareas 3 y 4, respectivamente).

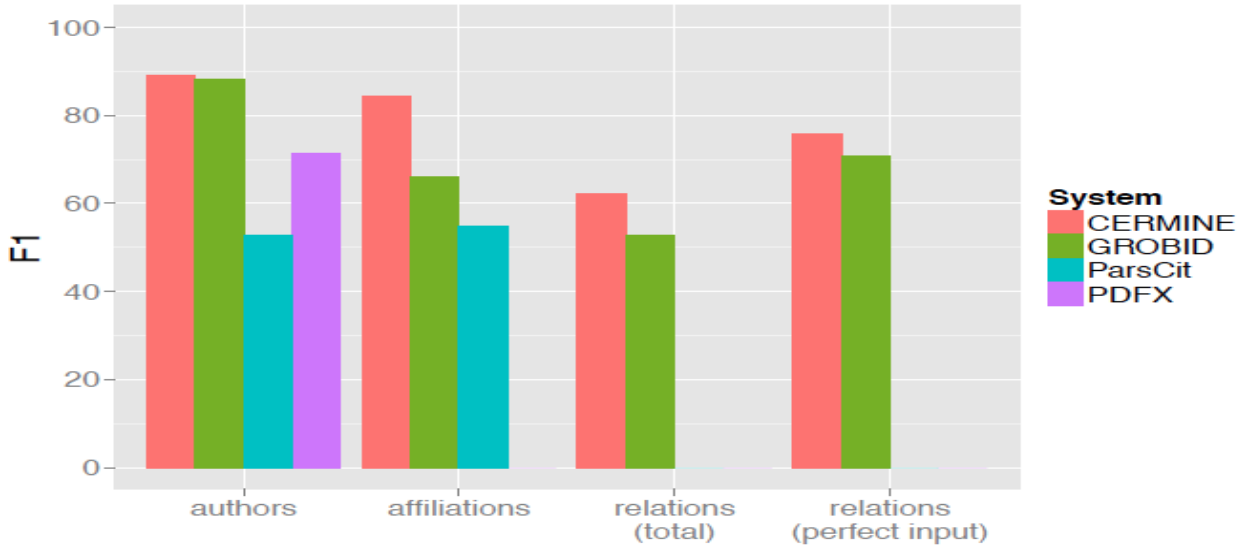


Figura 2: Resultados de la comparación de extracción de metadatos de las herramientas. Fuente: (Tkaczyk, Szostek, Fedoryszak, Dendek, Bolikowski 2015).

El tiempo necesario para extraer los metadatos de un documento depende principalmente de su número de páginas. La Figura 3 muestra el tiempo de procesamiento en función del número de páginas

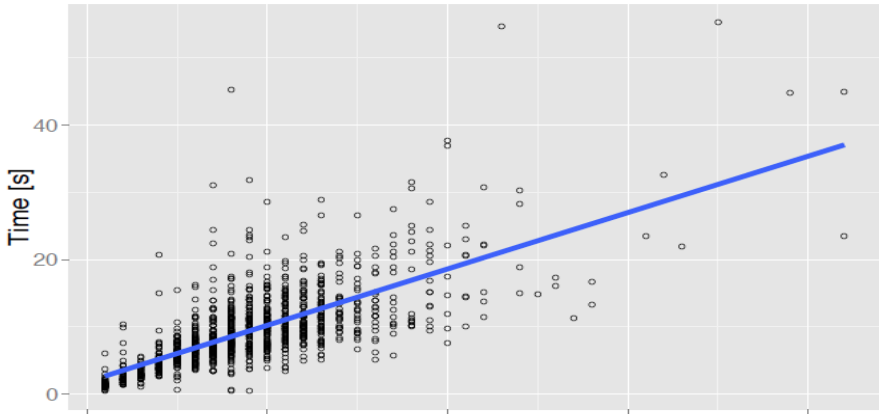


Figura 3: Tiempo de extracción de metadatos de los documentos en función del número de páginas. Fuente: (Tkaczyk, Szostek, Fedoryszak, Dendek, Bolikowski 2015).

para 1.238 documentos aleatorios. El tiempo medio de procesamiento de este subgrupo fue de 9,4 segundos.

La segmentación de la página y la clasificación de la zona inicial son los pasos que consumen más tiempo. Por defecto, CERMINE procesa la totalidad de los documentos de entrada con el fin de extraer su texto completo. Sin embargo, si el cliente está interesado únicamente en los metadatos del documento (por ejemplo, el título del documento, los autores y afiliaciones), es suficiente con analizar solo la primera y última página del documento, debido a que estos metadatos rara vez están presente en la parte media. En estos casos se recomienda restringir el análisis a un número fijo de páginas, y como resultado, incluso documentos de gran tamaño pueden ser procesados en un tiempo razonable.

1.7 **Tabla resumen de las herramientas estudiadas.**

A continuación, se presenta la Tabla I que resume las características de cada una de las herramientas estudiadas:

Tabla I: Resumen de herramientas.

Herramienta	Entrada	Salida	Licencia	Enfoque	Lenguaje de implementación
Apache Tika	.html, .xml, .doc, .odt, .pdf .txt, otros.	Archivo TXT	Código Abierto	No está disponible en	Java

				la bibliografía consultada	
GROBID	PDF	Archivo XML	Código Abierto	CRF(Campos Aleatorios Condicionales)	Java
LA-PDFText	PDF	Archivo XML	Código Abierto	Aproximación heurística	Java
CERMINE	PDF	Archivo XML	Código Abierto	Máquina de Soporte de Vectores	Java

Al estudiar cada una de las herramientas antes mencionadas y al analizar la tabla resumen mostrada anteriormente e interpretados los resultados obtenidos por la herramienta CERMINE mostrados en la Figura 1, se considera a CERMINE como la herramienta que mejores resultados ofrece comparada con las demás herramientas consultadas en la bibliografía.

1.8 Conclusiones parciales

1. El análisis de los principales conceptos relacionados con la extracción de metadatos y su interrelación, permitió comprender el problema planteado por la investigación.

2. Las variantes para la extracción de metadatos siguiendo un determinado enfoque están directamente ligadas a funciones matemáticas, la herramienta CERMINE usando una máquina de soporte de vectores ofrece resultados con mayor calidad que las demás herramientas consultadas en la bibliografía.

3. Luego de realizado el estudio del estado del arte y analizados los principales enfoques para la extracción de metadatos, se puede afirmar que la eficacia de la extracción de metadatos, aplicando los enfoques antes mencionados, depende en gran medida de la estructura del documento sobre el cual se realizará el proceso de extracción.

CAPÍTULO 2. DESCRIPCIÓN DE LA PROPUESTA

2.1 Introducción

En este capítulo se presenta un componente de extracción de metadatos bibliográficos para la biblioteca digital semántica desarrollada por el grupo de web semántica de la UCI. Se describe en detalle los principales artefactos de ingeniería generados como parte del proceso de diseño e implementación del componente. El componente fue implementado utilizando tecnologías actuales del desarrollo de software.

2.2 Descripción general de la propuesta

El componente propuesto automatiza tres actividades fundamentales, las cuales son:

1. Entrada de datos y carga de documentos

Esta actividad constituye la primera dentro del proceso de extracción y catalogación de los metadatos desde documentos en formato PDF. El sistema le solicita al usuario los datos necesarios para procesar de manera organizada cada una de las colecciones de documentos suministradas a la herramienta para su procesamiento. Los campos que deben ser completados por el usuario para procesar una colección de documentos son:

- **Tipo de colección:**
  En este campo se especifica el tipo de colección al que pertenecen los documentos que van a ser procesados. Las colecciones que soporta el componente son **revista** y **evento**.
- **Nombre de la colección:**
  Este campo se llena automáticamente una vez que el usuario selecciona una de las opciones en el campo anterior (Tipo de colección), es decir si el tipo de colección seleccionado por el usuario fue **Revista** el campo Nombre de la colección contendrá los nombres de todas las revistas que se encuentran registradas en el sistema, de otra forma si el tipo de colección seleccionada por el usuario fue **Evento** entonces el campo Nombre de la colección contendrá los nombres de cada uno de los eventos que ya se encuentren registrados en el sistema. En caso de que la revista o el evento de donde proviene la colección que se desea procesar no se encuentren registrados en el sistema, el usuario debe proceder a su registro en el nomenclador correspondiente, el cual puede ser encontrado en el módulo de configuración.
- **Volumen:**

Este campo se llena automáticamente conteniendo cada uno de los volúmenes asociados a la revista seleccionada por el usuario en el campo Nombre de la colección si la opción que seleccionó el usuario inicialmente en el campo Tipo de colección fue **Revista**. En caso de que la revista seleccionada no contenga el volumen deseado debido a que no se encuentra registrado en el sistema, el usuario debe proceder a su registro en el nomenclador correspondiente, el cual puede ser encontrado en el módulo configuración.

- **Número**

  El campo se llena automáticamente conteniendo cada uno de los números asociados al volumen seleccionado por el usuario, el cual a su vez se encuentra asociado a la revista seleccionada por el usuario en el campo Nombre de la colección si la opción que seleccionó el usuario inicialmente en el campo Tipo de colección fue **Revista**. En caso de que el volumen seleccionado no contenga el número deseado debido a que no se encuentra registrado en el sistema, el usuario debe proceder a su registro en el nomenclador correspondiente, el cual puede ser encontrado en el módulo configuración y posteriormente se realiza el procedimiento antes mencionado.

- **Edición**

  Este campo se llena automáticamente conteniendo cada uno de las ediciones asociadas al evento seleccionado por el usuario si la opción que seleccionada por el mismo en el campo Tipo de colección fue **Evento**. En caso de que el evento seleccionado no contenga la edición deseada debido a que no se encuentra registrado en el sistema, el usuario debe proceder a su registro en el nomenclador correspondiente, el cual puede ser encontrado en el módulo configuración y posteriormente se realiza el procedimiento antes mencionado.

Una vez que se han seleccionado correctamente las opciones deseadas en cada uno de los campos antes mencionados para garantizar que el sistema tenga todos los datos asociados a la colección, se procede a subir la colección de documentos. La colección se puede subir al servidor de la aplicación a través de dos vías, arrastrando los documentos hacia el área señalada o haciendo clic sobre la misma, lo cual mostrará una ventana desde donde se podrá acceder al directorio en el que se encuentran los documentos que se desean procesar.

2. Procesamiento de documentos

Esta actividad es el núcleo del componente, debido a que en ella es donde se realiza todo el procesamiento de los documentos para extraer sus metadatos bibliográficos, que posteriormente serán catalogados. Una vez que el sistema cuenta con todos los datos asociados a la colección, los cuales fueron suministrados durante la actividad de entrada de datos y carga de documentos y cuenta además

con la colección de documentos a procesar, a través de una función hash son renombrados cada uno de los documentos de la colección con su nombre encriptado en MD5. Posteriormente la herramienta CERMINE procesa de manera concurrente cada uno de los documentos contenidos en la colección de PDF y al mismo tiempo que va procesando los documentos almacena los metadatos extraídos en una tabla temporal de la base de datos. Luego, los documentos son movidos hacia un directorio en el servidor denominada *Procesed*.

3. Catalogación

Esta actividad consiste en que el usuario tenga la posibilidad de verificar si los metadatos que han sido extraídos por la herramienta son correctos o se corresponden con los metadatos del documento fuente y en caso de que existan errores estos puedan ser corregidos. Como se menciona anteriormente, una vez que la herramienta CERMINE comienza a procesar los documentos de la colección, los metadatos extraídos son almacenados en una tabla temporal de la base de datos de la aplicación y se muestra al usuario un listado con los documentos que han sido procesados, de los cuales se puede visualizar su título y un ícono que contiene el enlace hacia la vista de catalogación correspondiente al documento. Una vez que el usuario seleccione el documento que desea catalogar, se mostrará la vista de catalogación de los metadatos en la cual del lado izquierdo de la vista el usuario podrá visualizar el documento en formato PDF y del lado derecho se mostrará un formulario con los metadatos asociados al documento que logró extraer la herramienta CERMINE. En caso de que existan campos en blanco o errores en los metadatos extraídos, el usuario tendrá la opción de editar y corregir los metadatos. Cuando el usuario termine el proceso de catalogación, los metadatos son actualizados y posteriormente son almacenados en sus tablas correspondientes en la base datos. Además, al terminar la catalogación, el documento que ya ha sido procesado es movido hacia el repositorio de documentos procesados de la aplicación, concluyendo así la última actividad del componente.

2.2.1 Módulo Configuración

Para la gestión de cada uno de los documentos en formato PDF fue necesaria la creación de un módulo **Configuración** en el que se definen e implementan un total de nueve nomencladores destinados a gestionar información común y poco variable en el tiempo. A través de este se determinan los datos de uso general, las pautas operativas, minimizando de este modo los errores o aplicación de distintos criterios para una misma operación. Para cada uno de los nomencladores, si en algún proceso del sistema se encuentra alguna opción desactivada o no aparece un dato que debería estar, se debe recurrir a este módulo.

El módulo configuración interviene de forma directa en el proceso de extracción de metadatos. En este se necesitan datos que son referentes al tipo de colección que se desea gestionar, si el tipo de colección es un evento el mismo debe estar registrado en el sistema con sus respectivas ediciones. De igual manera sucede con las revistas. Por cada revista se deben gestionar los volúmenes y los números pertenecientes a la misma. Cada uno de los nomencladores permite la creación de campos de forma dinámica, brindando las opciones de nombrar, cambiar, modificar y asignar campos a cada nomenclador, la asignación de campos se realiza en dependencia de las características del nomenclador que se esté creando, permitiendo que cada grupo tenga solamente la información que le corresponda. A continuación se listan cada uno de los nomencladores implementados para este módulo:

1. Nomenclador Evento.
2. Nomenclador Revista.
3. Nomenclador Número.
4. Nomenclador Volumen.
5. Nomenclador Edición.
6. Nomenclador Editorial.
7. Nomenclador Formato.
8. Nomenclador Tipo de Tesis.
9. Nomenclador Estado de Documento.

En una primera versión de la aplicación solo se admite formato PDF y de igual manera ocurre con el estado del documento, el cual es almacenado en la base de datos con el estado **Catalogado** una vez que ya han sido corregidos los metadatos resultantes del proceso de extracción. Los resultados obtenidos de la aplicación de este componente en la biblioteca digital semántica se describen en el capítulo 3 de esta memoria de tesis.

2.3 Metodología de desarrollo de software

El desarrollo de todo software debe estar guiado por una metodología de desarrollo. De esta depende, en gran medida que el software tenga la calidad requerida. Existen dos grupos de metodologías: ágiles y tradicionales. No existe una metodología universal para cada tipo de proyecto. Se define una metodología según las características del equipo de desarrollo, el dominio de aplicación, tipo de contrato, complejidad y envergadura del proyecto. Debido a la necesidad de desarrollar la propuesta de solución en un breve período de tiempo, garantizando la flexibilidad necesaria en cuanto a la variación de los requisitos, no existiendo un contrato tradicional, siendo el cliente parte del equipo de

desarrollo, permitiendo reducir la generación de documentos y artefactos; se hace necesario optar por un enfoque ágil de desarrollo de software en lugar de un enfoque tradicional, por lo cual se definió la variación de AUP para la UCI en el escenario No.4 correspondiente a Historias de Usuario (HU), como la metodología de desarrollo que más se ajusta a las características del proyecto.

2.4 Entorno de desarrollo

Para la implementación del componente se utilizaron un grupo de herramientas a las cuales se hace referencia a continuación.

2.4.1 IntelliJ IDEA Community Edition 14.0.1

IntelliJ IDEA *Community Edition* es la versión de código abierto de IntelliJ IDEA, un IDE (*Integrated Development Environment*) Premier para Java, Groovy y otros lenguajes de programación[6]. Este es precisamente el IDE utilizado para desarrollar el componente de extracción de metadatos debido a que entre los lenguajes que soporta se encuentra Groovy, que es el lenguaje de programación utilizado para codificar el componente.

2.4.2 Grails 2.5.3

Grails es un *framework* de desarrollo web de gran alcance, para la plataforma Java destinada a multiplicar la productividad de los desarrolladores. Este es el *framework* utilizado para el desarrollo del componente, ofreciendo integración con la JVM (Java Virtual Machine), lo cual le permite ser productivo mientras que proporciona características de gran alcance, incluyendo ORM integrado, idiomas específicos de dominio, meta-programación en tiempo de compilación y programación asíncrona (Ledbrook, Smith 2014)**.**

2.4.3 Groovy 2.5

Groovy[7] es un lenguaje opcionalmente tipado y dinámico, con capacidades de compilación estática para la plataforma Java. Está orientado a mejorar la productividad del desarrollador gracias a una sintaxis concisa, familiar y fácil de aprender. Se integra con cualquier programa escrito en Java, incluyendo las capacidades de *scripting*, de autoría de lenguaje específico de dominio, tiempo de ejecución y tiempo de compilación, meta-programación y la programación funcional. Debido a todas

[6] http://www.jetbrains.org
[7] http://www.groovy-lang.org/

estas posibilidades y capacidades que ofrece además de su integración con el resto de las herramientas utilizadas, es el lenguaje empleado para la codificación de la aplicación.

2.4.4 PostgreSQL 9.4

PostgeSQL[8] es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. PostgreSQL utiliza un modelo cliente/servidor y utiliza multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Este sistema gestor de bases de datos es el utilizado en el componente implementado para gestionar la base de datos utilizada por el mismo.

2.4.5 Sistema de control de versiones Git 2.5.3

Git[9] es un sistema libre de control de versiones y de código abierto distribuido, diseñado para manejar desde pequeños proyectos hasta proyectos muy grandes con rapidez y eficacia. Es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. La función de Git que realmente lo hace destacar es su modelo de ramificación.

Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o *frontend* como **Cogito** o **StGIT**. Sin embargo, Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena. Debido a todas estas posibilidades y capacidades que ofrece además de su integración con el resto de las herramientas utilizadas, es el sistema de control de versiones empleado para la codificación de la aplicación.

2.5 Propuesta de arquitectura del componente

A continuación, en la Figura 4, se presenta la arquitectura del proyecto **Extracción, Publicación y Consumo de metadatos bibliográficos como datos enlazados**, la cual sigue el estilo arquitectónico flujo de datos. Este estilo arquitectónico se aplica cuando los datos de entrada son transformados a través de una serie de componentes en datos de salida (Rogers 2000). El patrón arquitectónico utilizado por el proyecto es tuberías y filtros el cual tiene un grupo de componentes llamados filtros conectados por tuberías que transmiten datos de un componente al siguiente. Cada filtro trabaja independientemente de aquellos componentes que se encuentran en el flujo de entrada o de salida;

[8] http://www.postgresql.org.es/
[9] https://git-scm.com/

está diseñado para recibir entrada de datos de una cierta forma y producir una salida de datos hacia el siguiente filtro de una forma específica. Sin embargo, el filtro no necesita conocer el trabajo de los filtros vecinos (Rogers 2000).

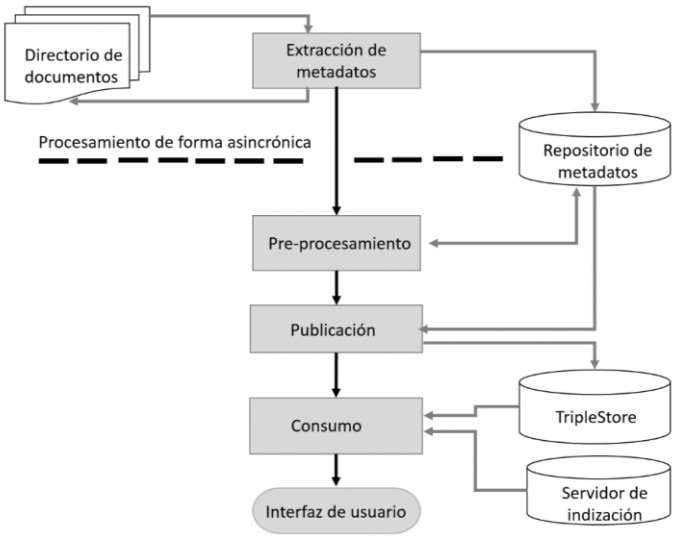


Figura 4: Arquitectura del proyecto Biblioteca Digital Semántica. (Fuente: Elaboración propia)

En la Figura 4 se puede observar que el proyecto cuenta con cuatro componentes fundamentales los cuales son Extracción de metadatos, Pre-procesamiento, Publicación y Consumo. El equipo de desarrollo del presente trabajo de diploma se encarga de la implementación del componente de extracción de metadatos. A continuación en la Figura 5, se puede visualizar la arquitectura definida para el componente.

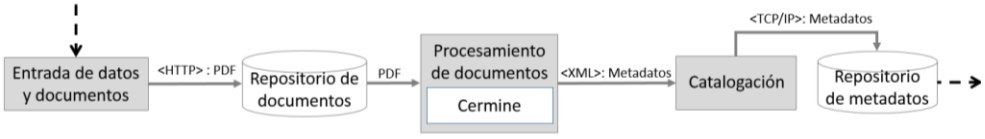


Figura 5: Arquitectura del componente Extracción de metadatos. (Fuente: Elaboración propia)

La propuesta de solución sigue una arquitectura de flujo de datos implementándose el patrón arquitectónico tuberías y filtros. Este es aplicado cuando los datos de entrada, en este caso los documentos en formato PDF, serán transformados en datos de salida mediante una herramienta que permitirá procesarlos para extraer sus metadatos bibliográficos. El proceso comienza cuando el usuario especifica los datos que le son solicitados por el sistema y la ruta para acceder a la colección de documentos en formato PDF que serán procesados. Una vez que ha sido especificada la ruta de acceso a la colección los documentos son descargados hacia el repositorio de documentos del componente.

Una vez que los documentos se encuentran en el repositorio de documentos de la aplicación comienza la actividad de procesamiento de los documentos. El procesamiento de los documentos se realiza utilizando la herramienta **Cermine**, la cual recibe los documentos en formato PDF, los procesa y devuelve sus respectivos metadatos. Luego de tener los metadatos devueltos por la herramienta **Cermine** se almacenan en el repositorio de metadatos y comienza la actividad de catalogación.

Durante la actividad de catalogación se muestran al usuario los metadatos que fueron extraídos de cada documento PDF y el usuario verifica que los metadatos extraídos son correctos y se corresponden con los metadatos contenidos por el PDF seleccionado por el usuario. De ser correctos estos metadatos pasan nuevamente al repositorio de metadatos, si los metadatos tienen algún error el usuario tiene la posibilidad de editarlos y de esta forma los corrige y son enviados al repositorio de metadatos.

2.6 Propuesta de modelo de datos

El modelo de datos de la propuesta de solución está formado por las clases persistentes que se pueden visualizar en la Figura 6, con sus propiedades y relaciones.

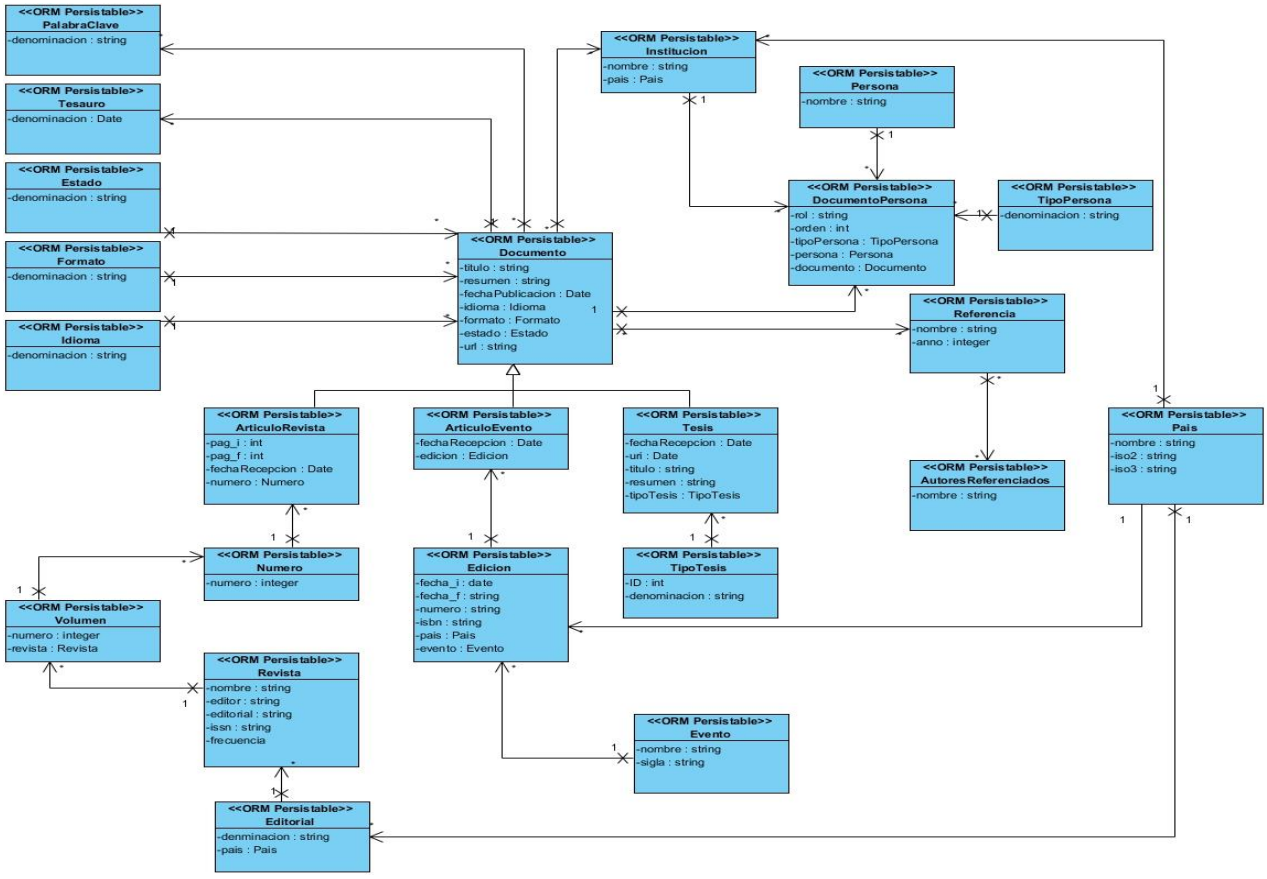


Figura 6: Modelo de datos de la propuesta de solución. (Fuente: Elaboración propia)

2.7 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:

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

2.7.1.1 Nomenclatura según el tipo de clases

Clases Controladoras: las clases que se encuentran dentro de la carpeta **controllers** después del nombre de la clase llevan la palabra: “Controller”.

Ejemplo: Revistacontroller.

Domain (Dominio): Las clases que se encuentran dentro de la carpeta **domain** el nombre que reciben es el de la tabla de la base de datos, pero siguiendo la nomenclatura de UpperCamelCase.

Ejemplo: AutoresReferenciados.

2.7.2 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 CamelCase.

Ejemplo de función: create(). El nombre de este método 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.

Ejemplo de atributo: fechaRecepción. El nombre del atributo está compuesto por dos palabras, la primera en minúsculas y la segunda iniciando con letra mayúscula.

2.7.3 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.

2.8 Captura y validación de requisitos

Entre las técnicas empleadas para la captura y validación de requisitos destacan: los talleres, reuniones con el cliente y el prototipado. El empleo de las mismas permite determinar los requisitos funcionales y no funcionales que seguidamente se detallan.

2.8.1 Requisitos funcionales del software

Una vez aplicadas las técnicas de captura y validación de requisitos se lograron identificar un total de 48 requisitos funcionales, los cuales se listan en la Tabla II.

Tabla II: Requisitos funcionales

No.	Funcionalidades	Prioridad
1	Crear revista	Alta
2	Editar datos de la revista	Alta
3	Listar revistas	Alta
4	Eliminar revista	Alta
5	Mostrar datos de la revista	Alta
6	Crear editorial	Alta
7	Editar datos de la editorial	Alta
8	Listar editoriales	Alta
9	Eliminar editorial	Alta
10	Mostrar datos de la editorial	Alta
11	Crear formato	Media
12	Editar datos del formato	Media
13	Listar formatos	Media
14	Eliminar formato	Media
15	Mostrar datos del formato	Media
16	Crear tipo de tesis	Baja
17	Editar tipo de tesis	Baja
18	Listar tipos de tesis	Baja
19	Eliminar tipo de tesis	Baja
20	Mostrar datos de un tipo de tesis	Baja
21	Crear número de revista	Alta

22	Editar datos del número de revista	Alta
23	Listar números de revistas	Alta
24	Eliminar números de revista	Alta
25	Mostrar datos del número de revista	Alta
26	Crear evento	Alta
27	Editar datos de evento	Alta
28	Listar eventos	Alta
29	Eliminar eventos	Alta
30	Mostrar datos del evento	Alta
31	Crear volumen de revista	Alta
32	Editar datos del volumen de revista	Alta
33	Listar volúmenes de revista	Alta
34	Eliminar volumen de revista	Alta
35	Mostrar datos del volumen de revista	Alta
36	Crear edición de evento	Alta
37	Editar datos de la edición de un evento	Alta
38	Listar ediciones de eventos	Alta
39	Eliminar edición de un evento	Alta
40	Mostrar datos de edición de evento	Alta
41	Crear estado de documento	Baja
42	Editar datos de estado de documento	Baja
43	Listar estados de documento	Baja
44	Eliminar estado de documento	Baja
45	Mostrar datos de estado de documento	Baja
46	Procesar colección de PDF	Alta
47	Mostrar documentos procesados	Alta
48	Catalogar metadatos extraídos	Alta

2.8.2 Requisitos no funcionales de software

Los requerimientos no funcionales son condiciones que debe cumplir un sistema para satisfacer un contrato o una especificación. Están regidos por las necesidades del usuario para resolver un problema o conseguir un beneficio determinado. Se refieren a las propiedades emergentes del sistema como la fiabilidad, el tiempo de respuesta, la capacidad de almacenamiento, la capacidad de los dispositivos de entrada/salida y la representación de datos que se utilizan en las interfaces del sistema. 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, Marciszack 2013).

2.8.2.1 **Software**:

1. En la pc del cliente debe estar instalada la versión 44.0 o superior del navegador web Mozilla Firefox.

2. En el servidor donde se va a desplegar la aplicación debe estar instalada la versión 1.7 de la Máquina Virtual de Java (JVM).
3. En el servidor donde se va a desplegar la aplicación debe estar instalada la versión 7.9 de Tomcat Server.

2.8.2.2 Hardware:

4. El servidor donde se va a desplegar la aplicación debe tener las siguientes propiedades:
   Microprocesador: Intel Core i5-2380P CPU @ 3.10Ghz
   Memoria RAM: 4 GB DDR3.
   Capacidad de almacenamiento (HDD): 1 Tera Byte (TB).
   Tarjeta de red: Fast-Ethernet 100 MB/s.

2.8.2.3 Diseño:

5. La apariencia de las vistas debe ser de color negro, azul y blanco.
6. El color de los botones debe ser azul para el botón **Listo**, rojo para el botón **Eliminar**, verde para el botón **Aceptar** y gris para el botón **Cancelar**.
7. El tipo de letra a utilizar en las interfaces y los mensajes de la aplicación es Open Sans, Sans-Serif.
8. Los colores de letra a utilizar en las interfaces y en los mensajes de la aplicación son el blanco y negro con tamaño de letra 14px.
9. La interfaz debe ser lo más sencilla posible, para que pueda ser manejada por cualquier tipo de usuario.

2.8.2.4 Usabilidad:

10. El tiempo de aprendizaje del sistema por un usuario deberá ser menor a 4 horas.
11. La aplicación web debe poseer un diseño responsable a fin de garantizar la adecuada visualización en múltiples computadores personales, dispositivos tableta y teléfonos inteligentes.
12. El sistema debe poseer interfaces gráficas bien formadas mediante el uso de los componentes de la arquitectura de la información y los patrones de interacción.
13. El sistema debe poder ser usado por cualquier persona que tenga conocimientos básicos de informática y del programa de mejora de software.

2.9 Historias de usuario

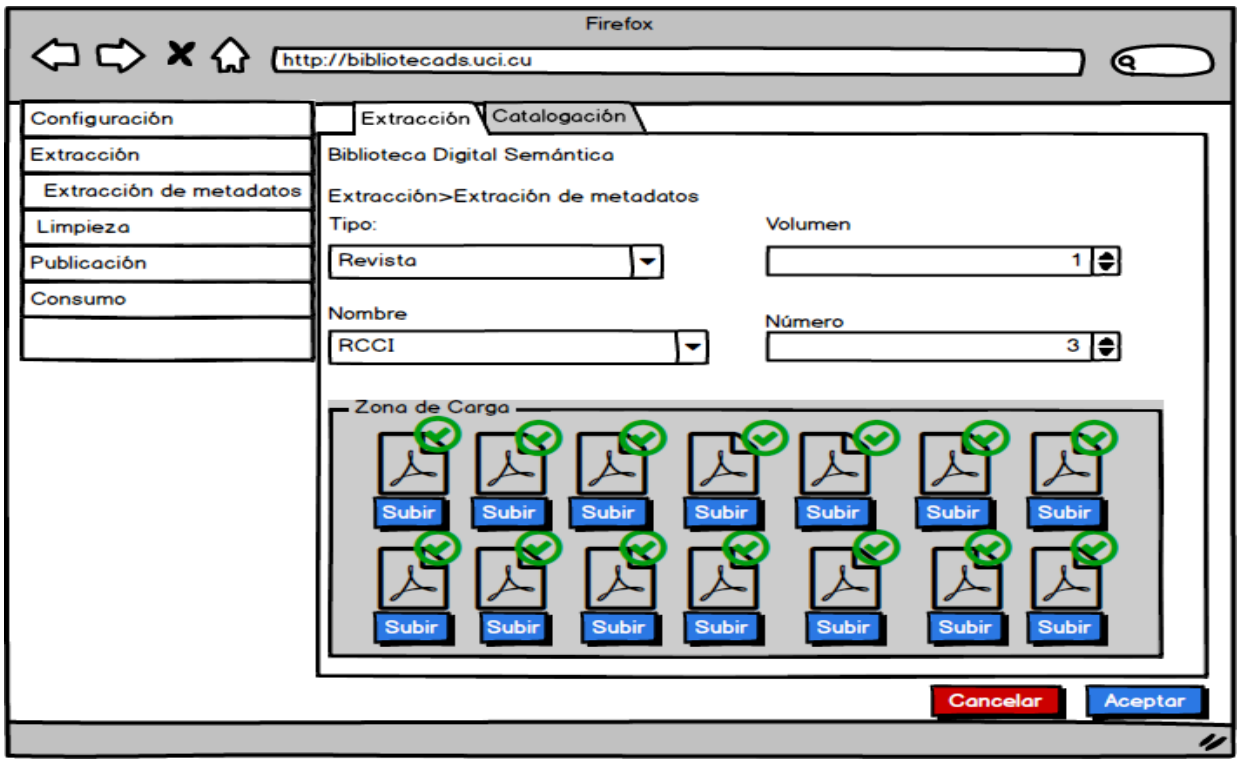
Las historias de usuario son la forma en que se especifican en la variación AUP-UCI los requerimientos funcionales del sistema. Estas se escriben desde la perspectiva del cliente, aunque los desarrolladores pueden brindar también su ayuda en la identificación de las mismas. El contenido de estas debe ser concreto y sencillo. Durante la fase de exploración se realizaron 48 historias de usuario. A continuación, se detallan 3 de ellas, las restantes pueden ser consultadas en el anexo 1.

2.9.1 Procesar colección de PDF

Tabla III: Historia de usuario Procesar colección de PDF

Número: 46	**Nombre del requisito:** Procesar colección de PDF
Programador: Paul Núñez García, Osbel Zorrilla Rivera	**Iteración Asignada:** 2
Prioridad: Alta	**Tiempo Estimado:** 30 días
Riesgo en Desarrollo: Bajo	**Tiempo Real:** 16 días
Descripción: El usuario elige el tipo de colección que desea procesar y luego selecciona el nombre que recibe dicha colección en función del tipo de colección que fue seleccionado anteriormente, entre otros datos que son solicitados por la aplicación en el caso específico del tipo de colección seleccionada, luego el usuario debe cargar en el sistema los PDF a los que desea extraer los metadatos. La acción se puede realizar a un solo PDF o a una colección más amplia de PDF. La aplicación debe extraer los metadatos según el tipo de documento(s) (Evento, Revista, Tesis) que está siendo analizado**.**	
Observaciones:	

Prototipo de interfaz:

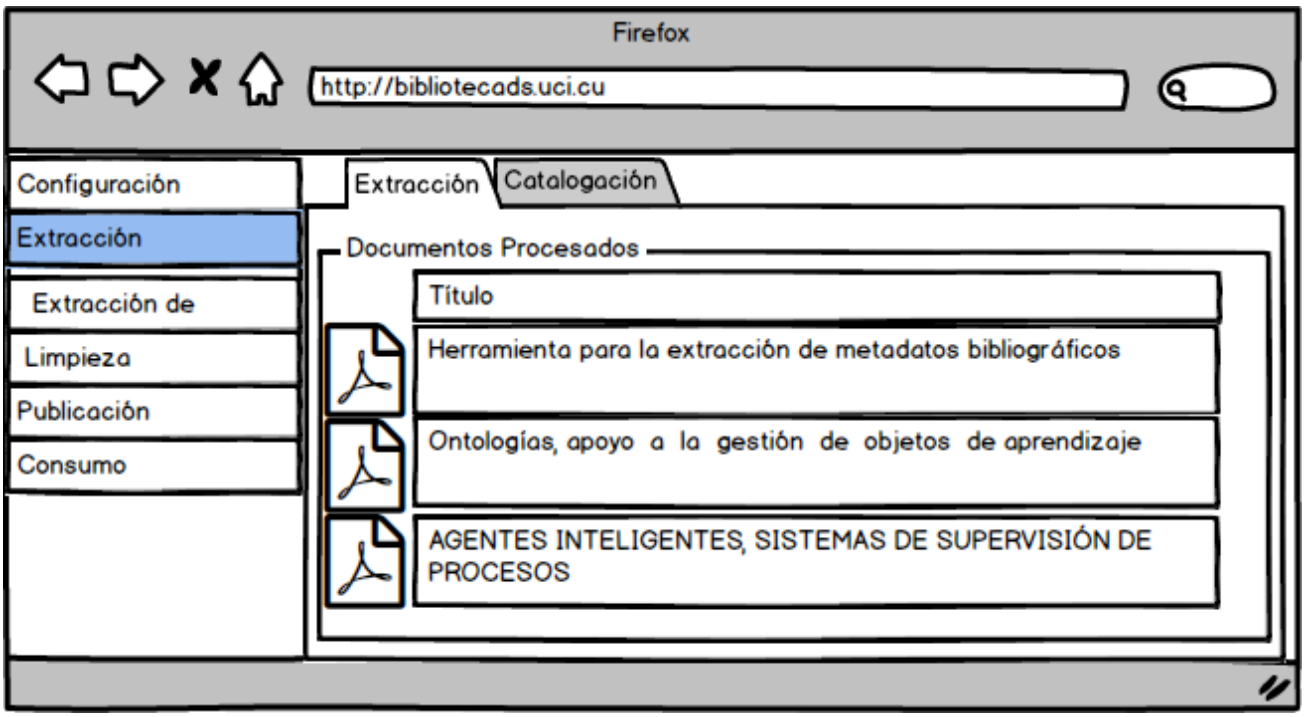


2.9.2 Mostrar documentos procesados

Tabla IV: Historia de usuario Mostrar documentos procesados.

Número: 47	**Nombre del requisito:** Mostrar documentos procesados
Programador: Paul Núñez García, Osbel Zorrilla Rivera	**Iteración Asignada:** 2
Prioridad: Alta	**Tiempo Estimado:** 30
Riesgo en Desarrollo: Bajo	**Tiempo Real:** 16
Descripción: El usuario tiene la opción de verificar los PDF que han sido procesados de la colección seleccionada, en la tabla mostrada en pantalla debe aparecer el título del PDF una vez que este haya sido procesado y un icono haciendo referencia al documento. Al hacer clic sobre el icono se deberá cargar la vista para catalogar los metadatos del documento seleccionado.	
Observaciones:	

Prototipo de interfaz:

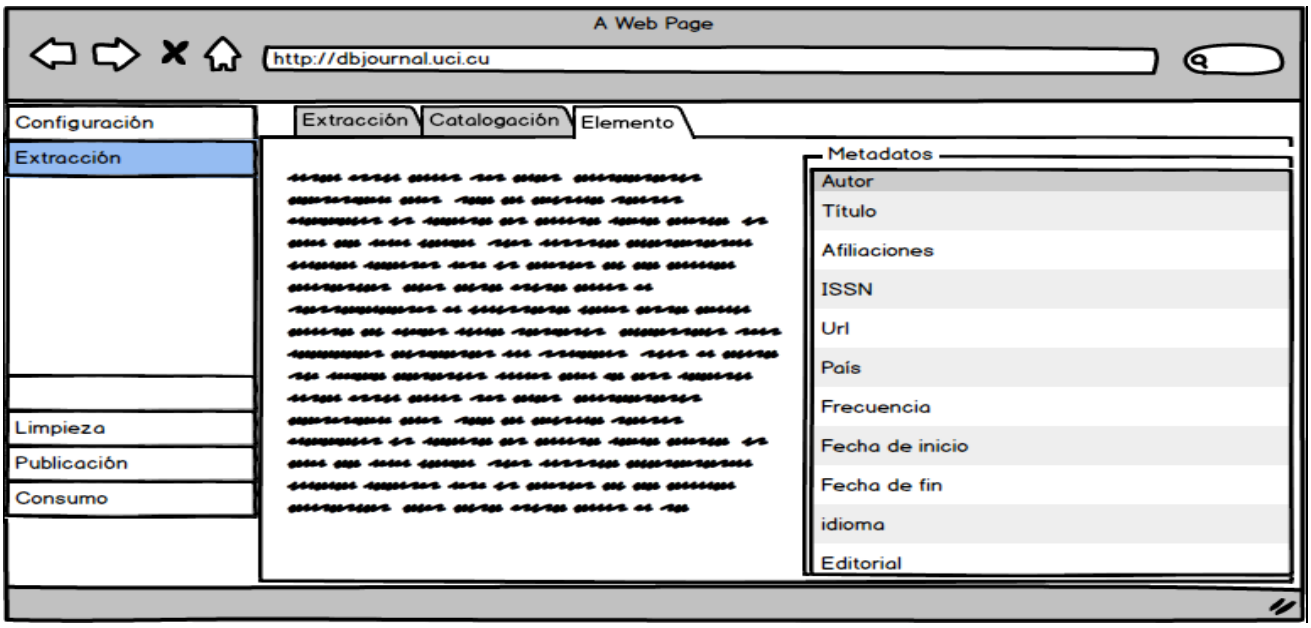


2.9.3 Catalogación de los metadatos extraídos

Tabla V: Historia de usuario Catalogar metadatos extraídos

Número: 48	**Nombre del requisito:** Catalogar metadatos extraídos
Programador: Paul Núñez García, Osbel Zorrilla Rivera	**Iteración Asignada:** 2
Prioridad: Alta	**Tiempo Estimado:** 30
Riesgo en Desarrollo: Bajo	**Tiempo Real:** 16
Descripción: El usuario tiene la opción de verificar a partir de cada uno de los PDF procesados si los metadatos extraídos fueron seleccionados de la forma correcta. En caso de que existan errores en los metadatos extraídos o metadatos incompletos el usuario podrá editar y corregir los errores que fueron detectados.	
Observaciones:	

Prototipo de interfaz:



2.10 Implementación de la clase Hilo y Extracción

Durante el desarrollo de la propuesta de solución, en un primer momento se contempló como forma para la extracción, realizar el proceso de forma secuencial y analizar solo un PDF en un instante de tiempo. Con el objetivo de disminuir el tiempo de procesamiento se utilizaron las bondades de Java para la programación multihilo. Este tipo de programación ocurre cuando varios procesos comparten recursos de procesamiento común (Sadowski, Ball, Bishop, Burckhardt, Gopalakrishnan, Mayo, Musuvathi, Qadeer, Toub 2011). La multitarea se puede observar en aplicaciones donde se puede subdividir operaciones específicas dentro de una única aplicación en hilos individuales. Cada uno de los hilos se pueden ejecutar en paralelo, el sistema operativo divide el tiempo de procesamiento no sólo entre diferentes aplicaciones, sino también entre cada hilo de proceso (Galvin, Gagne, Silberschatz 2013).

La utilización de hilos, proporciona una flexibilidad a la hora de plantearse el desarrollo de aplicaciones, debido a la simplicidad para crear, configurar y ejecutar hilos de ejecución. La utilización de diferentes hilos ejecutándose en paralelo para realizar varias tareas, permite una mejor respuesta a la entrada en tiempo real pues a diferencia de los programas de flujo único que realizan sus tareas ejecutando las sub-tareas secuencialmente, un programa multihilo permite que cada hilo comience y termine tan pronto como sea posible. A partir de este análisis y tomando como punto de partida las ventajas que

ofrece la programación multitarea fueron implementadas dos clases que permitieran este tipo de procesamiento, la clase **Extracción** y la clase **Hilo**.

En la clase **Extracción** se realiza el proceso de captura de datos, se especifica la dirección del repositorio y el tipo de colección a la que pertenecen los PDF que serán analizados. En dependencia del procesador del ordenador en que se esté realizando la extracción, se determina la cantidad de procesos que se pueden realizar de forma concurrente (hilos de procesos), ejecutando una instancia de la herramienta CERMINE por cada hilo. De esta forma se logra aprovechar el procesamiento del ordenador en su mayoría para disminuir el tiempo de extracción. La cantidad de PDF es distribuida entre la cantidad de procesadores para garantizar que cada uno de ellos sea analizado.

En la clase **Hilo** se procede a extraer los metadatos de cada uno de los PDF pertenecientes a la colección especificada por el usuario, en esta clase se redefine el método *run ()* para que en cada una de sus ejecuciones realice una llamada al método extraccion (). El método extraccion () recibe como

```
InputStream inputStream = new FileInputStream(direccion + i);
ContentExtractor extractor = new ContentExtractor();
extractor.setPDF(inputStream);
DocumentMetadata documentMetadata = extractor.getMetadata();
```

Figura 7: Instancia de la herramienta CERMINE. (Fuente: Elaboración propia)

parámetros la dirección del PDF, su nombre, y datos que son necesarios para para su posterior almacenamiento, en este método se instancia la herramienta CERMINE con cada PDF que recibe (ver Figura 7) y se realiza la extracción de cada uno de sus metadatos. Para cada documento en formato PDF se extrae el título, el resumen, palabras claves y sus autores, cada uno de estos datos son enviado a la base de datos para que en una fase posterior sean catalogados.

Debido a la necesidad de llevar a cabo una correcta gestión de los documentos en formato PDF, una vez estos son analizados por la herramienta CERMINE, el método moverpdf () mueve desde el directorio *Preprocesed* hacia el directorio *Procesed* cada uno de los documentos. La implementación de este método pude observarse en la Figura 8.

```
File origen = new File(archivoOrigen + nombre);
InputStream in = new FileInputStream(origen);
File destino = new File(rutaDestino + nombre);
OutputStream out = new FileOutputStream(destino);
byte[] buf = new byte[1024];
int len;
while ((len = in.read(buf)) > 0) {
    out.write(buf, 0, len);
}
out.close();
in.close();
```

Figura 8: Método moverpdf(). (Fuente: Elaboración propia)

2.11 Planificación de las pruebas

El proceso de pruebas de AUP constituye una de sus fortalezas. 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 la variación AUP-UCI este proceso se desagrega en tres disciplinas: Pruebas Internas, de Liberación y Aceptación (Sánchez 2014). En el caso específico del componente implementado, solo se utilizarán las pruebas internas y las pruebas de aceptación.

2.11.1 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. Para ello se realizarán pruebas unitarias las cuales están enfocadas a probar los elementos más pequeños del software. Este tipo de pruebas 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. 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**.

2.11.2 Pruebas de liberación

Las pruebas de liberación están 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. En el caso del componente a implementar este tipo de pruebas no será necesario realizarlas debido a que este constituye un prototipo funcional en su primera versión además de que estas como se mencionaba anteriormente son realizadas por una entidad certificadora externa y el cliente no las considera necesarias.

2.11.3 Pruebas de aceptación

Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido. Las pruebas de aceptación a utilizar serán creadas a partir de las HU. Durante una iteración la HU seleccionada en la planificación de iteraciones se convertirá en una prueba de aceptación. El cliente o usuario especificará los aspectos a probar de cada HU una vez que estas hayan sido implementadas. Cada una de ellas representa una salida esperada del sistema, por lo que para llevar a cabo este proceso se utilizarán pruebas de caja negra, creando para cada HU uno o más casos de prueba en dependencia de las funcionalidades que involucre. Cada caso de prueba debe contener un código, la HU a la que pertenece, el nombre, una breve descripción, la acción a probar, los datos de entrada, los resultados esperados y la evaluación de la prueba.

2.12 Conclusiones parciales

1. La definición y diseño de una arquitectura de software, los estándares de codificación, herramientas y lenguajes empleados durante la implementación de la aplicación resultaron imprescindibles para llegar a una solución viable a los problemas de aplicabilidad y rendimiento.
2. Las Historias de usuario descritas por el cliente con capacidad de evolucionar a medida que avanza el proyecto, potenciaron el desarrollo de la propuesta de solución, definiendo además cada uno de los requerimientos que debe cumplir el sistema.
3. Se definió el nivel de detalle de cada funcionalidad y se realizó una planificación de las iteraciones que permitieron la organización de las HU según la prioridad del cliente y su posterior separación en tareas de ingeniería de acuerdo al tiempo requerido para su implementación.

CAPÍTULO 3. VALIDACIÓN DE LA PROPUESTA

3.1 Introducción

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 (epígrafe 3.4). Se desarrolla un caso de estudio en un contexto real de utilización empleando artículos de revistas y eventos que han sido publicados en nuestro país (epígrafe 3.2). Adicionalmente, se realiza un pre-experimento para evaluar los tiempos de respuesta de la aplicación al realizar las diferentes tareas de extracción de metadatos solicitadas por los usuarios 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 (epígrafe 3.3). Por último, se relacionan aspectos representativos resultantes de la validación de la propuesta de solución (epígrafe 3.5) y se emiten las consideraciones generales de la sección (epígrafe 3.6).

Para una mejor comprensión del contenido de este capítulo 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, Correa, Rojas 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.

3.2 Pruebas de software

El principal objetivo del diseño de casos de prueba es obtener un conjunto de pruebas que tengan la mayor probabilidad de descubrir los defectos del software. Para llevar a cabo este objetivo, se usan dos categorías diferentes de técnicas de diseño de casos de prueba: prueba de caja blanca y prueba de caja negra.

3.2.1 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 extraer los metadatos de cada PDF, el método Extracción () el cual se encuentra contenido en la clase Hilo, ver Figura 9. 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. El uso de esta técnica es mostrado en el ejemplo siguiente:

```
public void Extraccion(String i, String direccion,String url,String tipoColeccion,Integer idColeccion) {
        DataBase metExt = new DataBase();                                   // (1) -Inicio
        try {
            InputStream inputStream = new FileInputStream(direccion + i);   // (2)
            ContentExtractor extractor = new ContentExtractor();
            extractor.setPDF(inputStream);
            DocumentMetadata documentMetadata = extractor.getMetadata();
            String title = documentMetadata.getTitle();
            String resumenPdf = documentMetadata.getAbstrakt();
            boolean flag = false;
            String autoresPdf = "";
            String palabrasClaves = "";
            for (String pClaves : documentMetadata.getKeywords()) {         //(3)
                if (flag) {                                                 //(4)
                    palabrasClaves += ",";                                  //(5)
                }
                flag = true;                                                //(6)
                palabrasClaves += pClaves;
            }
            autoresPdf=obtenerAutores(documentMetadata);
            title=remplazar(title);
            resumenPdf=remplazar(resumenPdf);
            autoresPdf=remplazar(autoresPdf);
            palabrasClaves=remplazar(palabrasClaves);

            String fecha_recepcion;
            SimpleDateFormat feRec=new SimpleDateFormat("dd/MM/yyyy");
            fecha_recepcion=feRec.format(new Date());
            String fecha_publicacion;
            SimpleDateFormat fePubil=new SimpleDateFormat("dd/MM/yyyy");
            fecha_publicacion=fePubil.format(new Date());

            LinkedList<String> parametros = new LinkedList<>();
            parametros.add(title);
            parametros.add(resumenPdf);
            parametros.add(autoresPdf);
            parametros.add(palabrasClaves);
            parametros.add(url);
            parametros.add(i);
            parametros.add(fecha_recepcion);
            parametros.add(fecha_publicacion);
            parametros.add(tipoColeccion);
            metExt.insertar("metadata_extraction", parametros,idColeccion);  // (7)
        } catch (Exception ex) {                                             //(8)
            ex.printStackTrace();                                            //(9)
        }
    }                                                                        //(10) -Fin
```

Figura 9: Funcionalidad encargada de extraer los metadatos de cada PDF. (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 10.

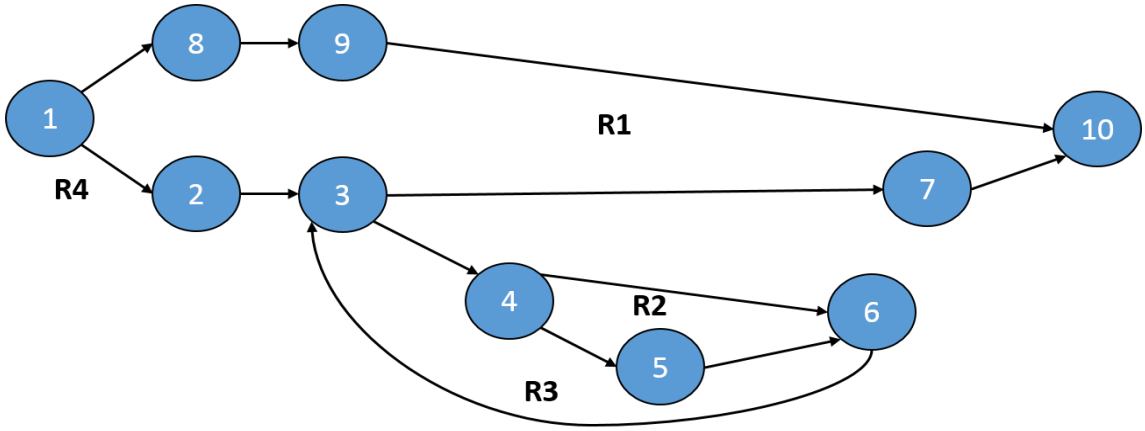


Figura 10: Grafo de flujo asociado al método Extraccion().(Fuente: Elaboración propia)

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:

$V(G) = (a - n) + 2$ $V(G) = p + 1$ $V(G) = r$

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 formas para afirmar un resultado seguro y confiable.

$V(G) = (12 - 10) + 2 = 4$ $V(G) = 3 + 1 = 4$ $V(G) = 4$

Luego se obtiene una complejidad ciclomática que toma valor 4, esta cifra representa la cantidad de caminos independientes que existen en el grafo de flujo construido para el método **Extraccion()**. A continuación, se muestra la Tabla VI con cada uno de los caminos independientes que fueron identificados en el grafo de flujo.

Tabla VI: Caminos independientes identificados en el grafo de flujo del método Extraccion().

Caminos independientes

Camino básico: #1	1-8-9-10
Camino básico: #2	1-2-3-4-6-3-7-10
Camino básico: #3	1-2-3-4-6-3-4-5-6-3-7-10
Camino básico: #4	1-2-3-7-10

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 VII: Caso de prueba para el camino básico #1.

Código: CPCB-01	
Descripción:	Los datos de entrada cumplirán los siguientes requisitos: - Variable dirección debe corresponder a un directorio incorrecto.
Condición de ejecución:	La dirección no debe existir en el directorio.
Entrada:	dirección:"/web-app/Uploads"
Resultados esperados:	No realizar ninguna acción sobre la Base de Datos.
Evaluación de la prueba: Satisfactoria	

Tabla VIII: Caso de prueba para el camino básico #2.

Código: CPCB-02	
Descripción:	Los datos de entrada cumplirán los siguientes requisitos: - El PDF cuyo nombre está contenido en la variable (i) solo contiene una palabra clave en su lista de palabras claves. -El parámetro dirección debe ser una dirección válida de repositorio.
Condición de ejecución:	El PDF debe estar contenido en el directorio que está siendo analizado.
Entrada:	Dirección: "web-app/Uploads/" i: Sample1.pdf
Resultados esperados:	Son insertados los metadatos correspondientes al PDF en la Base de Datos.
Evaluación de la prueba: Satisfactoria	

Tabla IX: Caso de prueba para el camino básico #3.

Código: CPCB-03	
Descripción:	Los datos de entrada cumplirán los siguientes requisitos: - El PDF cuyo nombre está contenido en la variable (i) contiene una lista de más de un elemento, en este caso varias palabras claves.
Condición de ejecución:	El PDF debe estar contenido en el directorio que está siendo analizado.
Entrada:	Dirección: “web-app/Uploads/” i: Sample1.pdf
Resultados esperados:	Son insertados los metadatos correspondientes al PDF en la Base de Datos.
Evaluación de la prueba: Satisfactoria	

Tabla X: Caso de prueba para el camino básico #4.

Código: CPCB-04	
Descripción:	Los datos de entrada cumplirán los siguientes requisitos: - El PDF cuyo nombre está contenido en la variable (i) no contiene palabras claves.
Condición de ejecución:	El PDF debe estar contenido en el directorio que está siendo analizado.
Entrada:	Dirección: “web-app/Uploads/” i: Sample1.pdf
Resultados esperados:	Son insertados los metadatos correspondientes al PDF en la Base de Datos exceptuando las palabras claves.
Evaluación de la prueba: Satisfactoria	

Resultados de las Pruebas Unitarias

El uso del marco de trabajo Groovy JUnit, constituye un entorno para ejecutar pruebas internas en el lenguaje de programación Groovy. Este entorno fue utilizado para realizar las pruebas unitarias. En una primera iteración se realizaron 19 casos de pruebas de ellos resultaron satisfactorios 14, los cuales representan un 74% del total de los casos de prueba realizados aproximadamente, ver Figura 11.

Figura 11: Resultados de las pruebas de caja blanca en la primera iteración.

Una vez corregidos los errores detectados en la primera iteración de las pruebas se realiza una segunda iteración con un total de 19 casos de prueba resultando el 100% de ellos satisfactorios, quedando corregidos todos los errores detectados en la primera iteración, ver Figura 12.

Done: 19 of 19 (in 0,027 s)

Test	Time elapsed	Usage Delta	Usage Before	Usage After	Results
testCreate	0,001 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testDelete	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testEdit	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testEventChanged	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testEventEdicionChanged	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testExtraccion	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testExtraer	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testGenerateName	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testGuardar	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testJournalChanged	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testJournalNumberChanged	0,001 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testJournalVolumenChanged	0,001 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testMovePDf	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testNotFound	0,001 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testSave	0,001 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testShow	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testUpdate	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testUpdateDocuments	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed
testUploadfile	0.0 s	0 Kb	19.323 Kb	19.323 Kb	Passed

Tests Passed: 19 passed
Total time: 0,005 s

Figura 12: Resultados de las pruebas de caja blanca en la segunda iteración. (Fuente: Elaboración propia)

3.2.2 Pruebas de caja negra

Las pruebas de caja negra, también denominadas pruebas de comportamiento, se centran en los requisitos funcionales del software. La prueba de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa (Rogers 2000). La prueba de caja negra no es una alternativa a las técnicas de prueba de caja blanca, más bien se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores que los descubiertos por los métodos de caja blanca. Para la realización de estas pruebas

de caja negra se empleó la técnica Partición de Equivalencia. Esta permite examinar los valores válidos e inválidos de las entradas existentes en el software.

Para realizar las pruebas de aceptación para el componente propuesto se realizaron un total de 30 casos de pruebas. A continuación, se presentan cuatro de los casos de pruebas realizados a las historias de usuario y el resto de los casos de prueba se encuentra en el anexo 2 del presente trabajo.

Tabla XI: Caso de prueba de aceptación CP-01.

Código: CP-01	Historia de Usuario: **HU-46**	
Nombre: Caso de prueba procesar colección de PDF.		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando se agrega una nueva colección para ser procesada.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Seleccionar el tipo de colección a gestionar	Evento o revista	1. Se deben mostrar los nombres de todos los eventos o revistas registrados.
Seleccionar el nombre de la colección a gestionar	COMPUMAT(Evento)	1. Se deben mostrar las ediciones correspondientes al evento seleccionado
Seleccionar el nombre de la colección a gestionar	RCCI(Revista)	1. Se deben mostrar todos los volúmenes pertenecientes a la revista
Seleccionar el número al cual pertenece la colección que será procesada	1	1. Se deben mostrar los números pertenecientes al volumen seleccionado.
Seleccionar la colección que va a ser procesada	Uno o varios PDF	1. Se debe mostrar un mensaje al usuario especificando si hay algún elemento que no se corresponda con el tipo especificado (PDF).
Evaluación de la prueba: Satisfactoria		

Tabla XII: Caso de prueba de aceptación CP-02.

Código: CP-02	Historia de Usuario: **HU-48**

Nombre: Caso de prueba mostrar documentos procesados.		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza una vez que el usuario ha cargado la colección de documentos que desea procesar, estos documentos una vez que son procesados son listados con el título del artículo en la pestaña documentos procesados.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Verificar que los documentos cargados en el sistema y que fueron procesados se listan correctamente por el título.	Cinco documentos en formato PDF para su procesamiento.	1. Se debe mostrar en la tabla de documentos procesados los cinco títulos de los artículos que fueron procesados además de un ícono que contiene el link para visualizar cada documento respectivamente.
Verificar que presionar el botón actualizar la tabla de documentos procesados se actualice correctamente.	Tres documentos en formato PDF para su procesamiento.	1. Se debe mostrar en la tabla de documentos procesados los cinco títulos de los artículos procesados anteriormente más los tres que fueron insertados nuevamente, además de los íconos que contiene el link para visualizar cada documento respectivamente.
Evaluación de la prueba: Satisfactoria		

Tabla XIII: Caso de prueba de aceptación CP-03.

Código: CP-03	Historia de Usuario: **HU-48**
Nombre: Caso de prueba catalogación de los metadatos extraídos.	
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a la catalogación de los metadatos extraídos donde se corrigen los metadatos que no coinciden con la salida esperada.	
Acción a probar:	*Datos de entrada:*
Corregir los metadatos de los PDF pertenecientes a la colección analizada	-Fecha de recepción: 28/5/2016 -Fecha de publicación: 30/5/2016
Actualizar los metadatos de los PDF	Metadatos corregidos: Autor, título, afiliación, etc.

Resultados esperados:
1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos pues la fecha de publicación debe ser menor que la fecha de recepción.
1. Se actualizan los metadatos corregidos. 2. Se deben mostrar un mensaje indicando a usuario que la acción fue realizada con éxito.

pertenecientes a la colección analizada.		3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla XIV: Caso de prueba de aceptación CP-04.

Código: CP-04	Historia de Usuario: **HU-26**	
Nombre: Caso de prueba crear evento.		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a crear un evento.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Inserción del nombre y las siglas del evento.	Nombre: COMPUM@T Siglas: COMPUM@T	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.).
Crear evento	Campos en blanco	1. Se debe mostrar un mensaje al usuario informando que se deben completar todos los campos.
Crear evento	COMPUMAT COMPUMAT	1. Se debe mostrar un mensaje indicando la confirmación de la creación del evento. 2. Se actualizan las migajas de pan
Evaluación de la prueba: Satisfactoria		

Tabla XV: Caso de prueba de aceptación CP-05.

Código: CP-05	Historia de Usuario: **HU-1**	
Nombre: Caso de prueba crear revista.		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar un evento.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*

Inserción de los datos de una nueva revista.		Nombre de la revista: RCCI Frecuencia: $emanal Editor: Yusniel Hidalgo Delgado Editorial: ISSN:588- Fecha inicio:25/5/2016	1 Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.).
Crear revista		Campos en blanco	1. Se debe mostrar un mensaje al usuario informando que se deben completar todos los campos.
Crear revista	RCCI Semanal Yusniel Hidalgo Delgado	1. Se debe mostrar un mensaje indicando la confirmación de la creación del evento. 2. Se actualizan las migajas de pan	
Evaluación de la prueba: Satisfactoria			

Resultados

Se realizaron un total de 30 casos de prueba de aceptación (caja negra), de los cuales 4 resultaron no satisfactorios, representando el 13 % de total de los casos de prueba, ver Figura 13.

Figura 13: Resultados de las pruebas de aceptación.

Durante la realización de las pruebas de caja negra, se detectaron 4 casos de prueba que resultaron no satisfactorios, asociados a estos 4 casos de prueba que resultaron no satisfactorios se identificaron

5 no conformidades debido a funciones incorrectas, 13 no conformidades resultaron ser de interfaz, 1 no conformidad debido a un error de acceso a la base de datos y 1 no conformidad de rendimiento para un total de 20 no conformidades. Cada error encontrado durante las pruebas realizadas fue debidamente mitigado.

3.3 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 1660 PDF los cuales están almacenados *a priori* en un directorio local y posteriormente estos son incorporados al servidor de la aplicación para ser procesados, siendo provenientes de las memorias del evento Informática 2013. 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) a 2.33GHz
- Memoria del sistema: 4GB RAM DDR3 SDRAM

Se proponen los siguientes escenarios para la evaluación:

1. Realizar la extracción de los metadatos bibliográficos de los artículos en formato PDF sin el empleo de la propuesta de solución, es decir de manera manual.

2. Realizar la extracción de los metadatos bibliográficos de los artículos en formato PDF utilizando la propuesta de solución como estímulo.

3.4 Diseño experimental

De acuerdo a la clasificación de los experimentos mostrada al inicio del capítulo, se emplea en la investigación 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 cinco tareas a realizar, enumeradas seguidamente:

1. Procesar 10 documentos en formato PDF.

2. Procesar 50 documentos en formato PDF.

3. Procesar 100 documentos en formato PDF.

4. Procesar 500 documentos en formato PDF.

5. Procesar 1000 documentos en formato PDF.

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

Tabla XVI: Diseño experimental propuesto.

Fuente de datos	Tareas	Observación simple	Estímulo	Observaciónn con estímulo
G	T_1	OS_1	E	OE_1
	T_2	OS_2		OE_2
	T_3	OS_3		OE_3
	T_4	OS_4		OE_4

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:** Tratamiento o estímulo. En este caso la aplicación del componente propuesto.
- **OE_i:** Observación realizada tras aplicar E. El indicador es el tiempo en segundos que tarda la herramienta en extraer los metadatos.

Según (Tkaczyk, Tarnawski, Bolikowski 2015) la herramienta CERMINE en una prueba realizada con un total de 1238 documento en formato PDF logra un tiempo promedio de 9.4 segundos por cada PDF procesado. Una vez que fueron implementadas las clases **Hilo** y **Extracción** con el objetivo de mejorar el tiempo de procesamiento de la herramienta se realizó la misma prueba con un total de 1238 documentos seleccionados de manera aleatoria, logrando resultados satisfactorios disminuyendo el tiempo promedio de procesamiento de los documentos de 9.4 segundos a 4.04 segundos por PDF, casi a la mitad del tiempo. A continuación en la Tabla XVII, se describen de manera detallada los resultados obtenidos con la implementación de estas clases.

*Tabla XVII: Resultados de la implementación de la clase **Hilo** y **Extracción***

No.	Cantidad de PDF	Tiempo Parcial	Tiempo Acumulado	Tiempo Promedio
1	100	00:07:35:35	00:07:35:35	4.55

2	200	00:06:34:21	00:14:09:56	3.94
3	300	00:06:13:31	00:20:22:88	3.73
4	400	00:07:15:79	00:27:38:67	4.35
5	500	00:06:53:84	00:34:32:52	4.13
6	600	00:06:25:74	00:40:58:26	3.85
7	700	00:06:05:43	00:47:03:70	3.65
8	800	00:07:04:22	00:54:07:93	4.24
9	900	00:06:41:80	01:00:49:73	4.01
10	1000	00:06:28:43	01:07:18:17	3.88
11	1100	00:06:42:21	01:14:00:38	4.02
12	1200	00:07:10:29	01:21:10:67	4.30
13	1238	00:02:15:37	01:23:25:04	1.35

Cantidad de documentos PDF procesados: **1238**.

Tiempo total de procesamiento: **01:23:25:04**.

Tiempo promedio por PDF: **4.04 segundos**.

En la tabla anterior se pueden visualizar los resultados obtenidos con la implementación de las clases Hilo y Extraccion, en la última columna se pueden observar los tiempos medios por cada 100 documentos analizados, una vez obtenidos estos resultados se calculó el tiempo medio total, obteniéndose un valor de 4.04 segundos por PDF, en la Figura 14 mostrada a continuación se encuentran representados cada uno de los tiempos medios por cada 100 documentos analizados y la recta que denota el tiempo medio por cada documento analizado.

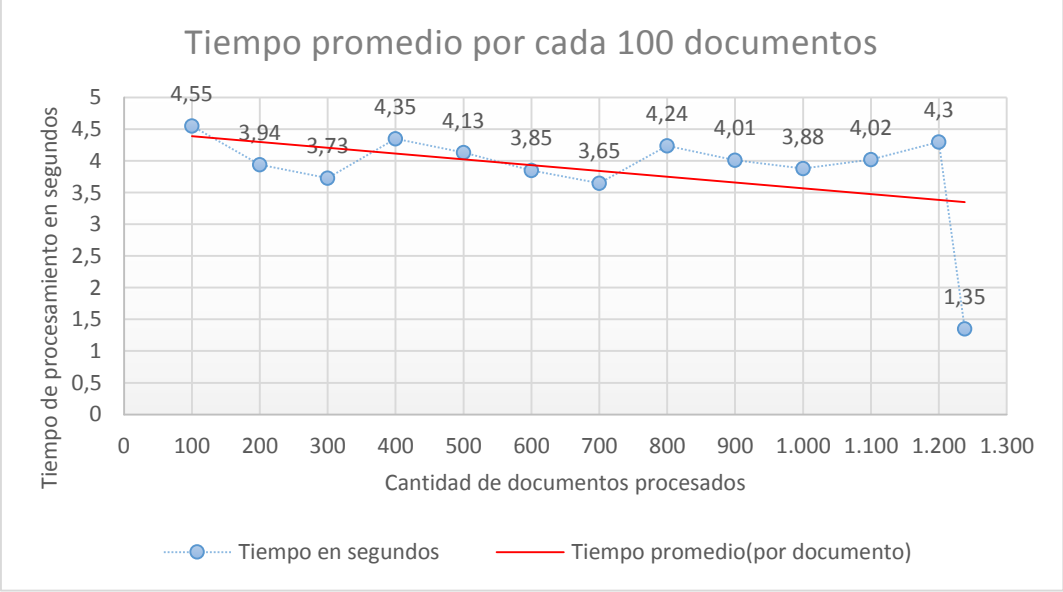


Figura 14: Tiempo promedio por cada 100 documentos analizados.

3.5 Análisis de resultados

Tabla XVIII: Análisis de resultados del experimento.

Fuente de Datos	Tareas	Observación Simple	Estímulo	Observación Estímulo
1660 PDF	T_1: Procesar 10 PDF	OS_1: 00:25:03:33	**CEM**	OE_1: 00:06:09:49
	T_2: Procesar 50 PDF	OS_2: 02:05:16:24		OE_2: 00:30:10:23
	T_3: Procesar 100 PDF	OS_3: 04:17:22:49		OE_3: 00:58:04:86
	T_4: Procesar 500 PDF	OS_4: 21:26:11:31		OE_4: 04:07:25:92
	T_5: Procesar 1000 PDF	OS_5: 42:12:22:18		OE_5: 09:53:14:75

La Tabla XVIII corresponde a la aplicación de la propuesta de solución como estímulo en el segundo 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 en este caso el componente para la extracción de metadatos al mismo grupo de tareas. Es decir, que al aplicar el estímulo se puede apreciar una reducción considerable de los tiempos que toma extracción de metadatos desde los documentos en formato PDF, quedando validada de esta forma la propuesta de solución. En la Figura 15 se reflejan más claramente los resultados obtenidos con la realización del experimento.

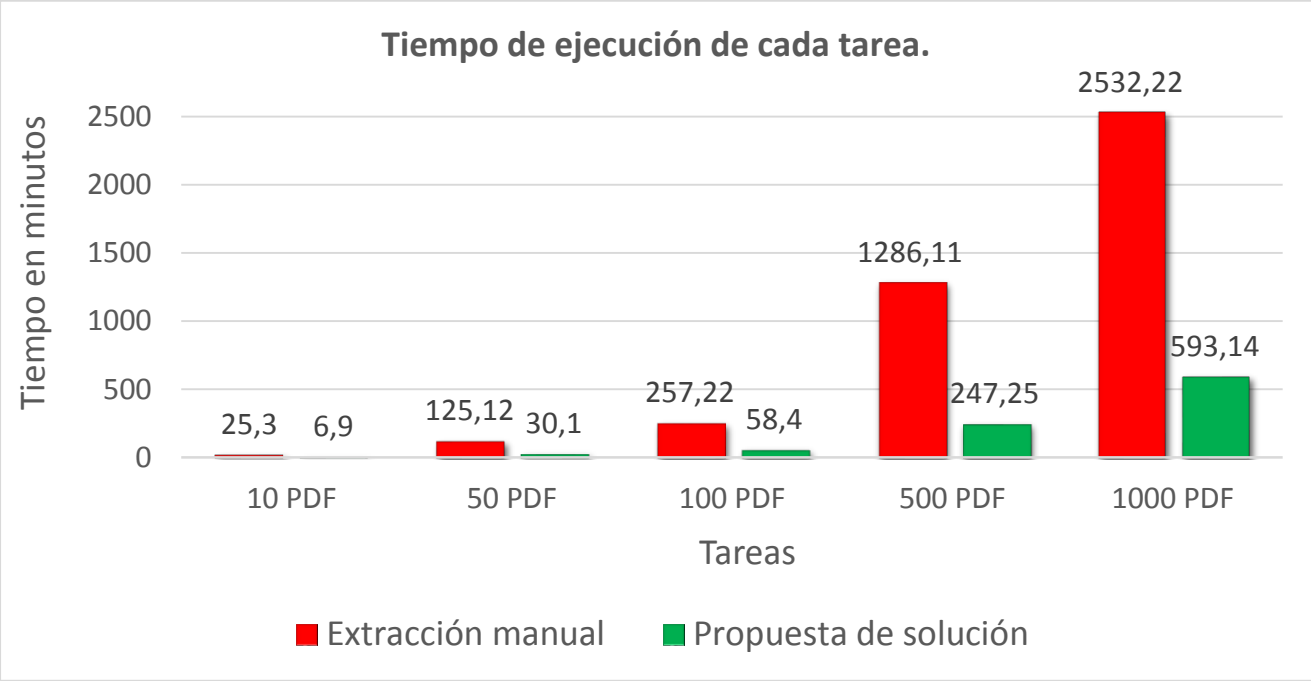


Figura 15: Gráfica comparativa de los resultados obtenidos al realizar el experimento. (Fuente: Elaboración propia)

3.6 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:

1. El empleo de la programación multihilo en la propuesta de solución mejoró los tiempos empleados por CERMINE en el proceso de extracción de metadatos bibliográficos desde múltiples archivos PDF.

2. Dada la relación entre el tiempo de realización de la extracción y el procesamiento del equipo, donde se ejecuta la extracción se puede afirmar que un aumento del segundo (procesamiento) implica una reducción del primero (tiempo).

3. El análisis del experimento aplicado demostró que la aplicación de un componente para la extracción de metadatos permite reducir el tiempo utilizado en este proceso por parte de los especialistas en bibliotecología.

CONCLUSIONES GENERALES

Atendiendo a los objetivos propuestos con esta investigación, se concluye que:

1. La revisión de la literatura evidenció que las aproximaciones existentes para la extracción de metadatos de documentos en formato PDF difieren en cuanto a los enfoques utilizados y los formatos en los que son almacenados los metadatos una vez extraídos.
2. A diferencia de las aproximaciones revisadas, la solución propone un método de extracción basado en programación multihilos. Su implementación permite al sistema operativo dividir el tiempo de procesamiento y reducir el tiempo de extracción.
3. El tiempo que demora la herramienta depende en gran medida de la capacidad de proceso del equipo, lo anterior se corroboró a través de un estudio de relación entre tiempo y procesamiento evidenciándose que la relación entre ellas es directa y significativa.

RECOMENDACIONES

1. La herramienta CERMINE integrada a la solución propuesta extrae metadatos de artículos científicos en formato PDF escritos en diferentes idiomas. En el caso de la extracción de metadatos de artículos científicos en idioma español CERMINE no alcanza el 100% de exactitud durante la extracción de los metadatos para todos los documentos debido a que no se ha entrenado con documentos en este idioma. Para corregir este problema se recomienda realizar el entrenamiento de la herramienta a partir de una colección de documentos científicos en idioma español.
2. Actualmente el componente implementado solo está diseñado para procesar artículos científicos publicados en revistas y eventos. Se recomienda extender las funcionalidades del componente para extraer metadatos de otros documentos científicos tales como Libros y Tesis.

REFERENCIAS BIBLIOGRÁFICAS

ADOBE ACROBAT DC, 2015, Archivos PDF, formato de documento portátil de Adobe | Adobe Acrobat DC. [online]. 2015. [Accessed 3 February 2016]. Available from: https://acrobat.adobe.com/es/es/products/about-adobe-pdf.html
00000

BATISTA, Gustavo EAPA, DELGADO, Myriam and BERNARDINI, Flávia, 2015, ENIAC 2013 Special Issue. *Journal of Intelligent & Robotic Systems*. 2015. Vol. 80, no. 1, p. 225–226.
00000

BERGMAN, RA, AFIFI, AK and MIYAUCHI, M, 2014, A digital library of anatomy information. Illustrated Encyclopedia of Human Anatomic Variation: Opus I: Muscular System: Alphabetical Listing of Muscles: Peroneus tertius muscle. *Last accessed on*. 2014. Vol. 21, no. 05.
00004

BUCKLAND, Michael, 2012, What kind of science can information science be? *Journal of the American Society for Information Science and Technology*. 2012. Vol. 63, no. 1, p. 1–7.
00072

BUTTON, Didy, HARRINGTON, Ann and BELAN, Ingrid, 2014, E-learning & information communication technology (ICT) in nursing education: A review of the literature. *Nurse education today*. 2014. Vol. 34, no. 10, p. 1311–1323.
00048

CHOUDHURY, Sagnik Ray, MITRA, Pinaki, KIRK, Andi, SZEP, Silvia, PELLEGRINO, Donald, JONES, Simon and GILES, C Lee, 2013, Figure metadata extraction from digital documents. In : *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*. IEEE. 2013. p. 135–139.
00018

GALVIN, Peter B, GAGNE, Greg and SILBERSCHATZ, Abraham, 2013, *Operating system concepts*. John Wiley & Sons, Inc.
03955

GARCÍA, Ariel Rodríguez, 2013, El aprovechamiento de los metadatos en las bibliotecas. *e-Ciencias de la Información*. 2013. Vol. 0, no. 0, p. 1–13.
00000

GRANITZER, Michael, HRISTAKEVA, Maya, KNIGHT, Robert, JACK, Kris and KERN, Roman, 2012, A Comparison of Layout Based Bibliographic Metadata Extraction Techniques. In : *Proceedings of the 2Nd International Conference on Web Intelligence, Mining and Semantics* [online]. New York, NY, USA : ACM. 2012. p. 19:1–19:8. WIMS '12. ISBN 978-1-4503-0915-8. Available from: http://doi.acm.org/10.1145/2254129.2254154
00012

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

GRIFFIN, Stephen M, 1998, Taking the Initiative for Digital Libraries. *Electronic Library*. 1998. Vol. 16, no. 1, p. 24–27.
00034

GRIFFIN, Stephen M., 1999, Digital Libraries Initiative – Phase 2: Fiscal Year 1999 Awards. *Bulletin of the American Society for Information Science and Technology*. 1999. Vol. 26, no. 1, p. 14–21. DOI 10.1002/bult.137.
00012

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.
00007

HERNÁNDEZ, Francisca and AGENJO, Xavier, 2010, Tendencias internacionales en el desarrollo funcional de la recuperación de la información: Linked Open Data (LOD). In : *X Workshop Rebiun sobre proyectos digitales: diez años de proyectos digitales: cambian las bibliotecas, cambian los profesionales. Valencia, 7 y 8 de octubre de 2010*. 2010.
00004

INITIATIVE, Dublin Core Metadata and OTHERS, 2014, Dublin core metadata initiative. . 2014.
00004

JONES, Brian M, SUNDERLAND, E Mark, SAWICKI, Marcin, LITTLE, Robert A and DAVIS, Tristan A, 2015, *Method and apparatus for utilizing an extensible markup language schema for managing specific types of content in an electronic document*. Google Patents.
00000 US Patent App. 14/797,274

KENNEDY, Graeme, 2014, *An introduction to corpus linguistics*. Routledge.
01468

LAFFERTY, John, MCCALLUM, Andrew and PEREIRA, Fernando CN, 2001, Conditional random fields: Probabilistic models for segmenting and labeling sequence data. . 2001.
08614

LEDBROOK, Peter and SMITH, Glen, 2014, *Grails in Action*. Manning Publications Co.
00023

LOPEZ, Patrice and ROMARY, Laurent, 2015, GROBID-Information Extraction from Scientific Publications. *ERCIM News*. 2015. Vol. 2015, no. 100.
00000

LÖSCH, Uta, BLOEHDORN, Stephan and RETTINGER, Achim, 2012, Graph kernels for RDF data. In : *The Semantic Web: Research and Applications*. Springer. p. 134–148.
00053

MAJI, Subhransu, 2015, Hidden Markov Models. . 2015.
00000

MARTÍNEZ, Luis Fernando Cano, 2014, *Modelos ocultos de Markov: Trabajo Fin de Grado*. Universidad de Sevilla.
00000

MATTMANN, Chris and ZITTING, Jukka, 2011, *Tika in action*. Manning Publications Co.

MEYER, David and WIEN, FH Technikum, 2015, Support vector machines. *The Interface to libsvm in package e1071*. 2015.
00274

MISSIER, Paolo, BELHAJJAME, Khalid and CHENEY, James, 2013, The W3C PROV family of specifications for modelling provenance metadata. In : *Proceedings of the 16th International Conference on Extending Database Technology*. ACM. 2013. p. 773–776.
00031

MORENO, Juan Carlos and 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.
00002

OHTA, Masaya, ARAUCHI, Daiki, TAKASU, Atsuhiro and ADACHI, Jun, 2014, Empirical evaluation of CRF-based bibliography extraction from reference strings. In : *Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on*. IEEE. 2014. p. 287–292.
00003

RAMAKRISHNAN, Cartic, PATNIA, Abhishek, HOVY, Eduard and BURNS, Gully APC, 2012a, Layout-aware text extraction from full-text PDF of scientific articles. *Source Code for Biology and Medicine*. 28 May 2012. Vol. 7, no. 1, p. 1–10. DOI 10.1186/1751-0473-7-7.
00026

RAMAKRISHNAN, Cartic, PATNIA, Abhishek, HOVY, Eduard and BURNS, Gully APC, 2012b, Layout-aware text extraction from full-text PDF of scientific articles. *Source code for biology and medicine*. 2012. Vol. 7, no. 1, p. 1.
00032

RENDÓN-MIRANDA, Juan C, ARANA-LLANES, Julia Y, GONZÁLEZ-SERNA, Juan G and GONZÁLEZ-FRANCO, Nimrod, 2014, Automatic classification of scientific papers in PDF for populating ontologies. In : *Proceedings of the 2014 International Conference on Computational Science and Computational Intelligence-Volume 02*. IEEE Computer Society. 2014. p. 319–320.
00002

ROGERS, Pressman, 2000, *Ingeniería de Software Quinta Edición (un enfoque práctico), McGrawHill*. Interamericana de España.
00001

SADOWSKI, Caitlin, BALL, Thomas, BISHOP, Judith, BURCKHARDT, Sebastian, GOPALAKRISHNAN, Ganesh, MAYO, Joseph, MUSUVATHI, Madanlal, QADEER, Shaz and TOUB, Stephen, 2011, Practical parallel and concurrent programming. In : *Proceedings of the 42nd ACM technical symposium on Computer science education*. ACM. 2011. p. 189–194.
00012

SÁNCHEZ, A., 2006, *Gran diccionario de uso del español actual* [online]. Sociedad General Española de Librería. ISBN 978-84-9778-224-1. Available from: https://books.google.com.cu/books?id=xE0GAQAACAAJ
00091

SÁNCHEZ, Tamara Rodríguez, 2014, *Metodología de desarrollo para la actividad productiva de la UCI*. 2014.
00000

Scimago Journal & Country Rank, 2013. [online], [Accessed 3 February 2016]. Available from: http://www.scimagojr.com/

SENSO, José A. and ANTONIO DE LA ROSA PIÑERO, 2003, El concepto de metadato: algo más que descripción de recursos electrónicos. *CiÃ\textordfemeninencia da InformaÃ\SÃ\poundso*. 2003. Vol. 32, p. 95 – 106.
00000

SICILIA, Miguel-Angel, 2014a, *Handbook of metadata, semantics and ontologies*. World Scientific.
00005

SICILIA, Miguel-Angel, 2014b, *Handbook of metadata, semantics and ontologies*. World Scientific.
00005

SINGH, T. and SHARMA, A., 2015, Research work and changing dimensions of digital library: A review. In : *Emerging Trends and Technologies in Libraries and Information Services (ETTLIS), 2015 4th International Symposium on*. January 2015. p. 39–42.
00001

TESTA, Patricia and CERIOTTO, Paula, 2011, Descripción de objetos digitales: metadatos. *Encuentro Nacional de Catalogadores (2do: 2009: Buenos Aires, Argentina)*. 2011. Vol. 1, p. 105–112.
00002

TKACZYK, Dominika, SZOSTEK, Paweł, FEDORYSZAK, Mateusz, DENDEK, Piotr Jan and BOLIKOWSKI, Łukasz, 2015, CERMINE: automatic extraction of structured metadata from scientific literature. *International Journal on Document Analysis and Recognition (IJDAR)*. 3 July 2015. Vol. 18, no. 4, p. 317–335. DOI 10.1007/s10032-015-0249-8.
00003

TKACZYK, Dominika, TARNAWSKI, Bartosz and BOLIKOWSKI, Lukasz, 2015, Structured Affiliations Extraction from Scientific Literature. *D-Lib Magazine*. 2015. Vol. 21, no. 11, p. 7.
00002

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.
00000

ZHANG, Fuzhi and ZHAO, Zihao, 2013, A Metadata Extraction Approach from Papers Based on Meta-learning⋆. . 2013.
00000

ANEXOS

1 Historias de Usuario

Tabla XIX: Historia de usuario Crear revista.

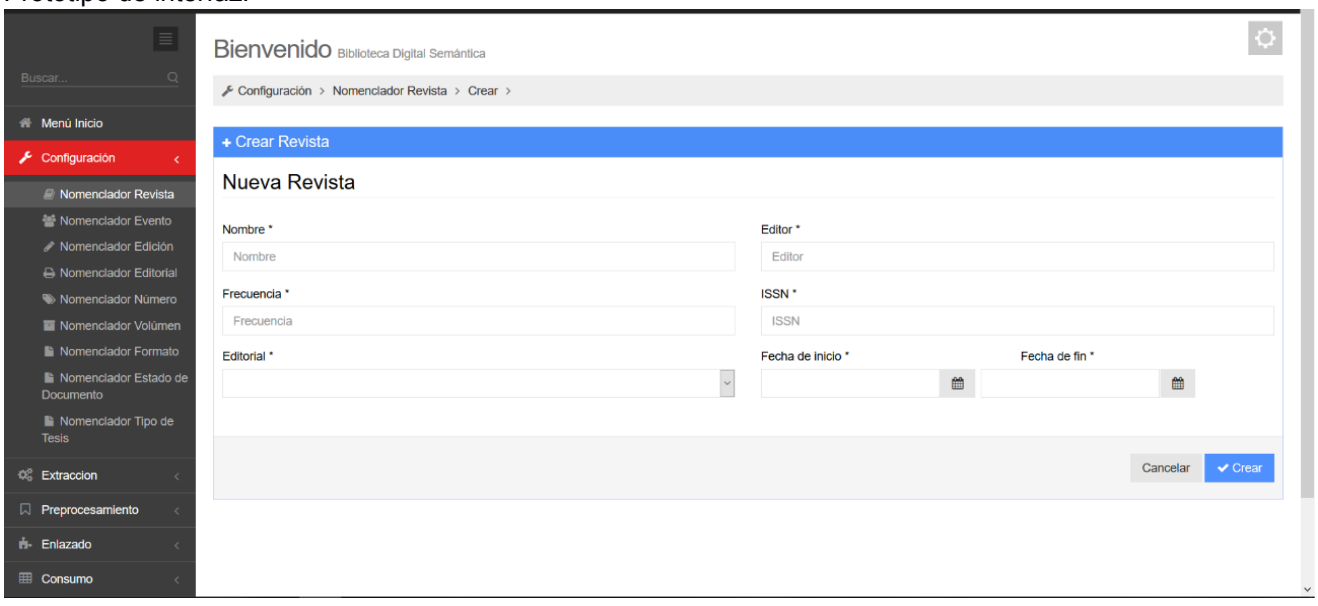
Número: 1	Nombre del requisito: Crear revista
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El sistema debe permitir al usuario crear una nueva revista introduciendo en la aplicación los datos que le son solicitados, tales como: nombre de la revista, la frecuencia, editor, la editorial, el ISSN, la fecha en que comenzó a ser lanzada la revista y la fecha en que termina de ser lanzada la revista	
Observaciones:	
Prototipo de interfaz: 	

Tabla XX: Historia de usuario Editar revista.

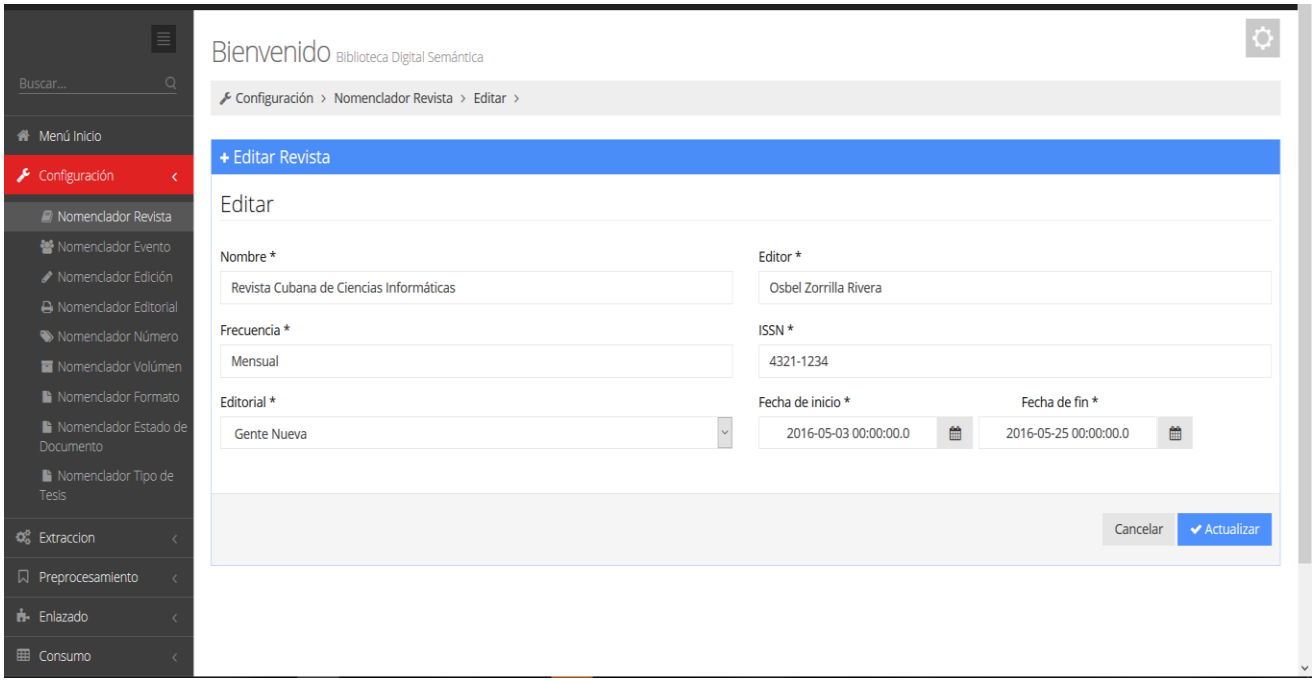
Número: 2	Nombre del requisito: Editar revista
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Editar dentro de las opciones que aparecen al final de la tupla de cada revista listada, la cual al hacer clic sobre ella permitirá editar los datos de la revista correspondiente a la tupla seleccionada.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XXI: Historia de usuario Listar revistas.

Número: 3	Nombre del requisito: Listar revistas
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días

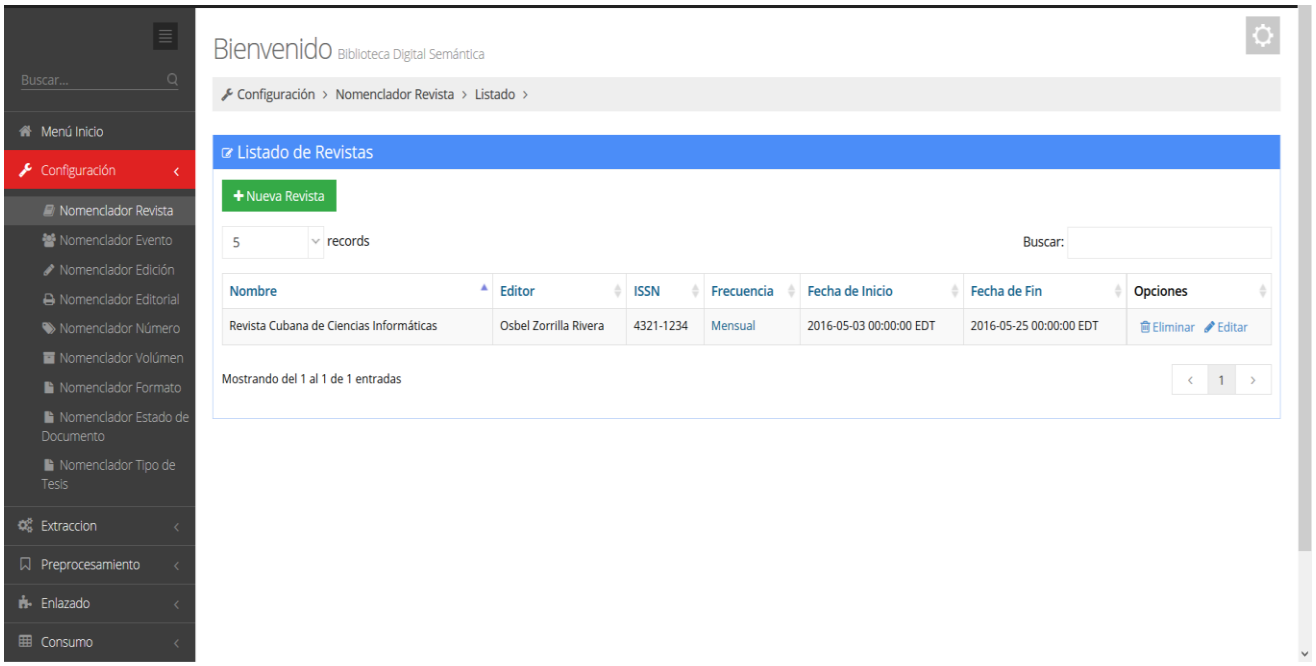
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador Revista se debe mostrar la lista de revistas que se encuentran registradas en el sistema con sus datos correspondientes, tales como: nombre de la revista, la frecuencia, editor, la editorial, el ISSN, la fecha en que comenzó a ser lanzada la revista y la fecha en que termina de ser lanzada la revista	
Observaciones:	
Prototipo de interfaz: 	

Tabla XXII: Historia de usuario Eliminar revista.

Número: 4	Nombre del requisito: Eliminar revista
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día

Descripción: En la pantalla principal del nomenclador se mostrará la opción Eliminar dentro de las opciones que aparecen al final de la tupla de cada revista listada, la cual al hacer clic sobre ella eliminará del sistema la revista correspondiente a la tupla seleccionada.
Observaciones:
Prototipo de interfaz: No aplica.

Tabla XXIII: Historia de usuario Mostrar datos de la revista.

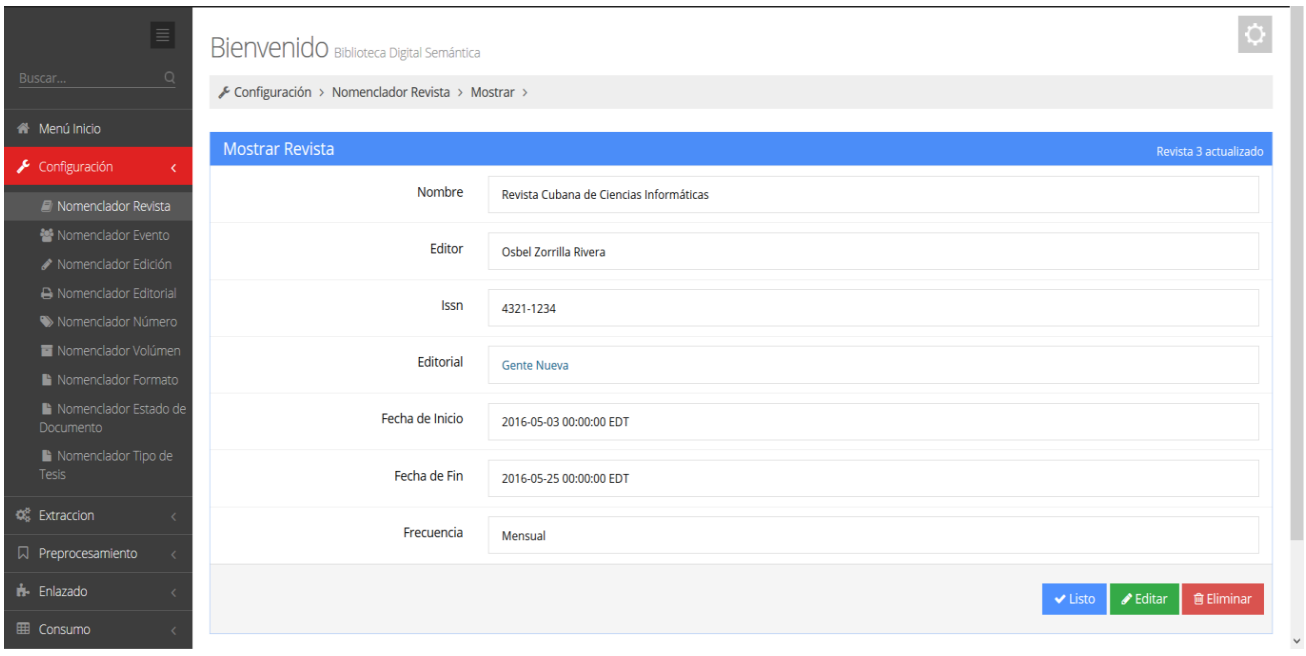
Número: 5	Nombre del requisito: Mostrar datos de la revista
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El sistema debe permitir al usuario visualizar los datos de la revista una vez que esta ha sido creada para que verifique si son correctos, de esta forma el usuario podrá identificar si ha cometido un error antes de registrar la revista en el sistema.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XXIV: Historia de usuario Crear editorial.

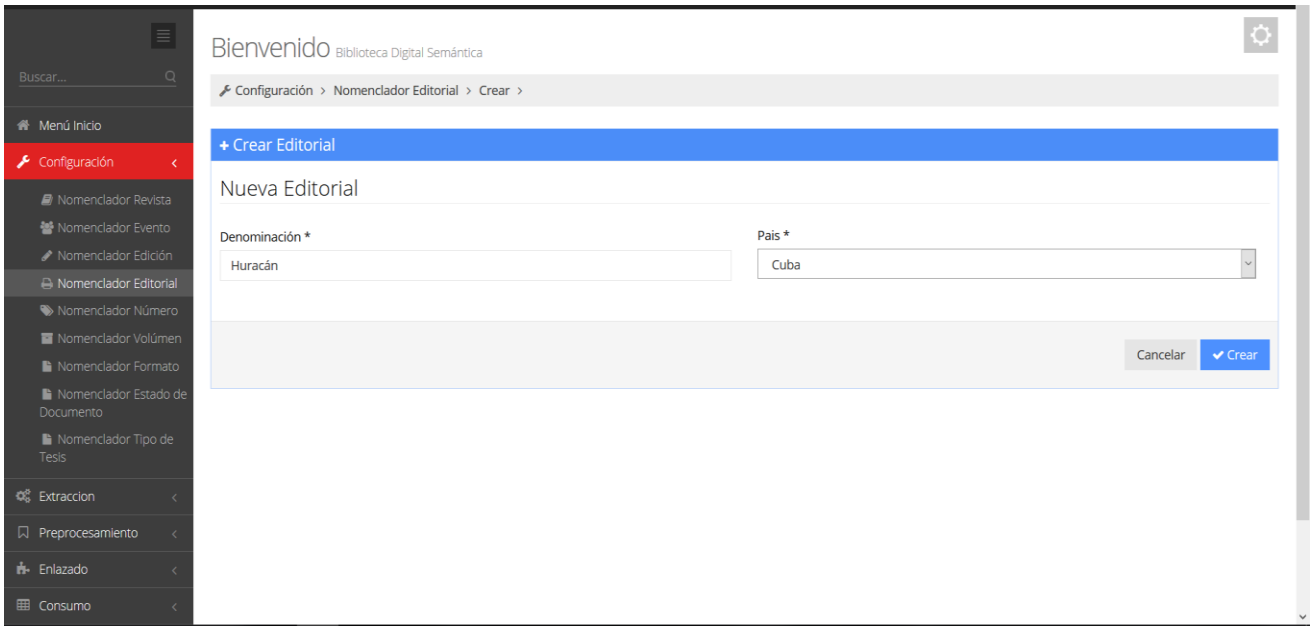
Número: 11	Nombre del requisito: Crear editorial
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1día
Descripción: El sistema debe permitir al usuario crear una nueva editorial, introduciendo en la aplicación los datos que le son solicitados, tales como: la denominación de la editorial y el país al que pertenece.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XXV: Historia de usuario Editar editorial.

Número: 7	Nombre del requisito: Editar editorial
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1

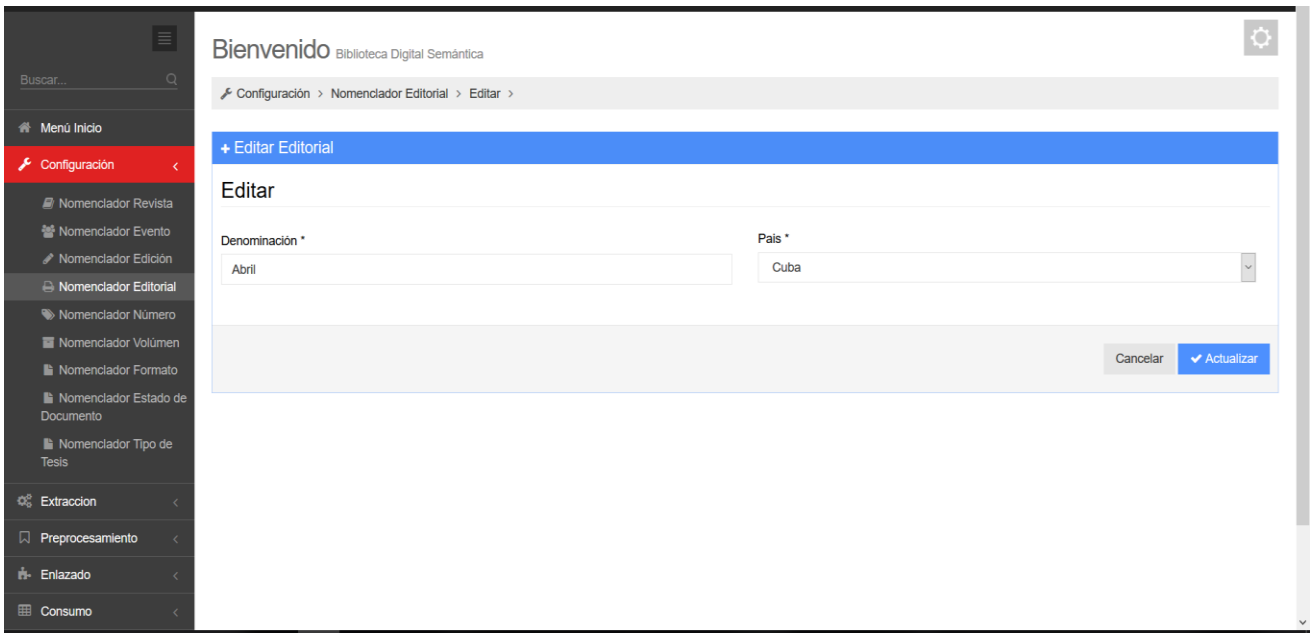
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Editar dentro de las opciones que aparecen al final de la tupla de cada editorial listada, la cual al hacer clic sobre ella permitirá editar los datos de la editorial correspondiente a la tupla seleccionada.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XXVI: Historia de usuario Listar editoriales.

Número: 8	Nombre del requisito: Listar editoriales
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Bajo	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día

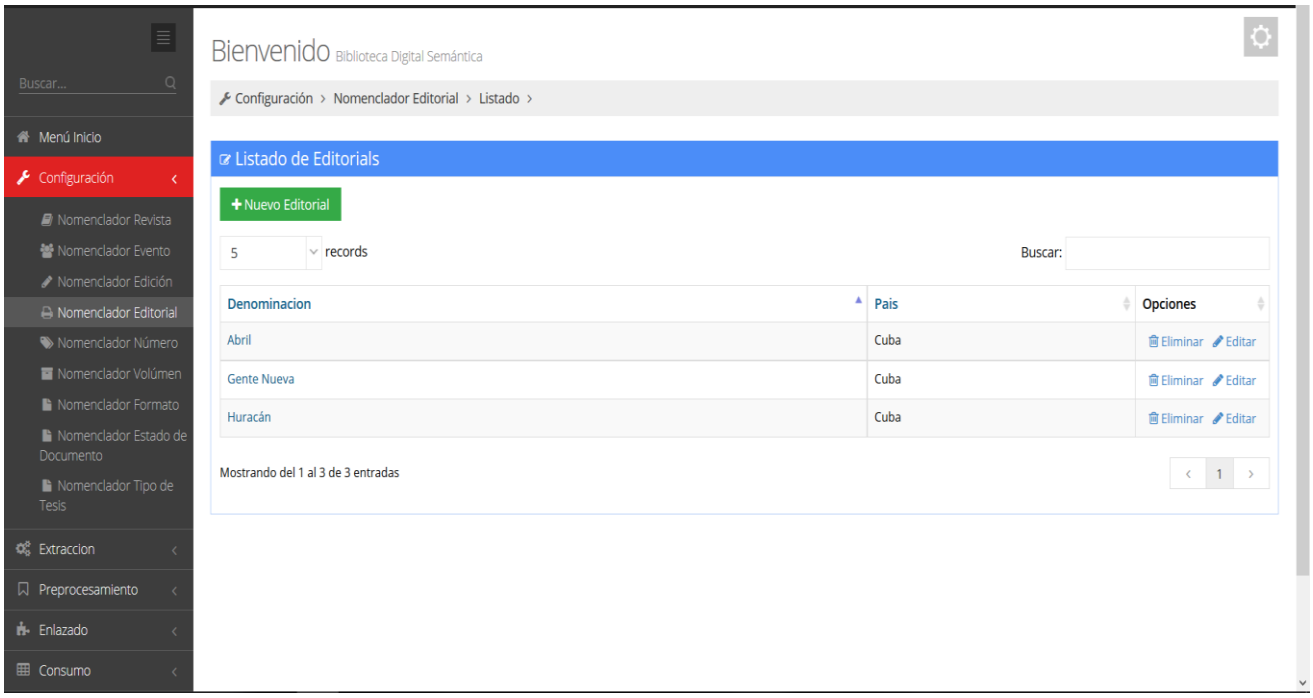
Descripción: En la pantalla principal del nomenclador Editorial se debe mostrar la lista de editoriales que se encuentran registradas en el sistema con sus datos correspondientes, tales como: la denominación de la editorial y el país al que pertenece.
Observaciones:
Prototipo de interfaz: 

Tabla XXVII: Historia de usuario Eliminar editorial.

Número: 9	Nombre del requisito: Eliminar editorial
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Eliminar dentro de las opciones que aparecen al final de la tupla de cada editorial listada, la cual al hacer clic sobre ella eliminará del sistema la editorial correspondiente a la tupla seleccionada.	
Observaciones:	
Prototipo de interfaz: No aplica.	

Tabla XXVIII: Historia de usuario Mostrar datos de la editorial.

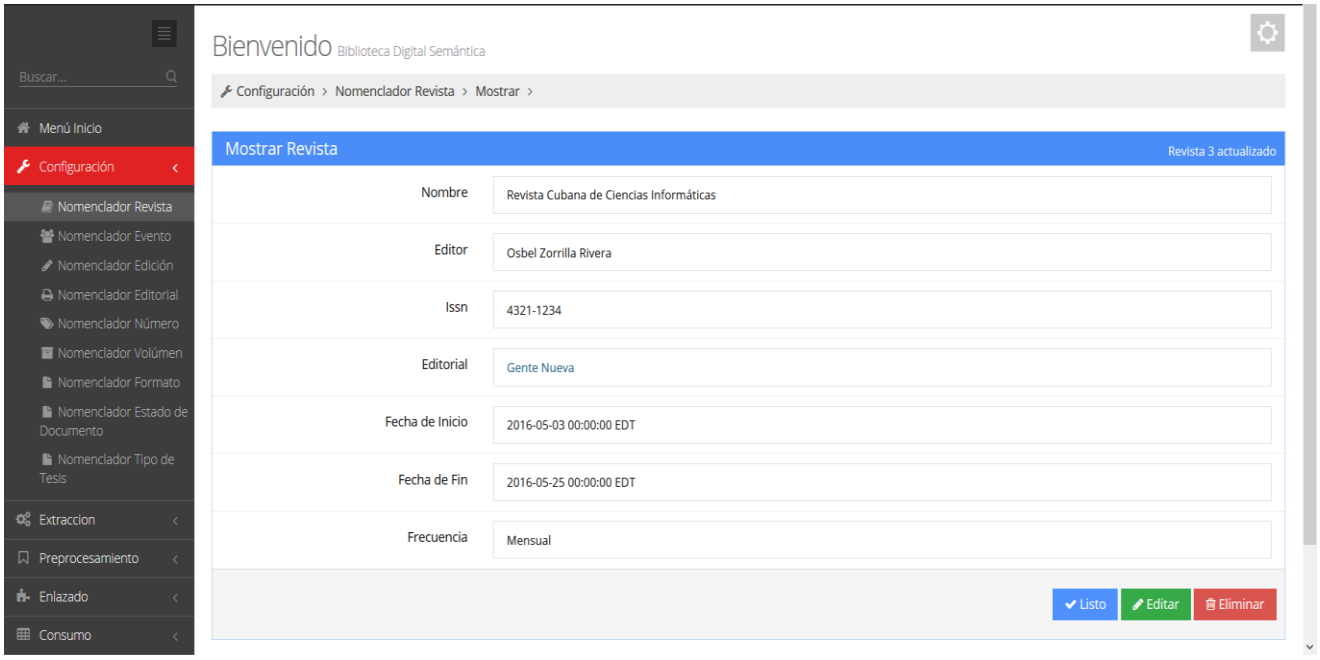
Número: 10	Nombre del requisito: Mostrar datos de la editorial
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El sistema debe permitir al usuario visualizar los datos de la editorial una vez que esta ha sido creada para que verifique si son correctos, de esta forma el usuario podrá identificar si ha cometido un error antes de registrar la editorial en el sistema.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XXIX: Historia de usuario Crear formato.

Número: 11	Nombre del requisito: Crear formato
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1

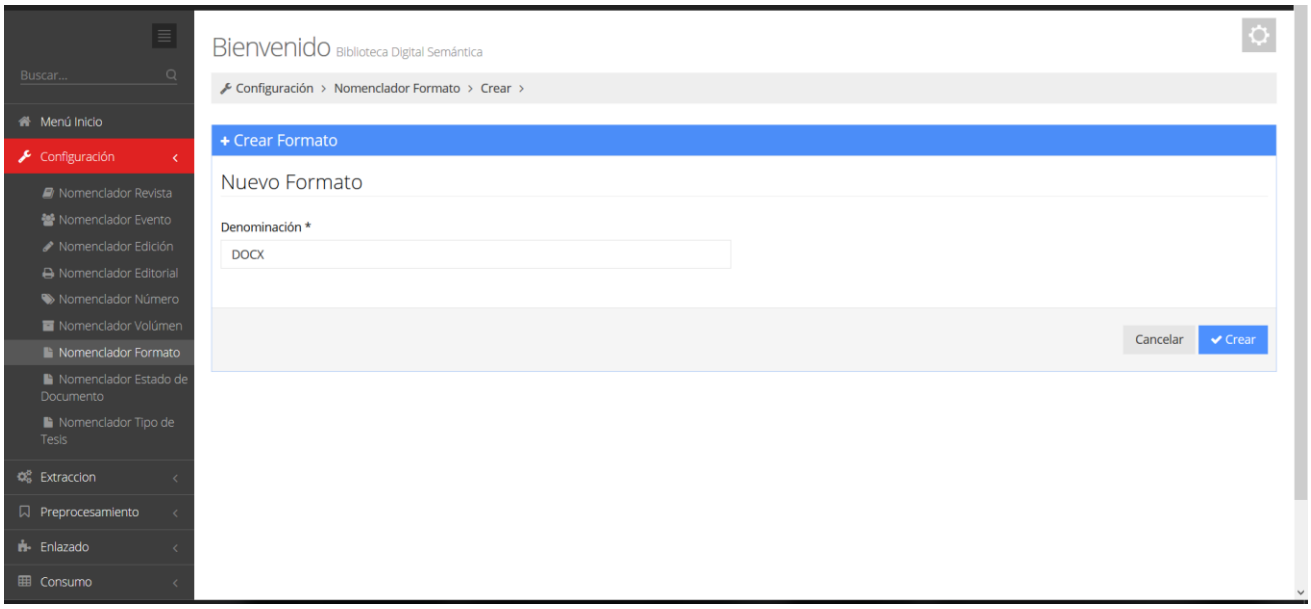
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El sistema debe permitir al usuario crear un nuevo formato de documento introduciendo en la aplicación los datos que le son solicitados, tales como: la denominación.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XXX: Historias de usuario Editar formato.

Número: 12	Nombre del requisito: Editar formato
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Editar dentro de las opciones que aparecen al final de la tupla de cada formato listado, la cual al hacer clic sobre ella permitirá editar los datos del formato correspondiente a la tupla seleccionada.	
Observaciones:	

Prototipo de interfaz:

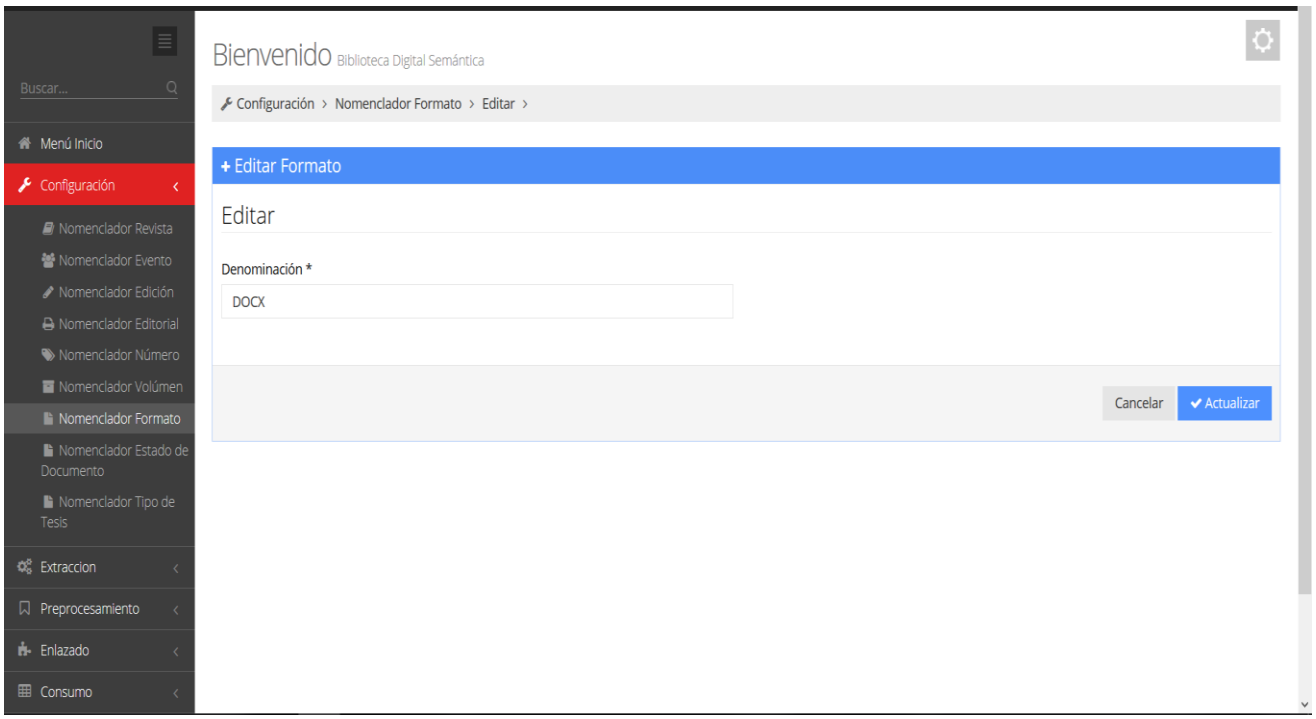


Tabla XXXI: Historia de usuario Listar formatos.

Número: 13	Nombre del requisito: Listar formatos
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador Formato se debe mostrar la lista de formatos de documentos que se encuentran registrados en el sistema con sus datos correspondientes, tales como: la denominación.	
Observaciones:	

Prototipo de interfaz:

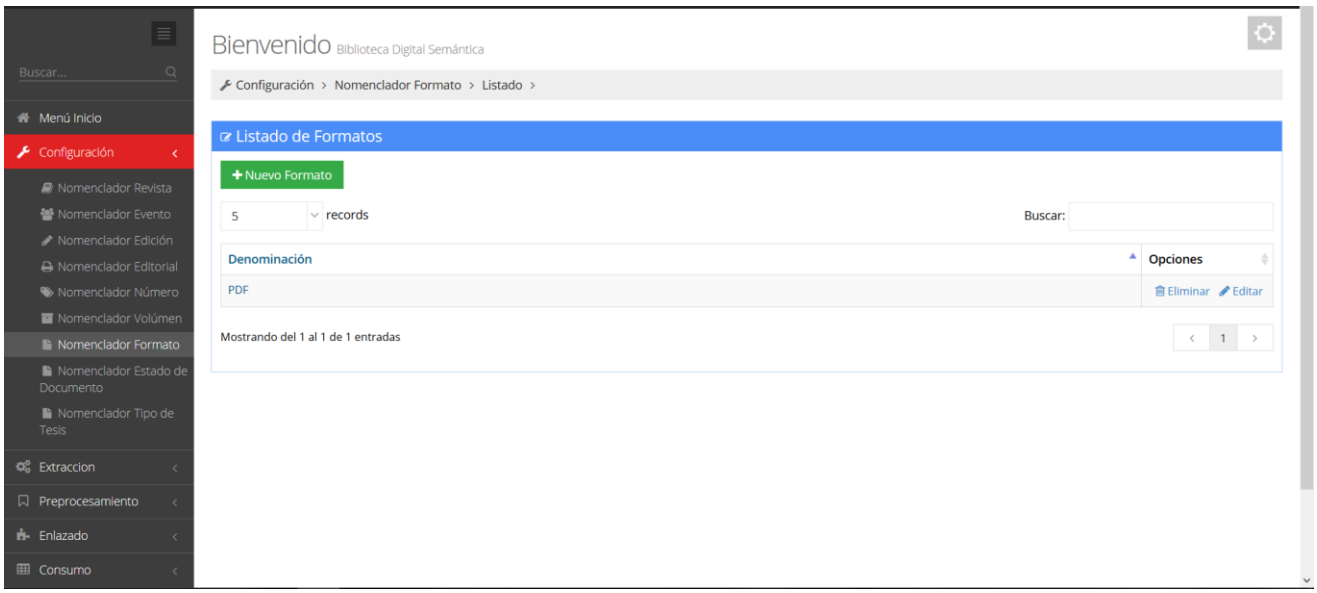


Tabla XXXII: Historia de usuario Eliminar formato.

Número: 14	Nombre del requisito: Eliminar formato
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Eliminar dentro de las opciones que aparecen al final de la tupla de cada formato listado, la cual al hacer clic sobre ella eliminará del sistema el formato correspondiente a la tupla seleccionada.	
Observaciones:	
Prototipo de interfaz: No aplica.	

Tabla XXXIII: Historia de usuario Mostrar datos del formato.

Número: 15	Nombre del requisito: Mostrar datos del formato

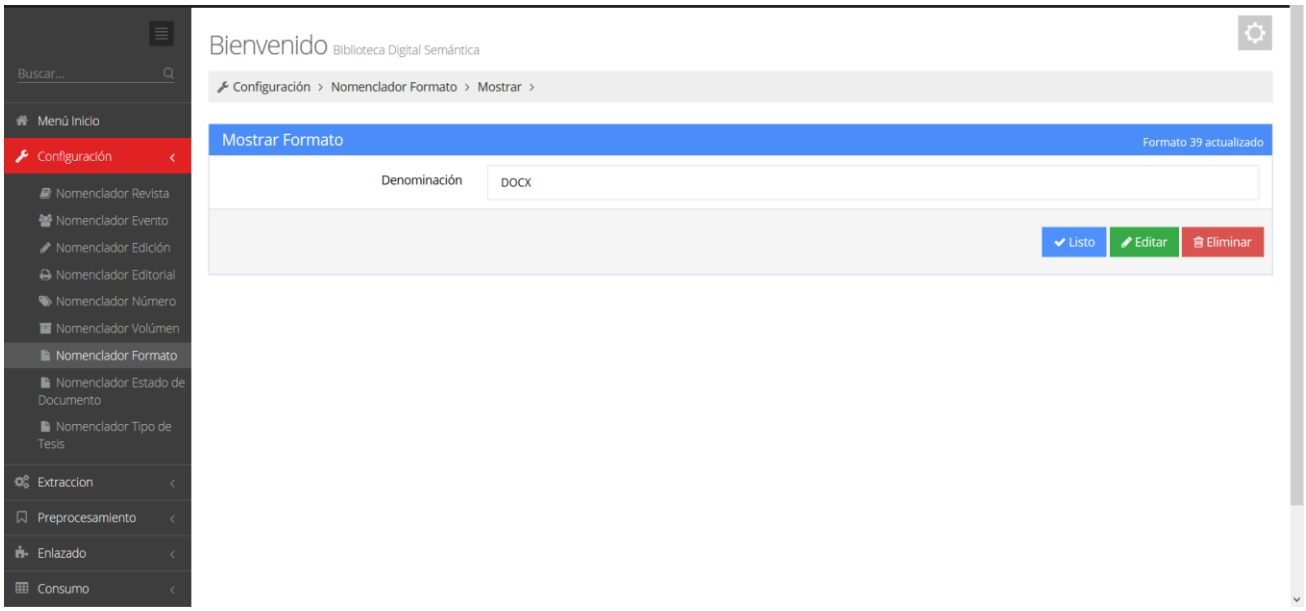
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 dia
Descripción: El sistema debe permitir al usuario visualizar los datos de la editorial una vez que esta ha sido creada para que verifique si son correctos, de esta forma el usuario podrá identificar si ha cometido un error antes de registrar la editorial en el sistema.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XXXIV: Historia de usuario Crear tipo de tesis.

Número: 16	Nombre del requisito: Crear tipo de tesis
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día

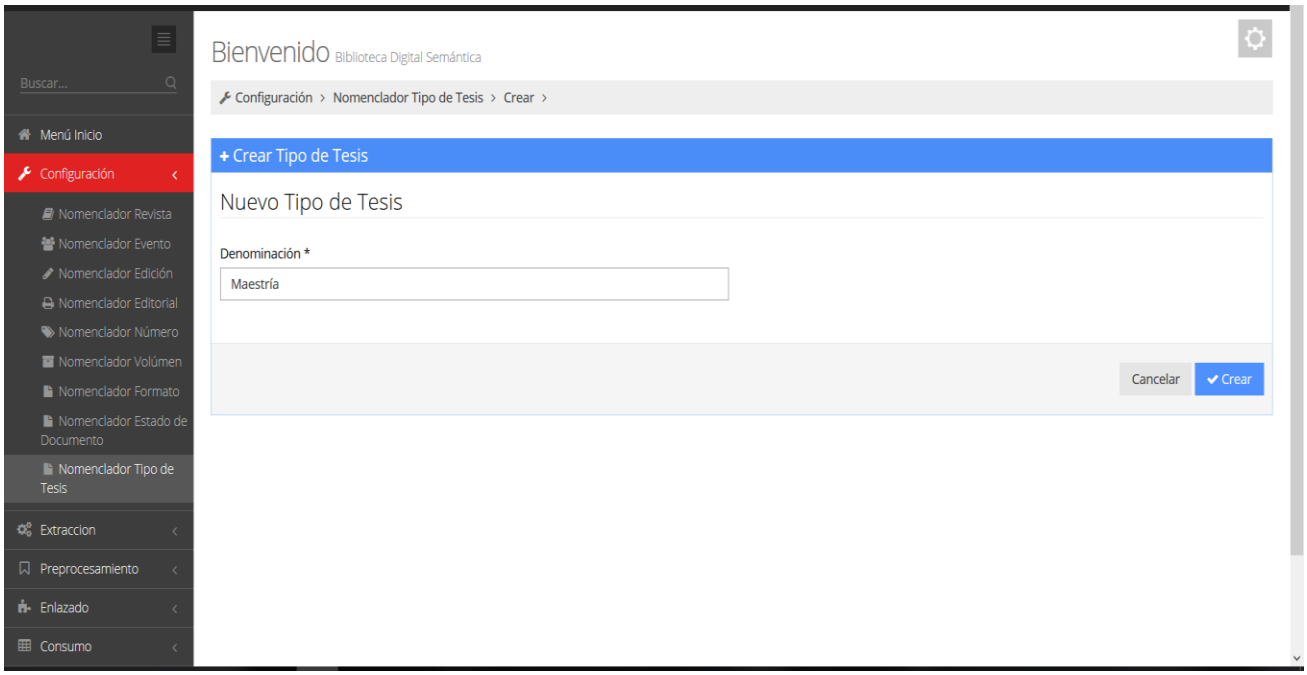
Descripción: El sistema debe permitir al usuario crear un nuevo tipo de tesis introduciendo en la aplicación los datos que le son solicitados, tales como: la denominación.
Observaciones:
Prototipo de interfaz: 

Tabla XXXV: Historia de usuario Editar tipo de tesis.

Número: 17	Nombre del requisito: Editar tipo de tesis
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Editar dentro de las opciones que aparecen al final de la tupla de cada tipo de tesis listado, la cual al hacer clic sobre ella permitirá editar los datos del tipo de tesis correspondiente a la tupla seleccionada.	
Observaciones:	

Prototipo de interfaz:

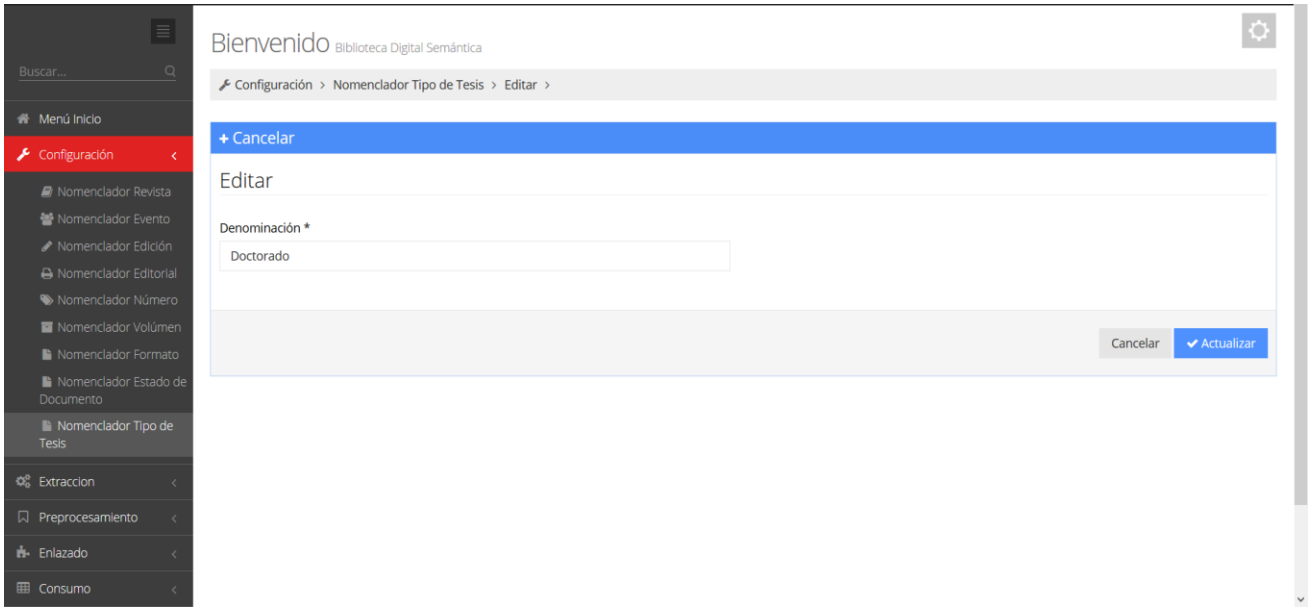


Tabla XXXVI: Historia de usuario Listar tipos de tesis.

Número: 18	Nombre del requisito: Listar tipos de tesis
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1día
Descripción: En la pantalla principal del nomenclador Tipo de tesis se debe mostrar la lista de tipos de tesis que se encuentran registrados en el sistema con sus datos correspondientes, tales como: la denominación.	
Observaciones:	

Prototipo de interfaz:

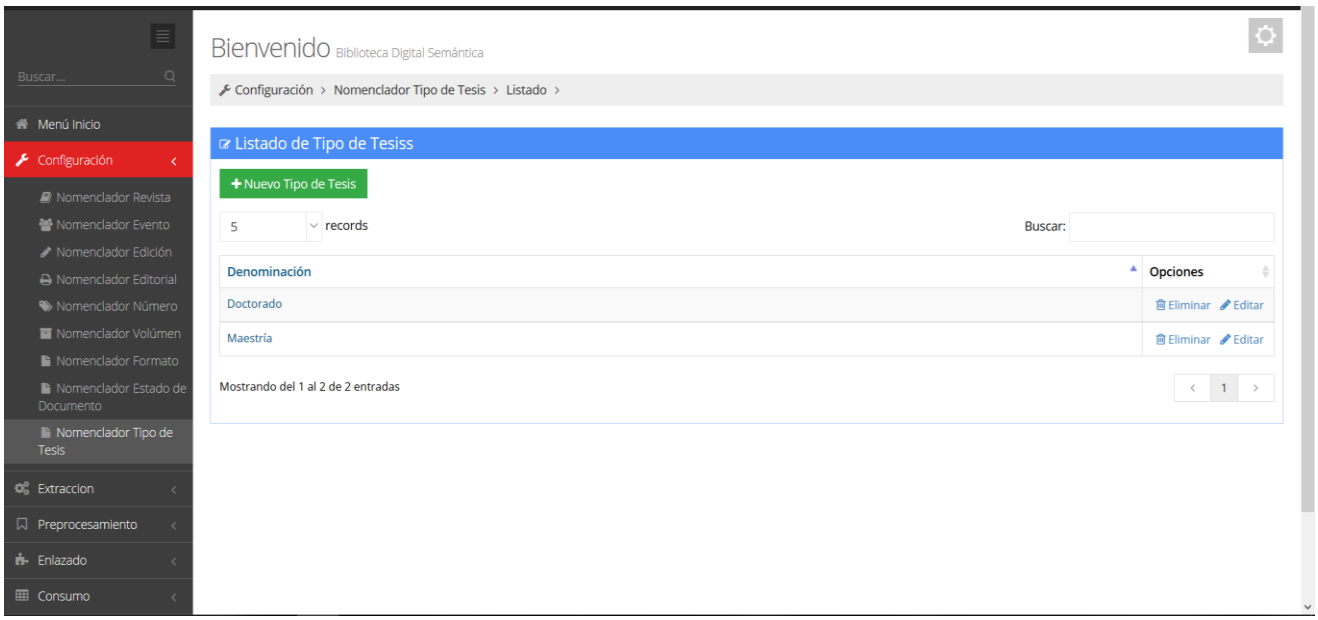


Tabla XXXVII: Historia de usuario Eliminar tipo de tesis.

Número: 19	Nombre del requisito: Eliminar tipo de tesis
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Eliminar dentro de las opciones que aparecen al final de la tupla de cada tipo de tesis listado, la cual al hacer clic sobre ella eliminará del sistema el tipo de tesis correspondiente a la tupla seleccionada.	
Observaciones:	
Prototipo de interfaz: No aplica.	

Tabla XXXVIII: Historia de usuario Mostrar datos del tipo de tesis.

Número: 20	Nombre del requisito: Mostrar datos del tipo de tesis

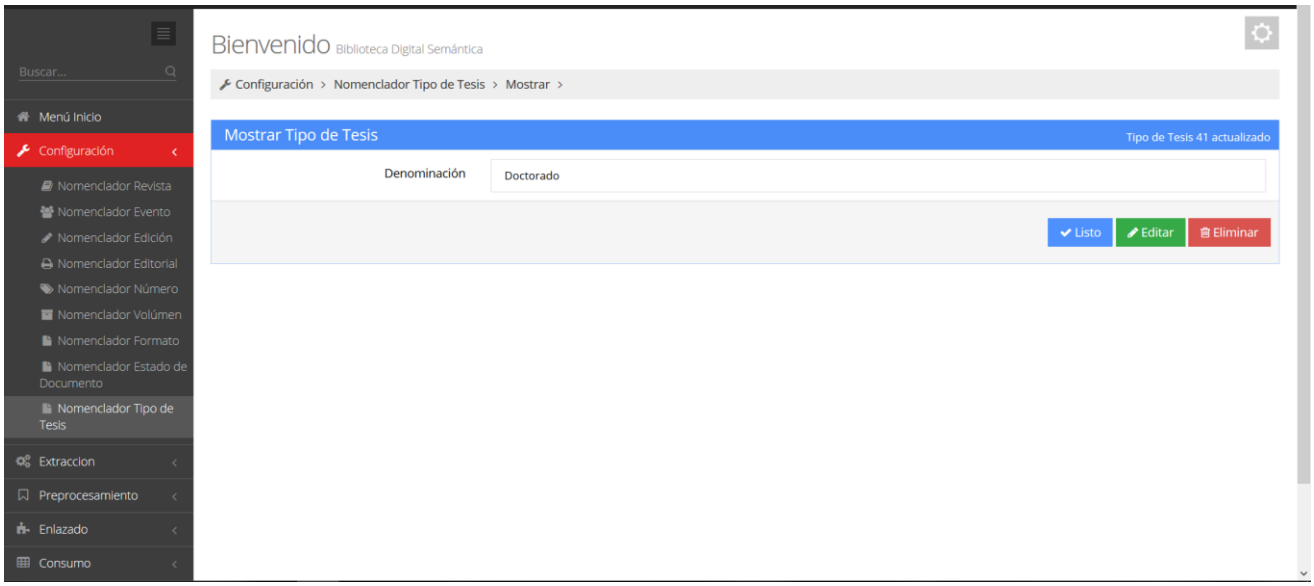
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1día
Descripción: El sistema debe permitir al usuario visualizar los datos del tipo de tesis una vez que esta ha sido creada para que verifique si son correctos, de esta forma el usuario podrá identificar si ha cometido un error antes de registrar el tipo de tesis en el sistema.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XXXIX: Historia de usuario Crear número de revista.

Número: 21	Nombre del requisito: Crear número de revista
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1día
Descripción: El sistema debe permitir al usuario crear un nuevo número de una revista introduciendo en la aplicación los datos que le son solicitados, tales como: el número que será asignado y el volumen al que pertenece el número de la revista.	

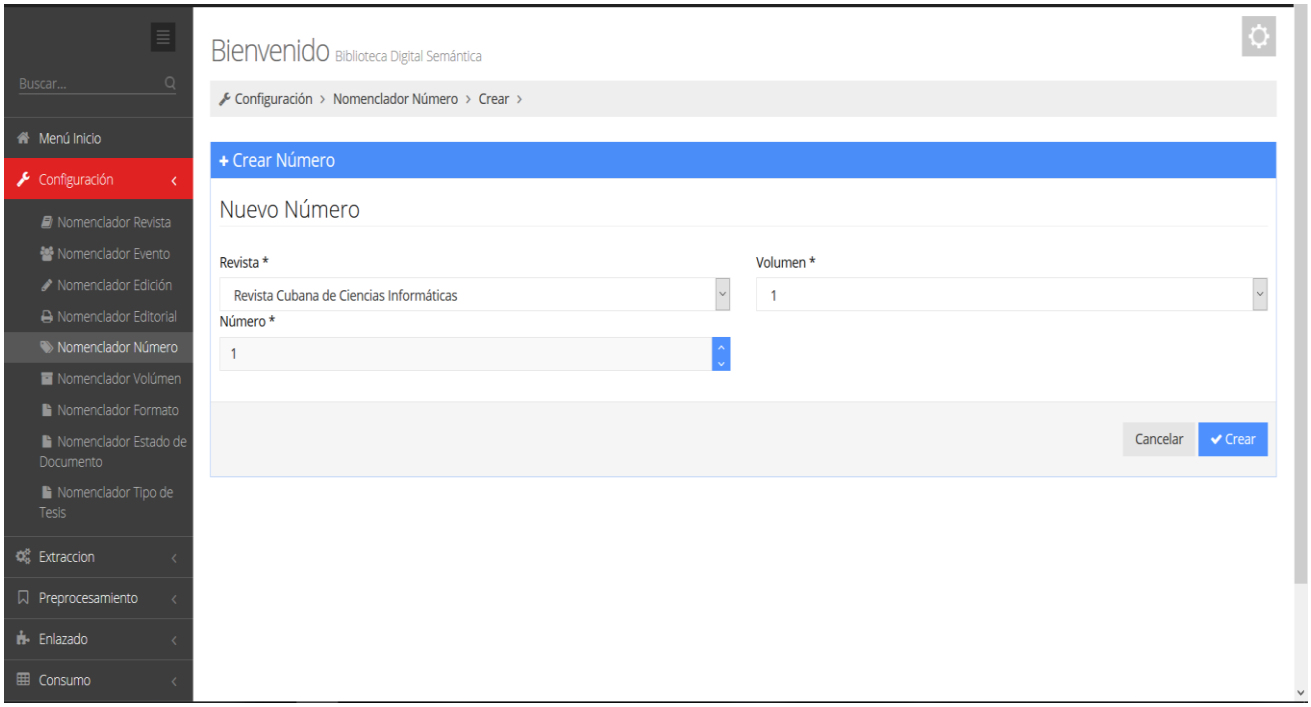
Observaciones:
Prototipo de interfaz: 

Tabla XL: Historia de usuario Editar número de revista.

Número: 22	Nombre del requisito: Editar número de revista
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Editar dentro de las opciones que aparecen al final de la tupla de cada elemento listado, la cual al hace clic sobre ella permitirá modificar los datos del número correspondiente a la tupla que ha sido seleccionada.	
Observaciones:	
Prototipo de interfaz:	

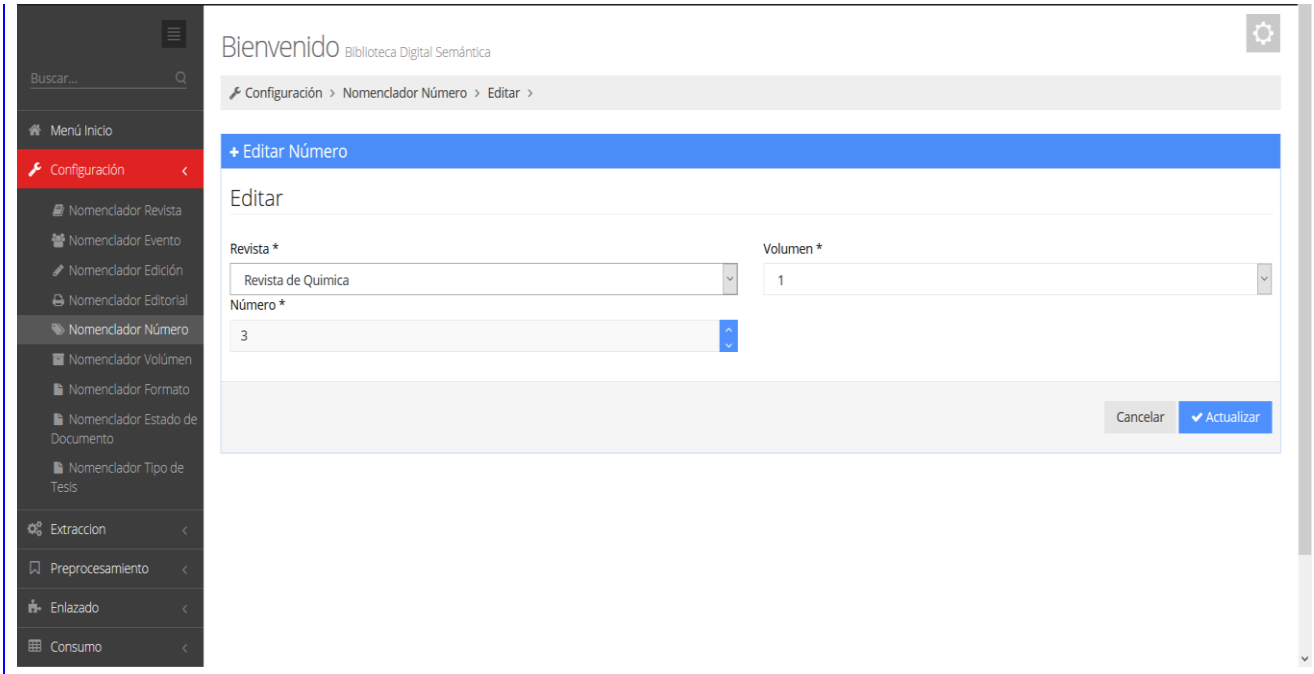


Tabla XLI: Historia de usuario Listar números de revistas.

Número: 23	Nombre del requisito: Listar números de revistas
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador Número se deben listar todos los números de las revistas registradas en el sistema, así como los volúmenes a los que pertenecen dichos números.	
Observaciones:	

Prototipo de interfaz:

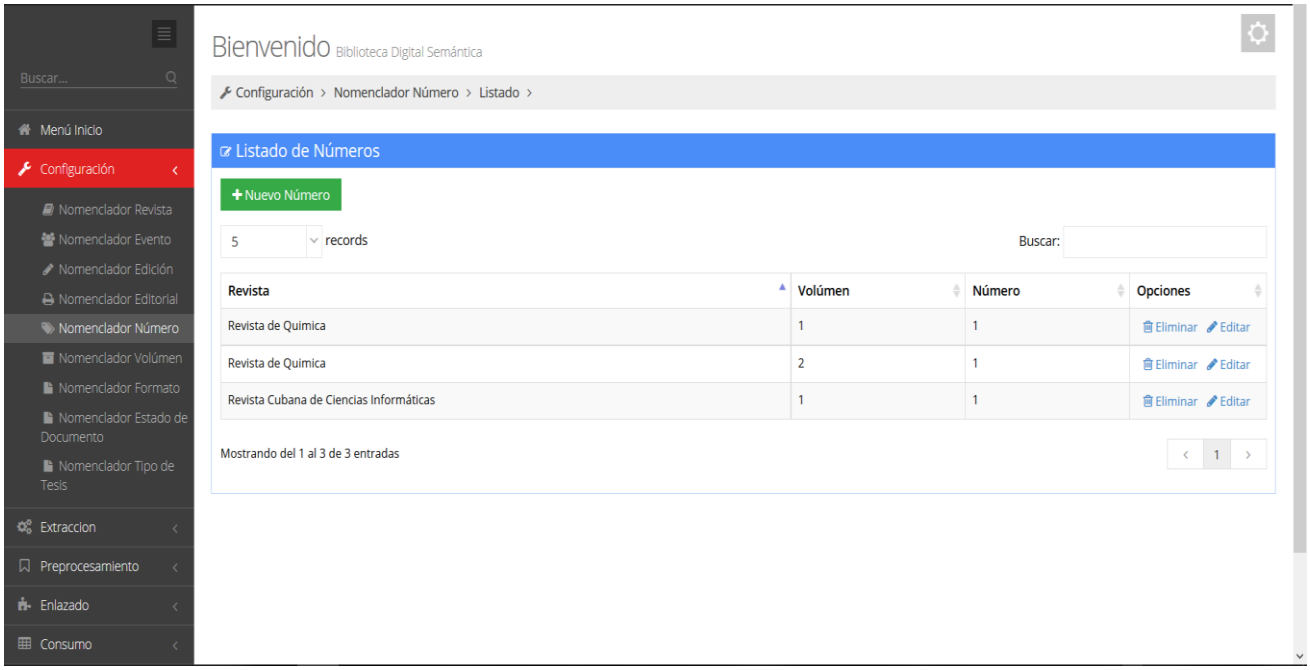


Tabla XLII: Historia de usuario Eliminar número de revista.

Número: 24	Nombre del requisito: Eliminar número de revista
Programador: Paul Nuñez Garcia, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Eliminar dentro de las opciones que aparecen al final de la tupla de cada elemento listado, la cual al hacer clic sobre ella eliminará el número correspondiente a la tupla seleccionada.	
Observaciones:	
Prototipo de interfaz: No aplica.	

Tabla XLIII: Historia de usuario Mostrar datos del número de la revista.

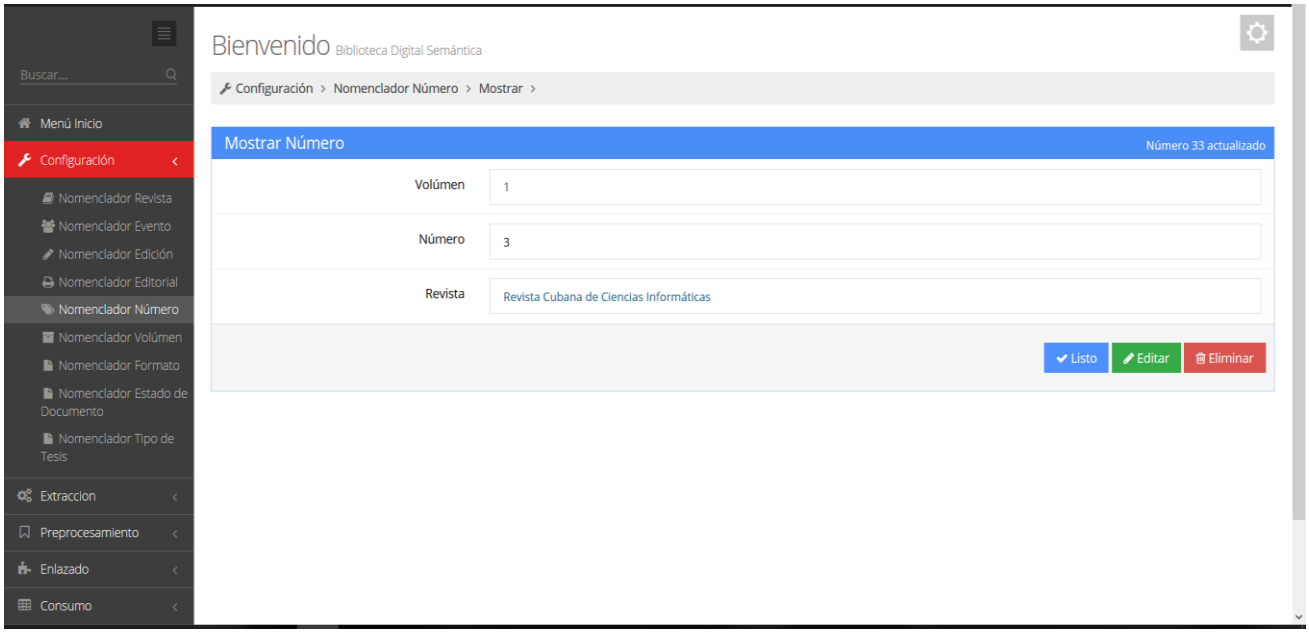
Número: 25	Nombre del requisito: Mostrar datos del número de la revista
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El sistema debe permitir al usuario visualizar los datos del número de la revista una vez que este ha sido creado para que verifique si son correctos, de esta forma el usuario podrá identificar si ha cometido un error antes de registrar la revista en el sistema.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XLIV: Historia de usuario Crear evento.

Número: 26	Nombre del requisito: Crear evento
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día

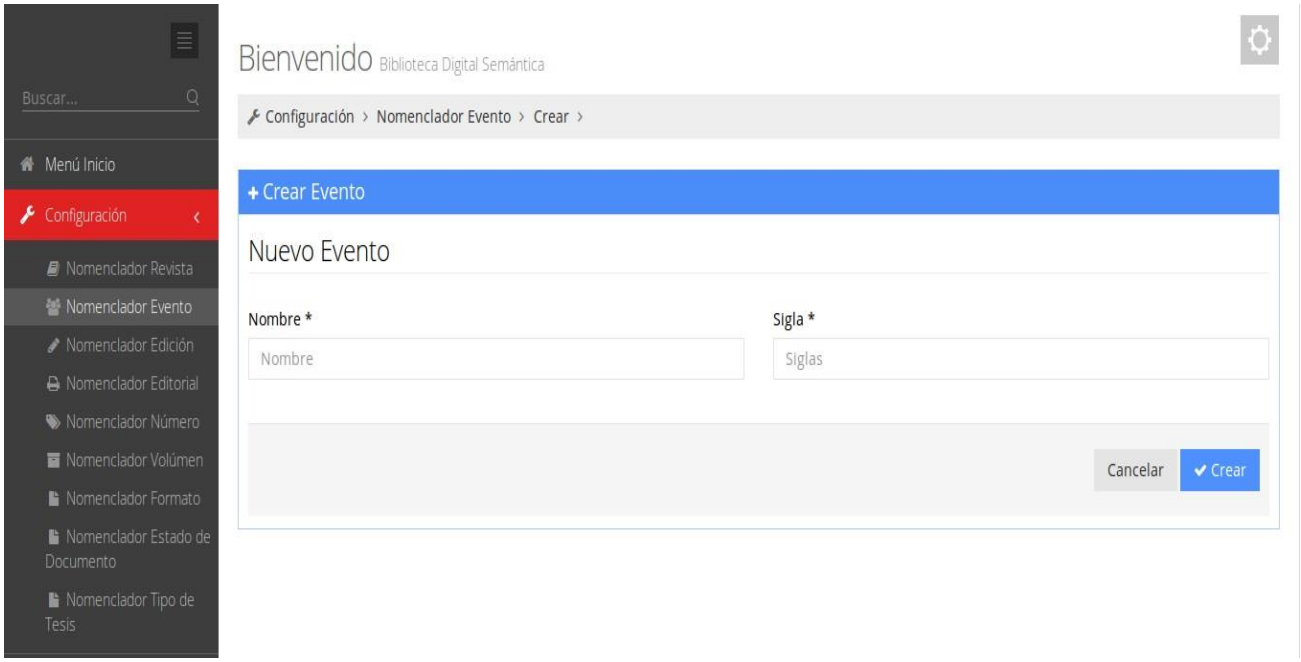
Descripción: El sistema debe permitir al usuario introducir en la aplicación los datos que son necesarios de los eventos que desea adicionar introduciendo el nombre del evento y debido a que en ocasiones los nombres de los eventos coinciden se definen también las siglas del evento.
Observaciones:
Prototipo de interfaz: 

Tabla XLV: Historia de usuario Editar evento.

Número: 27	Nombre del requisito: Editar evento
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: Descripción: En la pantalla principal del nomenclador se mostrará la opción Editar dentro de las opciones que aparecen al final de la tupla de cada evento listado, la cual al hacer clic sobre ella permitirá al usuario modificar los datos que considere necesarios del evento correspondiente a la tupla que ha sido seleccionada.	
Observaciones:	

Prototipo de interfaz:

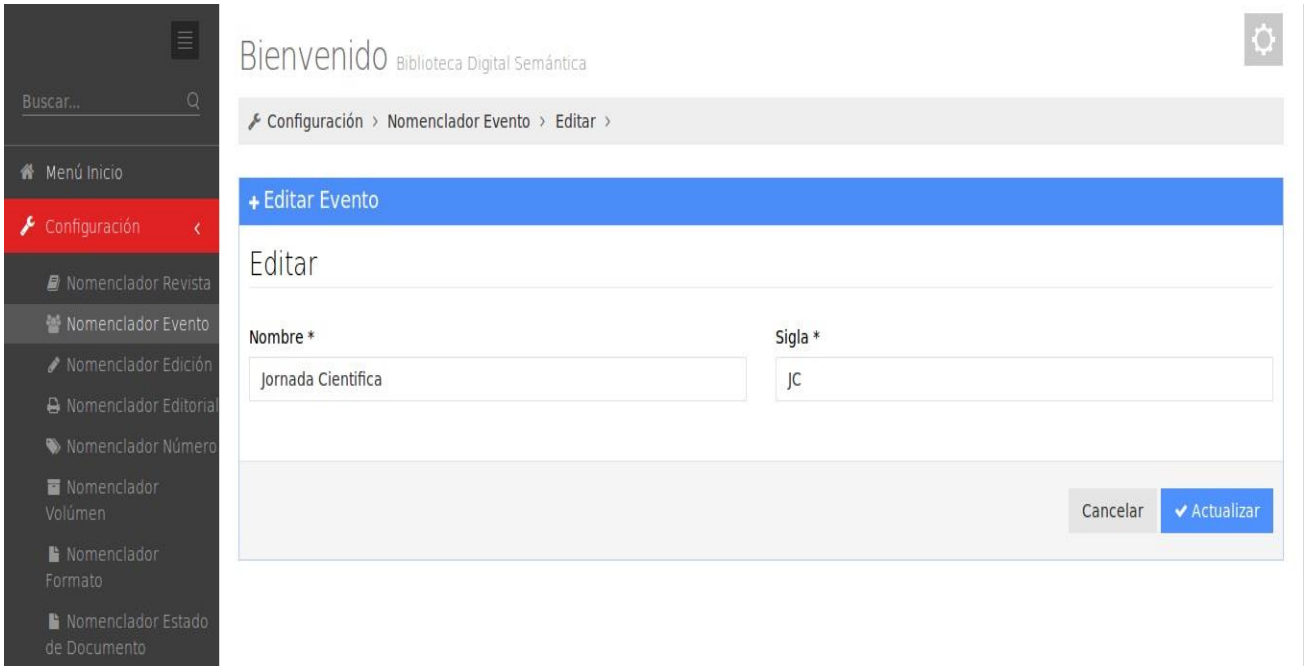


Tabla XLVI: Historia de usuario Listar evento.

Número: 28	Nombre del requisito: Listar evento
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 días
Descripción: En la pantalla principal del nomenclador Evento se deben listar todos los eventos que han sido registrados en el sistema con sus datos correspondientes tales como: Nombre y Siglas.	
Observaciones:	

Prototipo de interfaz:

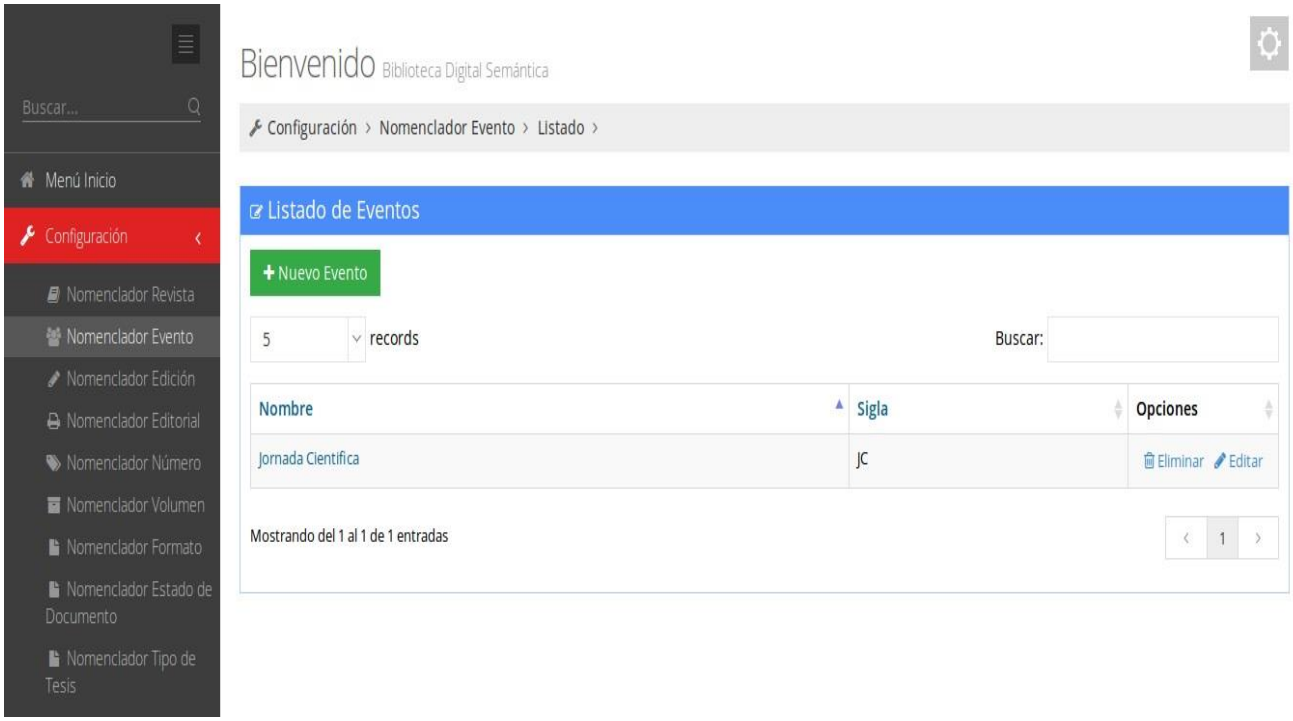


Tabla XLVII: Historia de usuario Eliminar evento.

Número: 29	Nombre del requisito: Eliminar evento
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador se mostrará la opción Eliminar dentro de las opciones que aparecen al final de la tupla de cada evento listado, la cual al hacer clic sobre ella eliminará el evento correspondiente a la tupla seleccionada.	
Observaciones:	
Prototipo de interfaz: No aplica	

Tabla XLVIII: Historia de usuario Mostrar datos de evento.

Número: 30	Nombre del requisito: Mostrar datos de evento

Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El sistema debe permitir al usuario visualizar los datos del evento una vez que este ha sido creado para que verifique si son correctos, de esta forma el usuario podrá identificar si ha cometido un error antes de registrar el evento en el sistema.	
Observaciones:	
Prototipo de interfaz: 	

Tabla XLIX: Historia de usuario Crear volumen.

Número: 31	Nombre del requisito: Crear volumen
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día

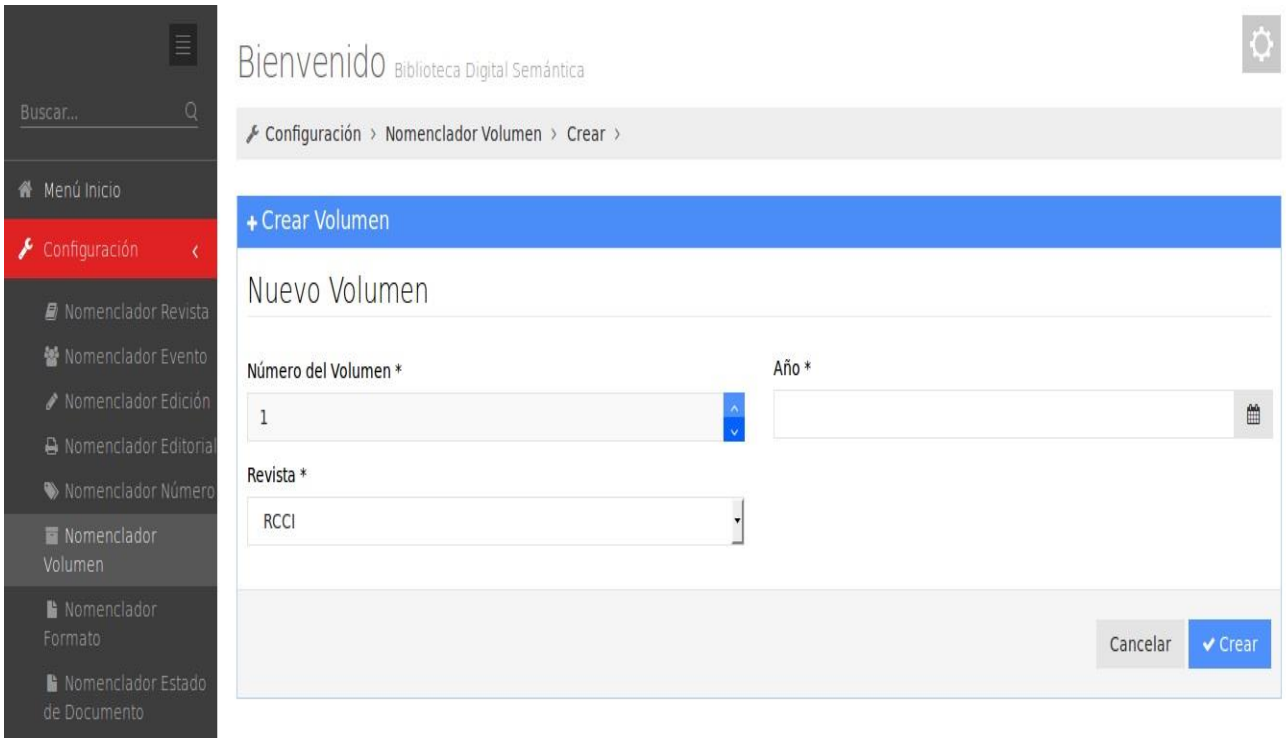
Descripción: El usuario introduce en la aplicación los datos que son necesarios del volumen que desea adicionar, este corresponde a una revista y de él se registran el número del volumen, la revista a la que pertenece y el año al que pertenece.
Observaciones:
Prototipo de interfaz: 

Tabla L: Historia de usuario Editar volumen.

Número: 32	Nombre del requisito: Editar volumen
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: Descripción: En la pantalla principal del nomenclador se mostrará la opción Editar dentro de las opciones que aparecen al final de la tupla de cada volumen listado, la cual al hacer clic sobre ella permitirá al usuario modificar los datos que considere necesarios del volumen correspondiente a la tupla que ha sido seleccionada.	
Observaciones:	

Prototipo de interfaz:

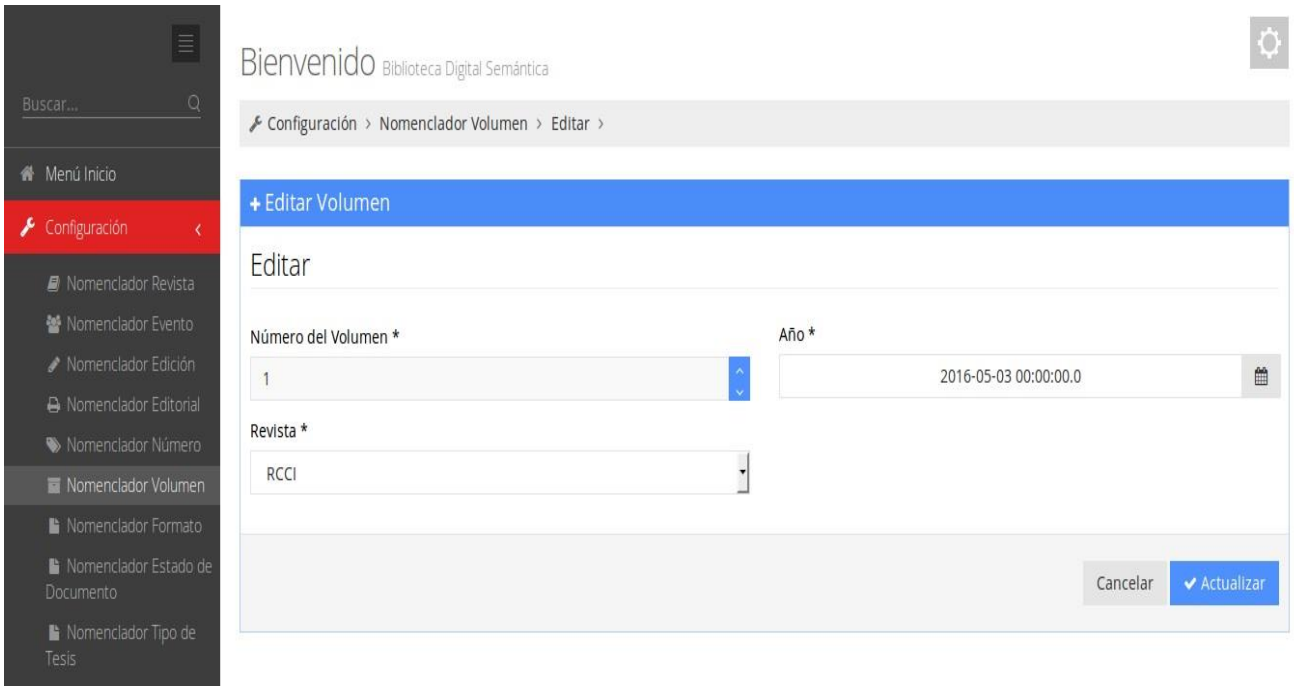


Tabla LI: Historia de usuario Listar volúmenes.

Número: 33	Nombre del requisito: Listar volúmenes
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador Volumen se deben listar todos los volúmenes que han sido registrados en el sistema con sus datos correspondientes tales como: la revista a la que pertenece, el número y el año.	
Observaciones:	

Prototipo de interfaz:

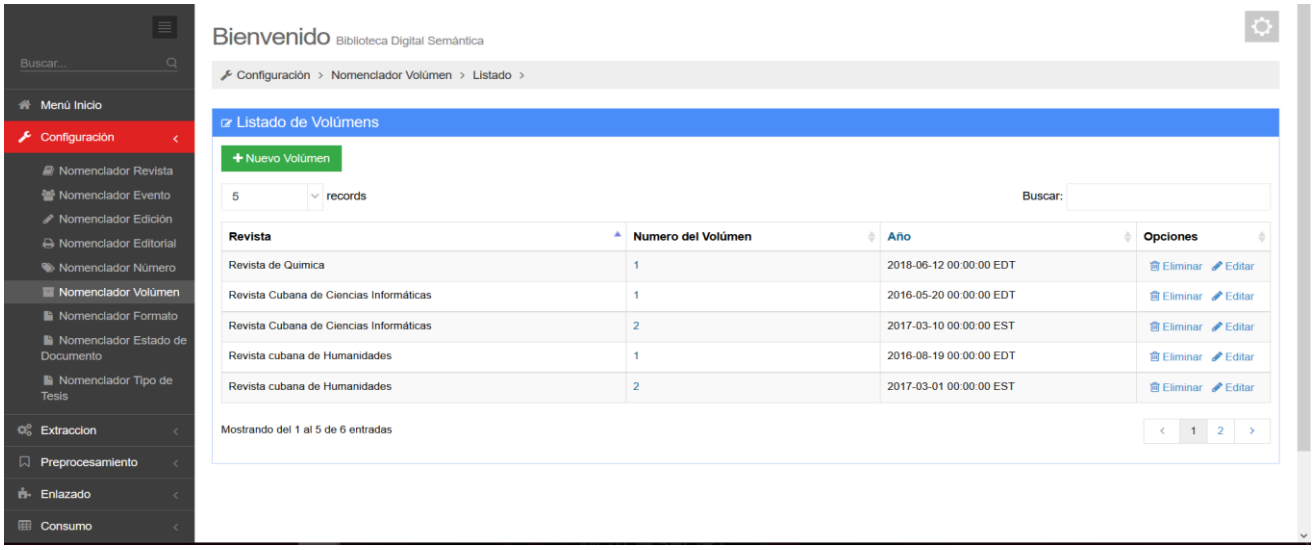


Tabla LII: Historia de usuario Eliminar volumen.

Número: 34	Nombre del requisito: Eliminar volumen
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 días
Descripción: En la pantalla principal del nomenclador se mostrará la opción Eliminar dentro de las opciones que aparecen al final de la tupla de cada volumen listado, la cual al hacer clic sobre ella eliminará el volumen correspondiente a la tupla seleccionada.	
Observaciones:	
Prototipo de interfaz: No aplica	

Tabla LIII: Historia de usuario Mostrar datos volumen.

Número: 35	Nombre del requisito: Mostrar datos volumen
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días

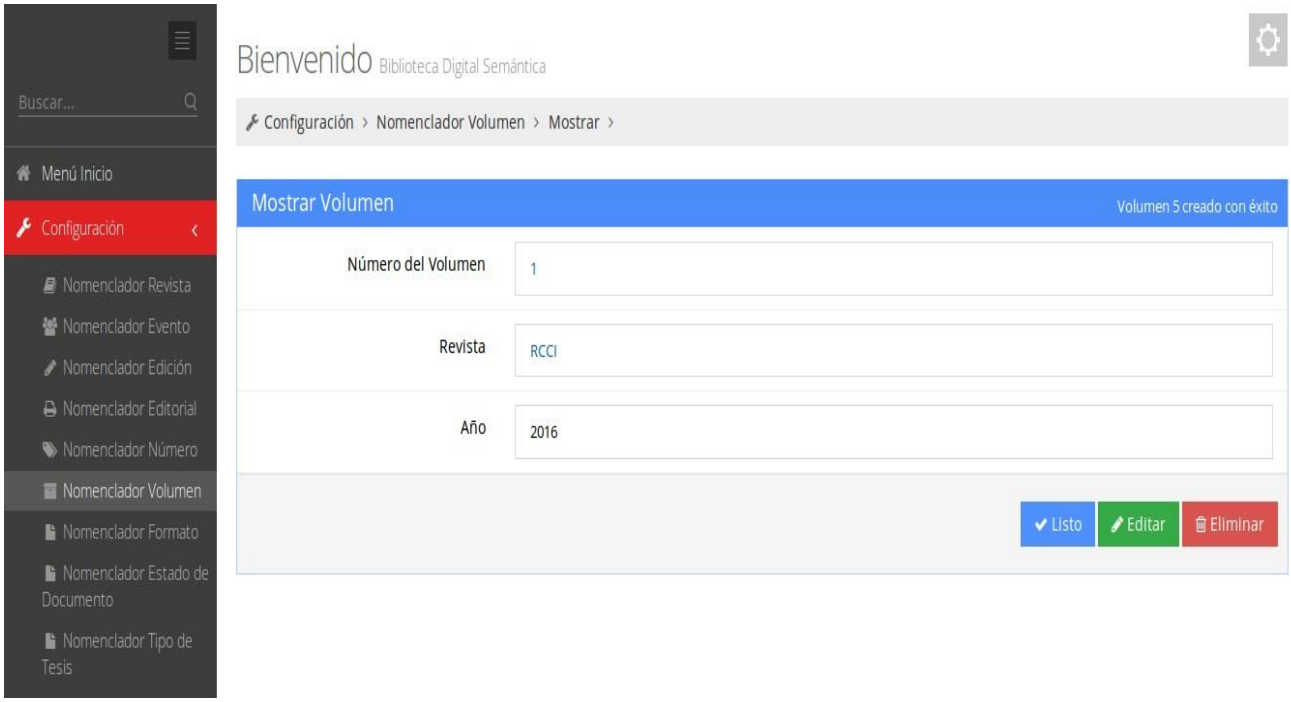
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 días
Descripción: El sistema debe permitir al usuario visualizar los datos del volumen una vez que este ha sido creado para que verifique si son correctos, de esta forma el usuario podrá identificar si ha cometido un error antes de registrar el volumen en el sistema.	
Observaciones:	
Prototipo de interfaz: 	

Tabla LIV: Historia de usuario Crear edición.

Número: 36	Nombre del requisito: Crear edición
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El usuario introduce en la aplicación los datos que le son solicitados por la aplicación para crear la edición deseada, de cada edición se debe registrar la ISBN, el número de la edición, el evento al que pertenece, el país en que se realiza y la fecha en la que se encuentra enmarcado.	

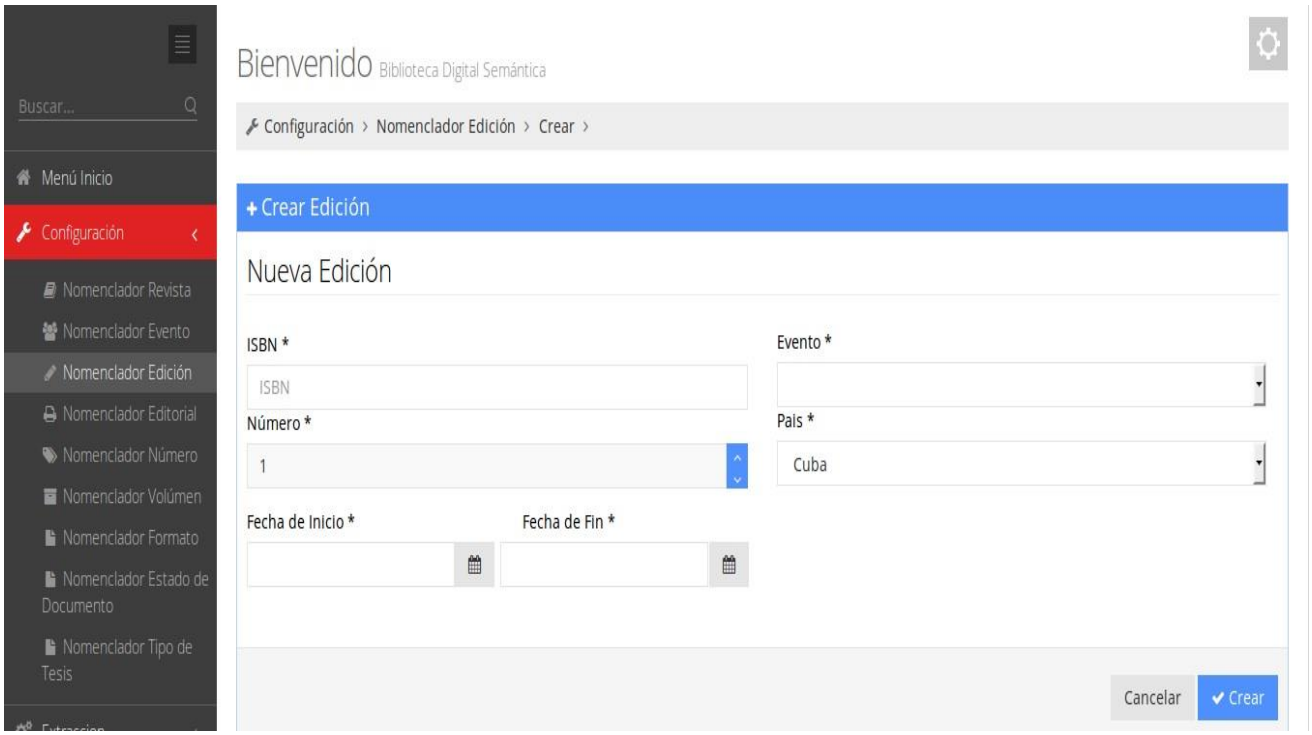
Observaciones:
Prototipo de interfaz: 

Tabla LV: Historia de usuario Editar edición.

Número: 37	Nombre del requisito: Editar edición
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: Descripción: En la pantalla principal del nomenclador se mostrará la opción Editar dentro de las opciones que aparecen al final de la tupla de cada edición listada, la cual al hacer clic sobre ella permitirá al usuario modificar los datos que considere necesarios del volumen correspondiente a la tupla que ha sido seleccionada.	
Observaciones:	

Prototipo de interfaz:

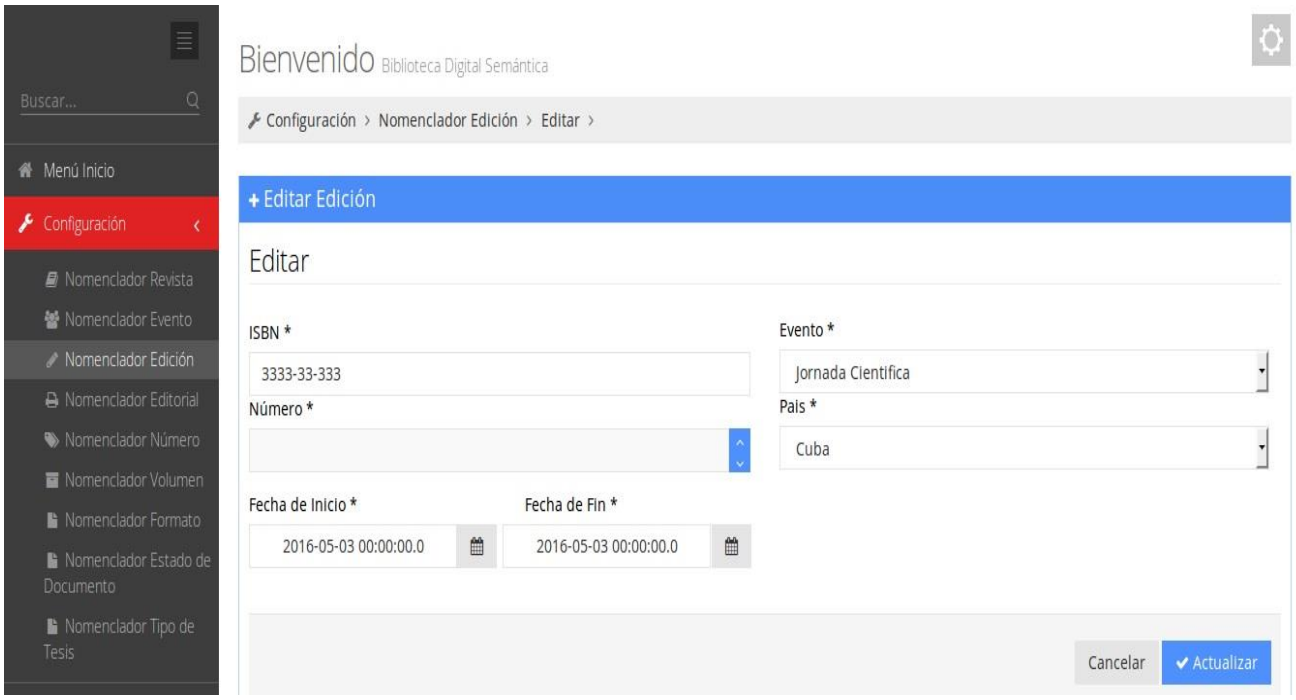


Tabla LVI: Historia de usuario Listar edición.

Número: 38	Nombre del requisito: Listar edición
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador Edición se deben listar todas las ediciones que han sido registradas en el sistema con sus datos correspondientes tales como: el número de la edición, isbn, país donde se realizó, Fecha de inicio, Fecha de fin y el evento al que pertenece la edición	
Observaciones:	

Prototipo de interfaz:

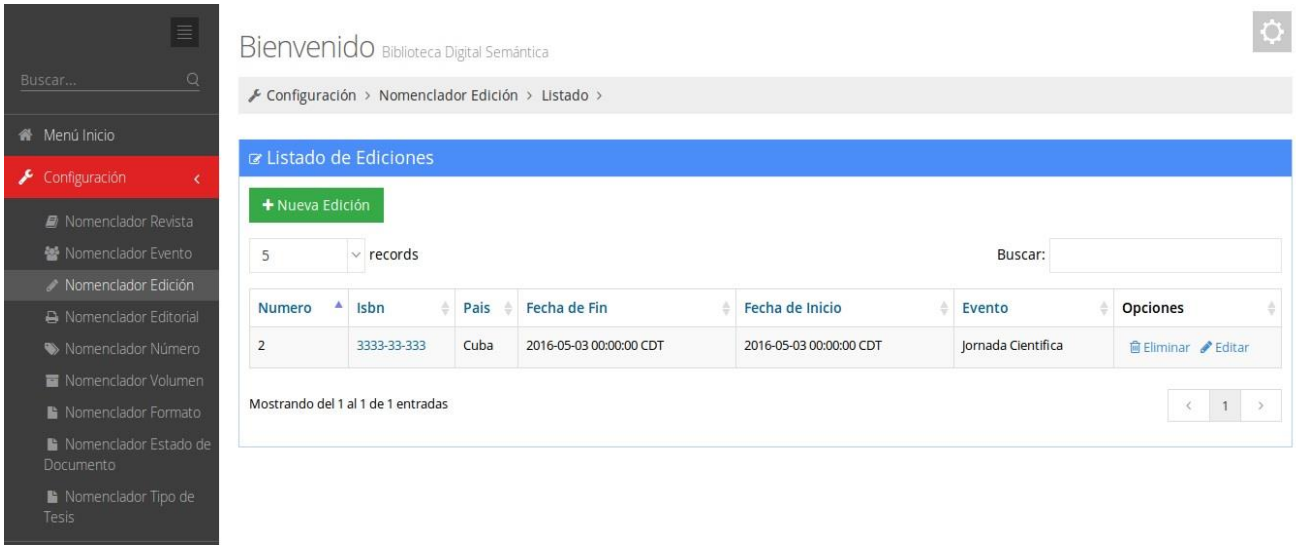


Tabla LVII: Historia de usuario Eliminar edición.

Número: 39	Nombre del requisito: Eliminar edición
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo:Bajo	Tiempo Real: 1 días
Descripción: En la pantalla principal del nomenclador se mostrará la opción Eliminar dentro de las opciones que aparecen al final de la tupla de cada edición listada, la cual al hacer clic sobre ella eliminará la edición correspondiente a la tupla seleccionada.	
Observaciones:	
Prototipo de interfaz: No aplica	

Tabla LVIII: Historia de usuario Mostrar datos de la edición.

Número: 40	Nombre del requisito: Mostrar datos de la edición
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días

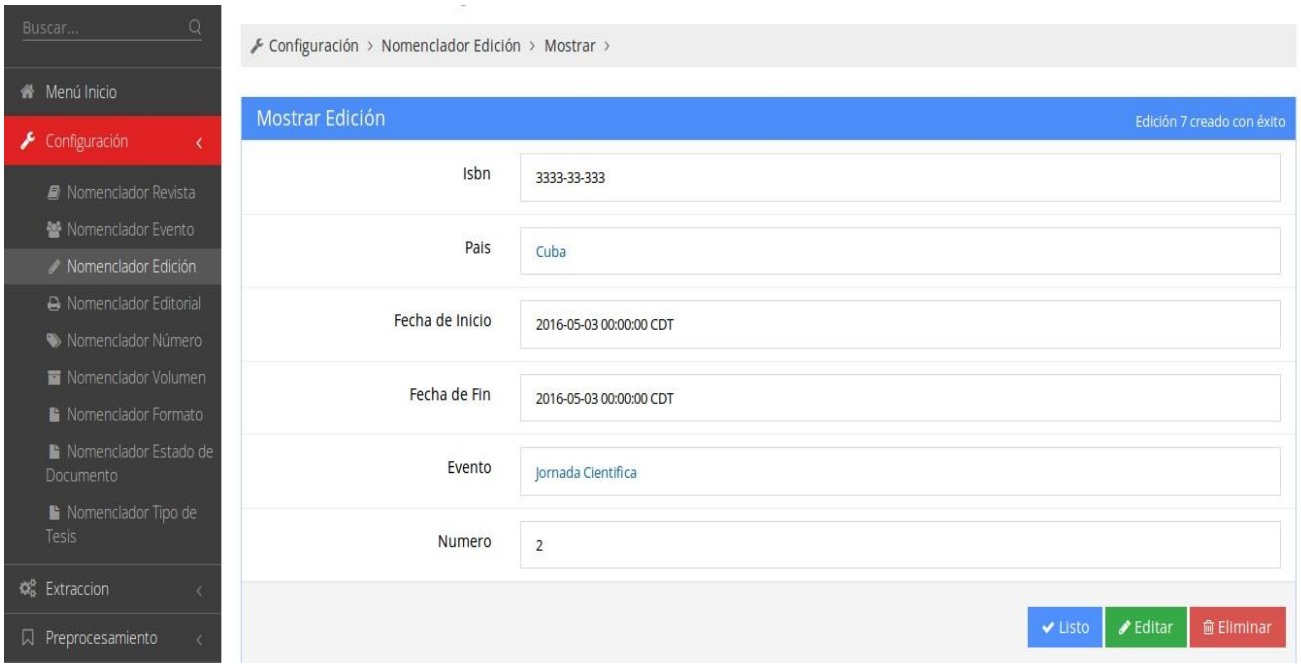
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El sistema debe permitir al usuario visualizar los datos de la edición una vez que esta ha sido creada para que verifique si son correctos, de esta forma el usuario podrá identificar si ha cometido un error antes de registrar el volumen en el sistema.	
Observaciones:	
Prototipo de interfaz: 	

Tabla LIX: Historia de usuario Crear estado.

Número: 41	Nombre del requisito: Crear estado
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El usuario introduce en la aplicación la denominación del estado que desea adicionar, en este caso es el único campo necesario.	
Observaciones:	

Prototipo de interfaz:

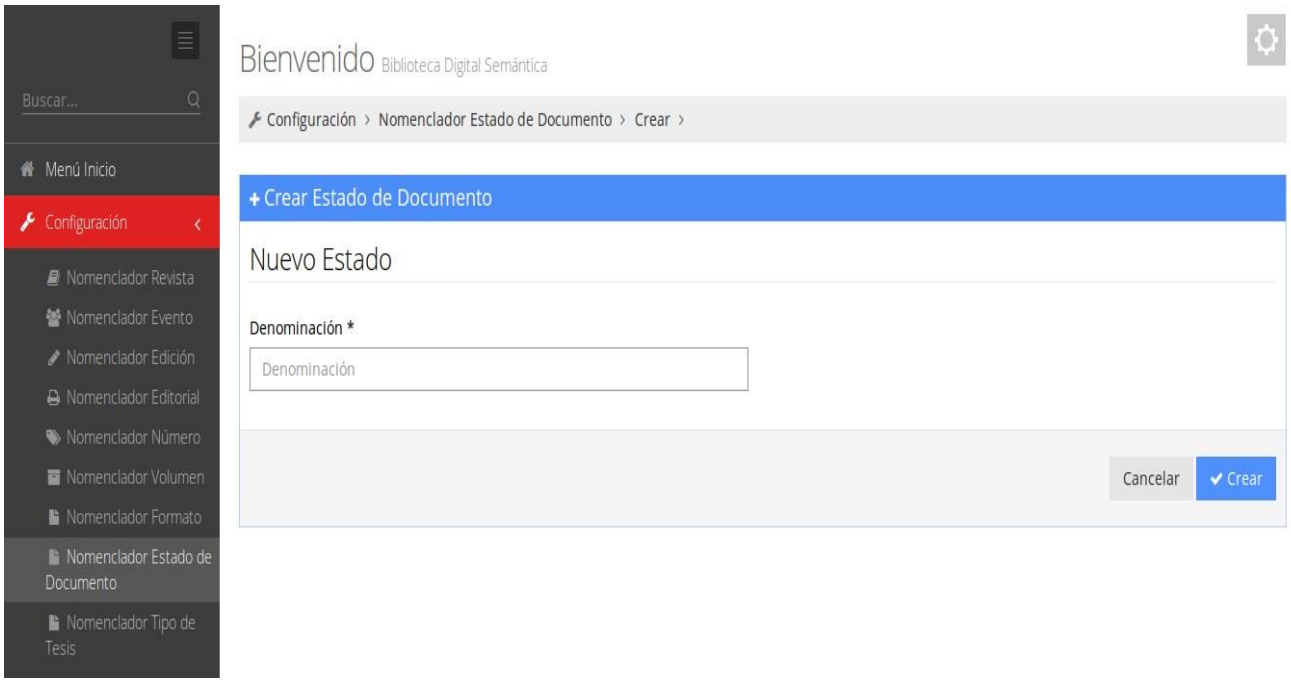


Tabla LX: Historia de usuario Editar estado.

Número: 42	Nombre del requisito: Editar estado
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: Descripción: En la pantalla principal del nomenclador se mostrará la opción Editar dentro de las opciones que aparecen al final de la tupla de cada estado listado, la cual al hacer clic sobre ella permitirá al usuario modificar los datos que considere necesarios del estado correspondiente a la tupla que ha sido seleccionada.	
Observaciones:	

Prototipo de interfaz:

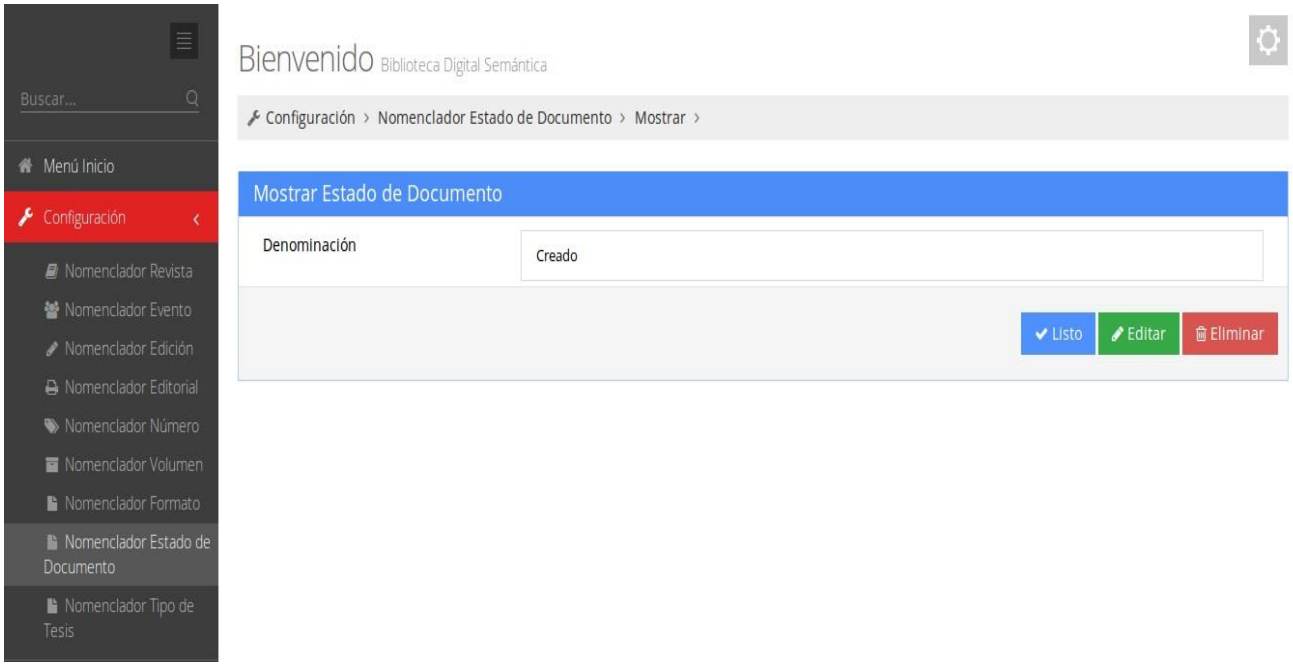


Tabla LXI: Historia de usuario Listar estado.

Número: 43	Nombre del requisito: Listar estado
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador estado se deben listar todos los estados que han sido registrados en el sistema con su correspondiente denominación.	
Observaciones:	

Prototipo de interfaz:

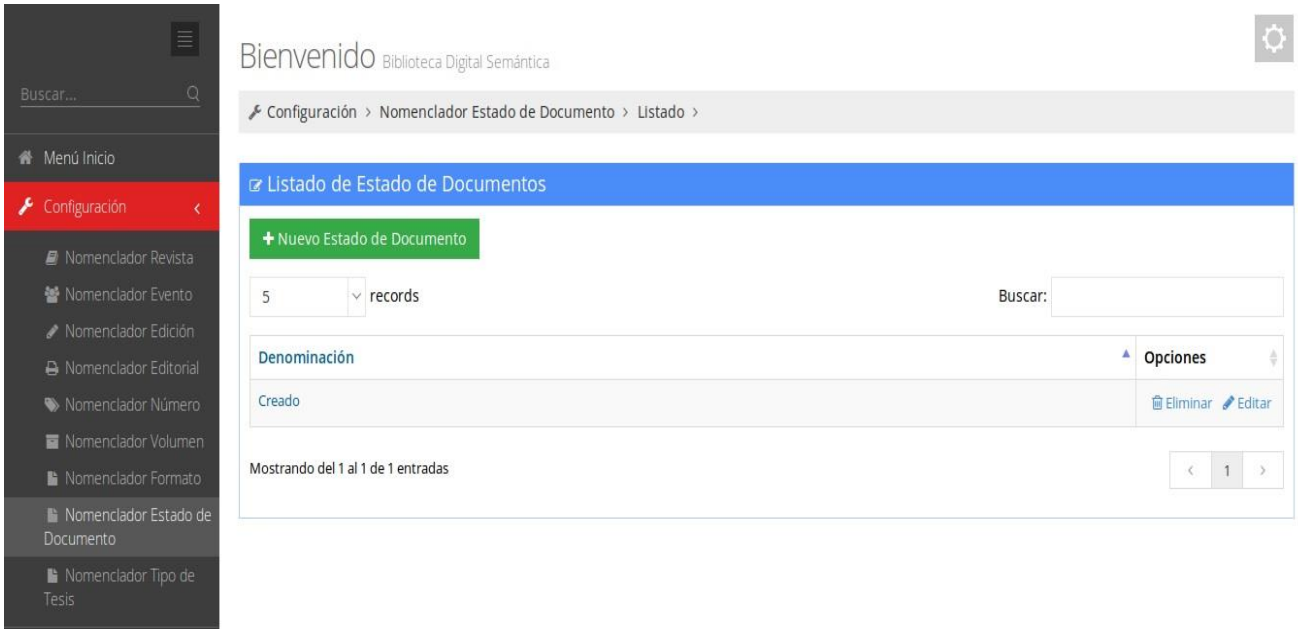


Tabla LXII: Historia de usuario Eliminar estado.

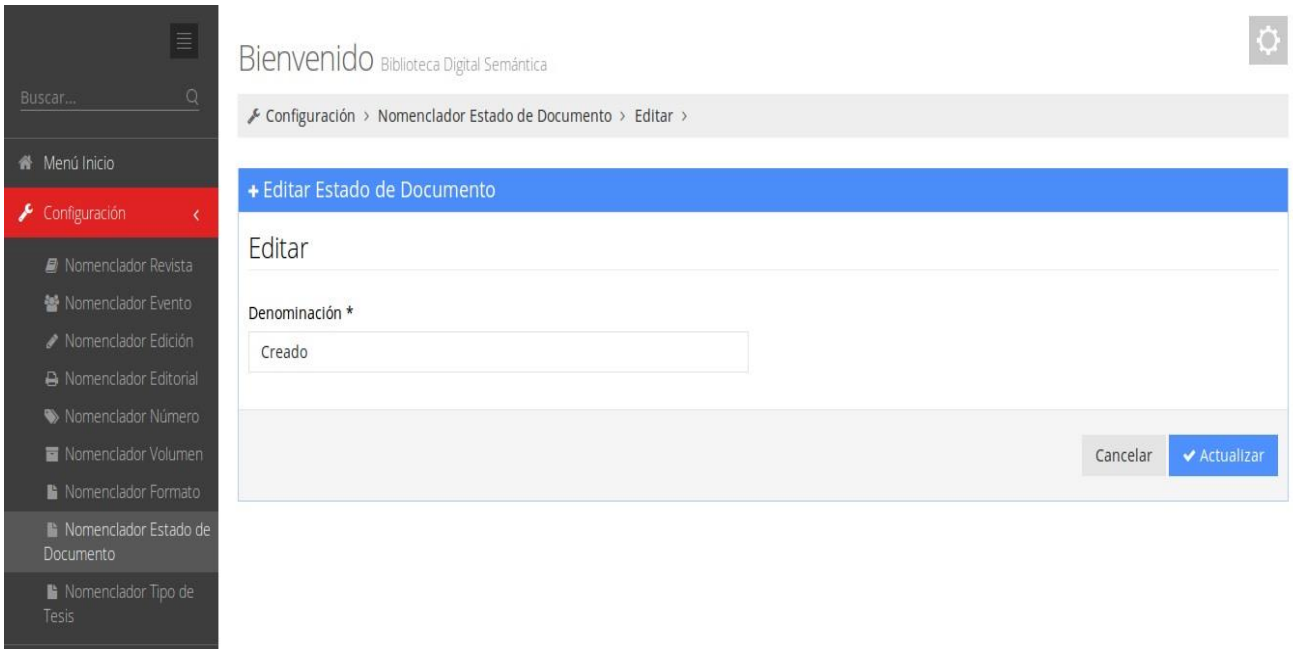
Número: 44	Nombre del requisito: Eliminar estado
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días
Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: En la pantalla principal del nomenclador Estado se deben listar todos los estados que han sido registrados en el sistema con su respectiva denominación.	
Observaciones:	
Prototipo de interfaz: No aplica	

Tabla LXIII: Historia de usuario Mostrar datos del estado.

Número: 45	Nombre del requisito: Mostrar datos del estado
Programador: Paul Nuñez García, Osbel Zorrilla Rivera	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 2 días

Riesgo en Desarrollo: Bajo	Tiempo Real: 1 día
Descripción: El sistema debe permitir al usuario visualizar los datos del estado una vez que este ha sido creado para que verifique si son correctos, de esta forma el usuario podrá identificar si ha cometido un error antes de registrar el estado en el sistema.	
Observaciones:	

Prototipo de interfaz:



2 Casos de prueba para Caja Negra

Tabla LXIV: Caso de prueba CP-06.

Código: CP-06	Historia de Usuario: **HU-27**	
Nombre: Caso de prueba editar evento		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a editar un evento.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Modificar datos del evento.	Nombre: COMPUMAT Siglas: COMPUMAT	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.)

Actualizar datos del evento	COMPUMAT2016 COMPUMAT	1. Los datos del evento deben ser actualizados en la base de datos. 2. Se debe mostrar un mensaje confirmando la actualización. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXV: Caso de prueba CP-07.

Código: CP-07	Historia de Usuario: **HU-29**	
Nombre: Caso de prueba eliminar evento		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar un evento.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Eliminar evento.	Datos del evento que se desea eliminar.	1. Los datos del evento deben ser eliminados en la base de datos. 2. Se debe mostrar un mensaje confirmando la eliminación. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXVI: Caso de prueba CP-08.

Código: CP-08	Historia de Usuario: **HU-2**	
Nombre: Caso de prueba editar revista		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a editar una revista.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Modificar revista	Nombre de la revista: RCCI	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.).

Actualizar datos del evento	Campos en blanco	1. -Los datos de la revista deben ser actualizados en la base de datos. 2. Se debe mostrar un mensaje confirmando la actualización. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXVII: Caso de prueba CP-09.

Código: CP-09	Historia de Usuario: **HU-4**	
Nombre: Caso de prueba eliminar revista		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar una revista.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Eliminar revista	Datos de la revista que se desea eliminar	1. Los datos de la revista deben ser eliminados en la base de datos. 2. Se debe mostrar un mensaje confirmando la eliminación. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXVIII: Caso de prueba CP-10.

Código: CP-10	Historia de Usuario: **HU-11**	
Nombre: Caso de prueba crear editorial		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a crear una editorial.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Inserción de datos de la editorial.	Nombre: @bril País: Cuba	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.).

Crear editorial	Campos en blanco	1. Se debe mostrar un mensaje al usuario informando que se deben completar todos los campos.
Crear editorial	Abril Cuba	1. Se debe mostrar un mensaje indicando la confirmación de la creación del evento. 2. Se actualizan las migajas de pan
Evaluación de la prueba: Satisfactoria		

Tabla LXIX: Caso de prueba CP-11.

Código: CP-11	Historia de Usuario: **HU-7**	
Nombre: Caso de prueba editar editorial		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a editar una editorial.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Modificar datos de la editorial.	Nombre: Abril País: México	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.)
Actualizar datos de la editorial	Abril México	1. Los datos de la editorial deben ser actualizados en la base de datos. 2. Se debe mostrar un mensaje confirmando la actualización. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXX: Caso de prueba CP-12.

Código: CP-12	Historia de Usuario: **HU-9**	
Nombre: Caso de prueba eliminar editorial		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar una editorial.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*

Eliminar editorial.	Datos de editorial que se desea eliminar.	1. Los datos de la editorial deben ser eliminados en la base de datos. 2. Se debe mostrar un mensaje confirmando la eliminación. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXI: Caso de prueba CP-13.

Código: CP-13	Historia de Usuario: **HU-11**	
Nombre: Caso de prueba crear formato		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a crear un formato.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Inserción de datos del formato.	Denominación: Pdf$	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.).
Crear formato	Campos en blanco	1. Se debe mostrar un mensaje al usuario informando que se deben completar todos los campos.
Crear formato	pdf	1. Se debe mostrar un mensaje indicando la confirmación de la creación del formato. 2. Se actualizan las migajas de pan
Evaluación de la prueba: Satisfactoria		

Tabla LXXII: Caso de prueba CP-14.

Código: CP-14	Historia de Usuario: **HU-12**
Nombre: Caso de prueba editar formato	
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a editar un formato.	

Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Modificar datos del formato.	Denominación: Pdf	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.)
Actualizar datos del formato.	pdf	1. Los datos del formato deben ser actualizados en la base de datos. 2. Se debe mostrar un mensaje confirmando la actualización. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXIII: Caso de prueba CP-15.

Código: CP-15	Historia de Usuario: **HU-14**	
Nombre: Caso de prueba eliminar un formato		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar un formato.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Eliminar formato.	Datos del formato que se desea eliminar.	1. Los datos del formato deben ser eliminados en la base de datos. 2. Se debe mostrar un mensaje confirmando la eliminación. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXIV: Caso de prueba CP-16.

Código: CP-16	Historia de Usuario: **HU-16**
Nombre: Caso de prueba crear tipos de tesis	
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a crear un tipo de tesis.	

Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Inserción de datos del tipo de tesis.	Denominación: Maes_tria	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.).
Crear tipo de tesis	Campos en blanco	1. Se debe mostrar un mensaje al usuario informando que se deben completar todos los campos.
Crear tipo de tesis	Maestría	1. Se debe mostrar un mensaje indicando la confirmación de la creación del tipo de tesis. 2. Se actualizan las migajas de pan
Evaluación de la prueba: Satisfactoria		

Tabla LXXV: Caso de prueba CP-17.

Código: CP-17	Historia de Usuario: **HU-17**	
Nombre: Caso de prueba editar tipo de tesis		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a editar un tipo de tesis.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Modificar datos del tipo de tesis.	Denominación: Doctorado	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.)
Actualizar datos del tipo de tesis.	Doctorado	1. Los datos del tipo de tesis deben ser actualizados en la base de datos. 2. Se debe mostrar un mensaje confirmando la actualización. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXVI: Caso de prueba CP-18.

Código: CP-18	Historia de Usuario: **HU-19**	
Nombre: Caso de prueba eliminar tipo de tesis		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar un tipo de tesis.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Eliminar tipo de tesis.	Datos del tipo de tesis que se desea eliminar.	1. Los datos del tipo de tesis deben ser eliminados en la base de datos. 2. Se debe mostrar un mensaje confirmando la eliminación. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXVII: Caso de prueba CP-19.

Código: CP-19	Historia de Usuario: **HU-21**	
Nombre: Caso de prueba crear número de revista		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a crear un número de revista.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Escoger revista	RCCI	1. Se deben listar todas las revistas disponibles.
Escoger volumen	RCCI	1. Se deben mostrar todos los volúmenes disponibles que tiene la revista seleccionada.
Seleccionar número	20	1. Se debe mostrar un mensaje al usuario si el número seleccionado no está disponible. 2. Se actualizan las migajas de pan
Crear número	RCCI 3 1	1. Se debe mostrar un mensaje indicando la confirmación de la creación del número de revista. 2. Se actualizan las migajas de pan
Evaluación de la prueba: Satisfactoria		

Tabla LXXVIII: Caso de prueba CP-20.

Código: CP-20	Historia de Usuario: **HU-22**	
Nombre: Caso de prueba editar número de revista		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a editar un número de revista.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Modificar datos del número de revista.	RCCI 1 1	1. Se debe mostrar un mensaje al usuario si el número seleccionado no está disponible.
Actualizar datos del número de revista.	RCCI 1 2	1. Los datos del número de revista deben ser actualizados en la base de datos. 2. Se debe mostrar un mensaje confirmando la actualización. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXIX: Caso de prueba CP-21.

Código: CP-21	Historia de Usuario: **HU-24**	
Nombre: Caso de prueba eliminar número de revista		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar un número de revista.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Eliminar número de revista	Datos del número de revista que se desea eliminar.	1. Los datos del número de revista deben ser eliminados en la base de datos. 2. Se debe mostrar un mensaje confirmando la eliminación. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXX: Caso de prueba CP-22.

Código: CP-22	Historia de Usuario: **HU-31**	
Nombre: Caso de prueba crear volumen de revista		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a crear un volumen de revista.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Escoger revista	RCCI	1. Se deben listar todas las revistas disponibles.
Crear volumen	20	1. Se debe mostrar un mensaje al usuario si el número seleccionado no está disponible.
Crear volumen	RCCI 3 1	1. Se debe mostrar un mensaje indicando la confirmación de la creación del número de revista. 2. Se actualizan las migajas de pan
Evaluación de la prueba: Satisfactoria		

Tabla LXXXI: Caso de prueba CP-23.

Código: CP-23	Historia de Usuario: **HU-32**	
Nombre: Caso de prueba editar volumen de revista		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a editar un volumen de revista.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Modificar datos del volumen de revista.	RCCI 2016 uno	1. Se debe mostrar un mensaje al usuario si el volumen seleccionado no está disponible. 2. En caso de errores con los datos de entrada debe mostrar al usuario un mensaje informando que datos están incorrectos

Actualizar datos del volumen de revista.	RCCI 2016 2	1. Los datos del número de revista deben ser actualizados en la base de datos. 2. Se debe mostrar un mensaje confirmando la actualización. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXXII: Caso de prueba CP-24.

Código: CP-24	Historia de Usuario: **HU-34**	
Nombre: Caso de prueba eliminar volumen de revista		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar un volumen de revista.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Eliminar volumen de revista	Datos del volumen de revista que se desea eliminar.	1. Los datos del volumen de revista deben ser eliminados en la base de datos. 2. Se debe mostrar un mensaje confirmando la eliminación. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXXIII: Caso de prueba CP-25.

Código: CP-25	Historia de Usuario: **HU-36**	
Nombre: Caso de prueba crear edición.		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a crear una edición.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Escoger evento	COMPUMAT	1. Se deben listar todos los eventos disponibles.

Escoger número	1	1. Se deben mostrar todos los números disponibles teniendo en cuenta el evento seleccionado.
Seleccionar país	Cuba	1. Se deben mostrar todos los países almacenados en la Base de Datos.
Crear edición	Campos en blanco	1. Se debe mostrar un mensaje al usuario informando que se deben completar todos los campos.
Crear edición	ISBN: 1**18	1. Se debe mostrar un mensaje al usuario informando que existen datos incorrectos.
Crear edición	COMPUMAT 123-121-3-325 Cuba 25/5/2015 30/5/2015	1. Se debe mostrar un mensaje indicando la confirmación de la creación del número de revista. 2. Se actualizan las migajas de pan
Evaluación de la prueba: Satisfactoria		

Tabla LXXXIV: Caso de prueba CP-26.

Código: CP-26	Historia de Usuario: **HU-37**	
Nombre: Caso de prueba editar edición		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a editar edición.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Modificar datos del volumen de revista.	COMPUMAT México uno	1. Se debe mostrar un mensaje al usuario si el número seleccionado no está disponible. 2. En caso de errores con los datos de entrada debe mostrar al usuario un mensaje informando que datos están incorrectos

Actualizar datos del volumen de revista.	COMPUMAT México 2	1. Los datos del número de revista deben ser actualizados en la base de datos. 2. Se debe mostrar un mensaje confirmando la actualización. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXXV: Caso de prueba CP-27.

Código: CP-27	Historia de Usuario: **HU-39**	
Nombre: Caso de prueba eliminar edición		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar edición.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Eliminar edición	Datos de la edición que se desea eliminar.	1. Los datos de la edición deben ser eliminados en la base de datos. 2. Se debe mostrar un mensaje confirmando la eliminación. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXXVI: Caso de prueba CP-28.

Código: CP-28	Historia de Usuario: **HU-41**	
Nombre: Caso de prueba crear estado		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a crear un estado de documento.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Inserción de datos del estado.	Denominación: Catalogado$	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres

		extraños, abuso de mayúsculas etc.).
Crear estado	Campos en blanco	1. Se debe mostrar un mensaje al usuario informando que se deben completar todos los campos.
Crear estado	catalogado	1. Se debe mostrar un mensaje indicando la confirmación de la creación del estado. 2. Se actualizan las migajas de pan
Evaluación de la prueba: Satisfactoria		

Tabla LXXXVII: Caso de prueba CP-29.

Código: CP-29	Historia de Usuario: **HU-42**	
Nombre: Caso de prueba editar estado		
Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a editar estado.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Modificar datos del estado.	Denominación: cre@do	1. Se debe mostrar un mensaje indicando a usuario que datos están incorrectos (uso de caracteres extraños, abuso de mayúsculas etc.)
Actualizar datos del estado.	Creado	1. Los datos del estado deben ser actualizados en la base de datos. 2. Se debe mostrar un mensaje confirmando la actualización. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		

Tabla LXXXVIII: Caso de prueba CP-30.

Código: CP-30	Historia de Usuario: **HU-44**
Nombre: Caso de prueba eliminar estado	

Descripción: En este caso de prueba se verifica el procedimiento que se realiza cuando un usuario procede a eliminar estado.		
Acción a probar:	*Datos de entrada:*	*Resultados esperados:*
Eliminar estado	Datos del estado que se desea eliminar.	1. Los datos del estado deben ser eliminados en la base de datos. 2. Se debe mostrar un mensaje confirmando la eliminación. 3. Se actualizan las migajas de pan.
Evaluación de la prueba: Satisfactoria		