Universidad de las Ciencias Informáticas

Facultad 3

Plataforma SOFIA para la distribución electrónica de libros digitales en Cuba

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

Autor(es): Luis Ruben Lima Mateo

Ramón Emanuel Escalona Mayo

Tutor(es): Ing. Vladimir Milán Núñez

Lic. Raynel Batista Téllez

La Habana, junio 2019


"Yo fallé en algunos exámenes, pero mi compañero pasó todo.
Ahora él es un ingeniero de Microsoft y yo soy el dueño de Microsoft"
Bill Gates

DECLARACIÓN DE AUTORÍA

Declaramos por este medio que nosotros: Luis Ruben Lima Mateo y Ramón Emanuel Escalona Mayo, ser autores de la presente tesis que tiene por título: *Plataforma SOFIA para la distribución electrónica de libros digitales en Cuba* 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 _____.

Autores:

_____ _____

Luis Ruben Lima Mateo **Ramón Emanuel Escalona Mayo**

Tutores:

_____ _____

Ing. Vladimir Milán Núñez **Lic. Raynel Batista Téllez**

DATOS DE CONTACTO

Ing. Vladimir Milián Núñez

Ingeniero en Ciencias Informáticas, de la Universidad de las Ciencias Informáticas, 2008. Máster en Ciencias en la Universidad de las Ciencias Informáticas, 2018, Cuba. Profesor Asistente de las disciplinas Física y Matemática Aplicada de la Universidad de las Ciencias Informáticas. Miembro del grupo de Investigación de Inteligencia Artificial y Reconocimiento de Patrones desde el 2008. Líder de proyecto y líder técnico (Arquitecto de software principal) de varios proyectos del centro de desarrollo de Telemática de la UCI en el periodo entre 2008 – 2011.

Miembro de la Asociación Cubana de Reconocimiento de Patrones (ACRP) desde el 2013 y presidente en funciones de la delegación de base de la UCI. Editor Asociado de la Revista Cubana de Ciencias Informáticas. Miembro del comité organizador y del comité científico de varios eventos.

Autor y co-autor de varias publicaciones en revistas y memorias de eventos. Profesor de un curso de postgrado. Tutor de 24 Tesis de pre-grado para Ingeniería.

Lic. Raynel Batista Téllez

Sociólogo, editor, Coordinador General de Ediciones Futuro, sello editorial de la Universidad de las Ciencias Informáticas. Profesor Auxiliar. Ha impulsado proyectos interdisciplinares que vinculan las ciencias sociales y el cálculo computacional, así como las humanidades digitales, en colaboración con actores de América Latina, Europa y Asia.

Ha dirigido varios proyectos en temas como sociocibernética, interacción hombre-computadora y social media. Posee varias publicaciones, participaciones en eventos, proyectos, distinciones y premios. Ha asesorado trabajos de diploma, tesis de maestría y doctorado.

Es miembro de la Junta Nacional Editorial del Ministerio de Cultura, Organización Internacional de Sociología, Sociedad Cubana de Derecho e Informática, Asociación Cubana de Comunicadores Sociales, Organización Internacional de Editores, Junta Editorial de la Revista Cubana de Ciencias Informáticas.

Ha mostrado interés por los nuevos entornos de lectura, las redes académicas de interacción social y la ciencia de los datos. Sus principales resultados se enmarcan en la sociología de la innovación y la antropología digital, con especial interés en la curación de contenidos digitales.

AGRADECIMIENTOS

Luis Ruben Lima Mateo

Mi madre: por ser el amor de mi vida, por su apoyo incondicional en todo momento sin importar las adversidades de la vida, por todo ese amor que solo una madre sabe dar, por ser la mejor madre del mundo.

Mi padre: por ser mi ejemplo y guía a lo largo de este gran camino, por ser mi fiel consejero y ser la persona que supo formarme como hombre para poder ser quien soy hoy.

Mi hermano: por ser mi fiel compañero, por estar ahí a mi lado como un pilar cuando te he necesitado, por ser la persona que a pesar de todo se preocupa por mí y por ser para ti tu mayor voto de confianza.

Mis 2 niñas bellas: dicen que un hijo es la mayor bendición de la vida, pero para mí hasta este momento ustedes han sido lo más maravilloso que he podido disfrutar, gracias por hacerme muy feliz y creer en mí.

Mi cuñada Dainelys: gracias por todo lo bello que hemos compartido y por todo lo que has sido capaz de hacer por mí.

Mi primo Reinier y Aidita: gracias por todos sus consejos y regaños durante estos 5 años y por ayudarme durante esta trayectoria.

Mis primos y familia: por creer en mí y ayudar a que este sueño se hiciera realidad.

La profe Dariela Elvira: quiero darle un agradecimiento muy especial no por ser nuestra guía sino por ser más que eso: ser madre, amiga y consejera durante estos 4 años, por ser la persona que cuando necesitaba ayuda y consejos estaba ahí sin importar el tiempo como si fuera su hijo (muchas gracias de todo corazón, usted es una de las personas más especiales que he conocido en la vida)

Mi familia de hermanos: Yandri, Yadiel, Ginarte, Alfredo, Carlos, David, Alejandro, Kaiser, Mario, Guillermo, Liam, Yoan, Pedro, Ronny, Rolye, Henry, Ary, Maira, Yinet, Marielis, Lili, Hany, Claudia, Niurvis muchas gracias por todos los momentos que compartimos juntos, con ustedes me llevo los mejores recuerdos de mi vida.

Todos mis amigos: muchas gracias por todo lo vivido juntos, sus chistes, las fiestas en general, nunca los olvidare.

Raynel y Vladimir: muchas gracias por todo su apoyo incondicional, por esos momentos que compartimos juntos, por todos los consejos y lecciones dadas. Por sus enseñanzas quiero dedicarle estas frases: "Los grandes objetivos no se logran con dinero ni con talento, se logran con pasión, paciencia y perseverancia"," Si quieres llegar donde la mayoría no llega, debes hacer lo que la mayoría no hace"

Profesores: muchas gracias por todas sus enseñanzas y lecciones, gracias a ustedes puedo ser el profesional que soy hoy.

Mi compañero Ramón: quiero agradecerte por ser un hermano y amigo para mí, por esa persona que estuvo en los momentos buenos y malos sin importar nada, gracias porque sin ti este sueño no se hubiese convertido en realidad.

Ramón Emanuel Escalona Mayo

Hoy, culmina mi etapa como estudiante universitario, etapa que se resumen en largas noches de estudios, sacrificio, sufrimiento, etapas que quedan en mi memoria. Todo lo que alcanzado hubiera sido imposible sin el apoyo de mi familia, de mi madre, de mi padre que hoy no se encuentra aquí, pero él siempre me impulso a convertirme en un gran profesional. También quiero agradecerles a mis tíos en especial Ibrahim, Diana y Daniellys, a mi querida hermana Esther, a mis abuelos Ailsa, Margot y Arturo.

A todos a mis amigos por estar en los momentos difíciles y en los momentos de felicidad a Ronny, Yadiel, Ruben, Daybert, Yandri, Jorge Ernesto, Alejandro, Guillermo, David, Dianabel, Pedro Alejandro, Arisdalia, Asley,Alfredo, Lian, Mileny,Rolando Ernesto,Gisselle,Marcos Antonio, Hany y Mayito.

A todos los profesores que tuve durante estos 5 años en la Universidad, a todos les estoy muy agradecido, y en especial quiero mencionar a dos de ellos, que no solo me ayudaron en el plano académico, también en lo personal, ellos son Dariela y Carlos Cordovi.

Tambien no podía quedarse sin mis agradecimientos Miriam Nicado, quien en la etapa más difícil en la Universidad ella estuvo ahí presente y me brindo todo su apoyo.

A Raynel Batista Tellez y Vladimir Milian Nuñez por ayudarme en la realización de este trabajo.

A mi hermanita como cariñosamente la llamo yo, Indira y su esposo Yunier que fueron los que me iniciaron en el maravilloso mundo de la programación y sin ellos hubiera sido muy difícil salir bien en los exámenes de programación.

DEDICATORIA

Luis Ruben Lima Mateo

A mi madre Ileana, mi padre Carlos, mi hermano Carlos, mis niñas Maria Carla y Ana Carla, mi familia y amigos.

Ramón Emanuel Escalona Mayo

A mi madre Dialá Mayo Chacón, mi padre Ramón Adaberto Escalona Aquilera, mi hermana Esther Escalona Mayo y toda mi familia

Resumen

La distribución electrónica de libros digitales ha demostrado a nivel global su efectividad para promover la lectura y el consumo de obras literarias en un contexto marcado por el uso intensivo de tecnologías digitales. La investigación, cuyos resultados se describen en este informe, tuvo como propósito desarrollar una plataforma cubana para la distribución electrónica de las obras literarias producidas en el país, en respuesta a la escasa visibilidad y posicionamiento que condicionan su comercialización. El empleo de las herramientas y tecnologías seleccionadas para la implementación de la solución propició la correspondencia entre los resultados obtenidos y los esperados. Las aplicaciones de diversas pruebas de software demostraron la efectividad de la solución implementada y su correspondencia con estándares internacionales de calidad, seguridad y usabilidad. La plataforma SOFIA para la distribución electrónica de obras literarias cubanas ofrece mayores niveles de racionalidad y eficiencia en los procesos productivos, es soberana ante ambientes jurídicos de terceros países, incorpora los principios que rigen el proceso de informatización y refrenda la Constitución de la República.

Palabras claves:

Distribución electrónica, libros digitales, plataforma, posicionamiento, visibilidad.

Abstract

The electronic distribution of digital books has proved globally its effectiveness in promoting the reading and consumption of literary works in a context marked by the intensive use of digital technologies. The research, the results of which are described in this report, aimed to develop a Cuban platform for the electronic distribution of literary works produced in the country, in response to the scarce visibility and positioning that condition their commercialization. The use of the tools and technologies selected for the implementation of the solution led to the correspondence between the results obtained and those expected. The applications of various software tests demonstrated the effectiveness of the implemented solution and its correspondence with international standards of quality, security and usability. The SOFIA platform for the electronic distribution of Cuban literary works offers higher levels of rationality and efficiency in production processes, is sovereign before legal environments of third countries, incorporates the principles that govern the process of computerization and endorses the Constitution of the Republic.

Keywords:

Digital books, electronic distribution, platform, visibility, positioning.

ÍNDICE

INTRODUCCIÓN

La producción de libros en Cuba mantiene un alto reconocimiento internacional y dentro de la política cultural de la Revolución dada su importancia como sostén de los planes educacionales y la formación de un lector crítico, aspiración consecuente con los propósitos de construir una sociedad socialista bajo la premisa martiana “leer es crecer”.

Las Ferias del Libro representan el mecanismo de promoción literaria con mayor tradición e impacto en el país mediante el cual se distribuyen más del 80% de las obras literarias cubanas. En la edición del 2018 se presentaron más de 600 novedades literarias y más de cuatro mil títulos, y fueron comercializados cerca de 2 millones de ejemplares, lo cual representó 10,6% más que en la edición anterior y confirmó el interés por la lectura. (1)

Las Ferias del Libro han marcado un hito en el mundo de la lectura de nuestro pueblo, demostrando que el pueblo cubano es gran consumidor de obras literarias. A pesar de esto es el único evento literario de magnitud con el que cuenta por lo que fuera de este espacio es de gran dificultad obtener algunos ejemplares. Sin embargo, no se descarta la influencia que pueden tener los bajos precios en estas cifras puesto que toda la producción de libros en Cuba es subsidiada por el Estado. El mismo destina millones de dólares para lograr que la literatura y los conocimientos sean accesibles para todo el pueblo sin importar sus ingresos ya que los mismos son vendidos a un precio muy por debajo de su coste de producción. (1)

Además, la impresión de los libros no se hace bajo demanda, sino mediante estadística, lo cual trae consigo que en ocasiones se impriman muchos ejemplares de un libro y sobren, y otros ejemplares falten por su poca impresión. Los libros más antiguos son difíciles de encontrar y en ocasiones no se encuentran en ninguna editorial.

El país cuenta con numerosas librerías, pero la distribución de los ejemplares por las mismas se hace por estadística y casi ninguna librería, ni editorial tiene página de cara a Internet y la tecnología utilizada en las mismas es obsoleta y el proceso de impresión de los libros es caro y largo.

En un contexto nacional marcado por la transformación de la organización de la economía, la distribución de libros también ha de cambiar para ser sostenible. Esto se ve reflejado en los principios asociados a la actualización del modelo económico cubano respaldado también en el proyecto de Reforma Constitucional:” El Estado tiene como fines esenciales los siguientes: asegurar el desarrollo educacional, científico, técnico y cultural del país”. (2) La industria editorial cubana, apoyada en alto grado por el Estado, necesita funcionar con mayores niveles de racionalidad y eficiencia en sus procesos de producción y distribución.

El surgimiento de Internet marcó un hito histórico en el desarrollo de la sociedad y la tecnología de la información. Desde entonces han ocurrido profundas transformaciones en la industria editorial y en los diversos puntos de la cadena tradicional de producción del libro.

El desarrollo científico y técnico actual ha permitido automatizar los procesos de distribución y comercialización de contenido digital, ejemplo de ello son las plataformas Amazon, Libranda, Edigita, Libreka, entre otras. El uso de las mismas se ve limitado por los ambientes jurídicos de terceros países y por la conectividad.

La mayor transformación que ha experimentado la industria editorial ha sucedido en la era de Internet, cuando las empresas se vieron forzadas a integrarse a la cibersociedad. Esto se debe al intercambio de información, de forma rápida y cómoda entre dos o más personas, que se ha desarrollado desde un ambiente más estático a uno más dinámico y social.

El mundo de la comunicación, fundamentalmente en su vertiente cultural, se presenta bajo el paradigma digital, que en la actualidad domina Internet. El desarrollo constante de las publicaciones electrónicas y su comercialización puede considerarse como una de las manifestaciones más evidentes de la irrupción de las TIC en el sector editorial. La llegada de las publicaciones electrónicas, supone una apertura de posibilidades para el proceso editorial.

La conectividad dejará de ser un mito también en Cuba mediante el avance de la informatización de la sociedad, especialmente las telecomunicaciones están marcando el sendero del salto tecnológico, pues nuestro país fue el de mayor desarrollo de la Internet en América Latina en el 2017. Las tecnologías han cambiado los entornos de lectura, hay un nuevo lector, la mayor parte de la población que está compuesta por jóvenes y niños tienen acceso a dispositivos móviles y tiene cerca de 2 millones de líneas celulares y más de 5 mil conexiones diarias lo cual debe ser explotado al máximo. (3)

El país cuenta con aplicaciones como Todus, Banca Móvil, Kalis, las cuales están marcando una nueva “era de la Internet en Cuba”.

La universidad no está distante de los avances tecnológicos y editoriales, la misma dispone del sello editorial “Ediciones Futuro Cuba”, reconocido como miembro del Consejo Nacional Editorial del Instituto Cubano del Libro (el cual cuenta con 180 sellos editoriales y más de 15 mil títulos), para promover la cultura científica a través de publicaciones de alto impacto, así como el registro y la comercialización de resultados de la ciencia. (4)

Debido a esto surge la necesidad de que el país cuente con un sistema propio para la distribución digital del libro el cual sea utilizado además como repositorio. Con el objetivo de que puedan publicarse todas

las obras, artículos y revistas con la que cuenta el país para que esté accesible para todo el pueblo desde cualquier rincón del país.

Luego de analizar la situación problemática se identifica el siguiente **problema a resolver**:

El modo en que se distribuyen las obras literarias producidas por las editoriales cubanas limita la visibilidad y posicionamiento de sus contenidos.

Se plantea como **objeto de estudio** las herramientas para la promoción de obras literarias en Cuba centrando su campo de acción en la distribución de obras literarias cubanas.

Para darle solución al problema anterior se plantea como **objetivo general** desarrollar una plataforma para la distribución electrónica de obras literarias cubanas que contribuya a la visibilidad y posicionamiento de su contenido.

Proponiendo como **idea a defender**: si se desarrolla una plataforma para la distribución electrónica de obras literarias producidas en Cuba se contribuirá a la visibilidad y posicionamiento de sus contenidos.

Desglosándose en los siguientes **objetivos específicos**:

- Caracterizar los procesos de negocio y soluciones tecnológicas relacionadas con la distribución electrónica de libros digitales.
- Diseñar los artefactos e implementar la propuesta de solución para la distribución electrónica de libros digitales en Cuba.
- Validar la solución informática propuesta aplicando diferentes pruebas.
- Para dar cumplimiento al objetivo planteado se proponen las siguientes tareas de la investigación:
- Análisis de los principales conceptos, contribuciones y tendencias sobre la distribución electrónica de literatura.
- Identificación de las tecnologías, herramientas y técnicas disponibles para la distribución electrónica de literatura mediante soluciones de software.
- Definición de los aspectos que distinguirán la propuesta de solución.
- Implementación de la solución informática.
- Evaluación técnica de la propuesta de solución.

El **resultado esperado** es obtener una Plataforma web para la distribución digital de libros en Cuba, la cual contribuirá a la evolución del proceso de posicionamiento y distribución de las obras literarias cubanas.

Los **Métodos teóricos** utilizados para cumplir las tareas son los siguientes:

- Histórico-Lógico: Permitió realizar un estudio de los principales conceptos asociados a la distribución de contenido digital.
- Hipotético-deductivo: se utilizó para guiar la investigación desde el planteamiento del problema hasta la verificación de la solución a partir de las validaciones, orientando la secuencia lógica de las tareas que se realizaron.
- Análisis y Síntesis: Se utiliza para identificar y analizar las diversas funcionalidades de las distribuidoras de libro a nivel internacional que pueden ser aplicadas en la solución.
- Modelación: Permitió la creación del modelo conceptual para entender el contexto en el que se enmarca la investigación.

Los **Métodos empíricos** utilizados para cumplir las tareas fueron:

- Entrevista: Permitió obtener la información necesaria relacionada con los problemas presentes en las editoriales cubanas y en la industria del libro e general.

El presente documento se estructura en **tres capítulos**:

Capítulo 1. Fundamentación teórica. En este capítulo se definió algunos conceptos importantes para la posterior comprensión de la investigación. Se realizó una breve reseña y valoración sobre las soluciones existentes de los sistemas de distribución y venta de libros digitales, y se explican las herramientas, metodologías y lenguajes que serán utilizadas en la construcción de la solución.

Capítulo 2. Análisis y diseño de la plataforma SOFIA. En este capítulo se definió la solución que se propone y se realizó la selección de los requerimientos del sistema que se pretende implementar. Se hizo el análisis de la solución que se propone, se modelan y describen los recursos ingenieriles necesario acorde a la metodología XP que representan las funcionalidades del sistema, aplicando los patrones de arquitectura y diseño seleccionados.

Capítulo 3. Evaluación de la solución propuesta. En este capítulo se describió la implementación y posterior validación realizada al producto obtenido como solución.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En este capítulo se expone la fundamentación teórica del trabajo, incluyéndose en el mismo el estudio de otras soluciones informáticas existentes en la actualidad, de las metodologías, modelos de desarrollo de software y herramientas a utilizar.

1.1 Conceptos asociados

Los conceptos asociados son de vital importancia para una mejor comprensión del negocio, para ello se tuvo en cuenta los conceptos de libro, libro digital, editorial, repositorio

1.1.1 Libro

Es una obra impresa, manuscrita o pintada en una serie de hojas de papel, pergamino, vitela u otro material, unidas por un lado (es decir, encuadernadas) y protegidas con tapas, también llamadas cubiertas. Un libro puede tratar sobre cualquier tema. (5)

También se llama "libro" a una obra de gran extensión publicada en varias unidades independientes, llamados "tomos" o "volúmenes". Otras veces se llama también "libro" a cada una de las partes de una obra, aunque físicamente se publiquen todas en un mismo volumen.

1.1.2 Editorial

Es una empresa que se dedica económicamente a la publicación de materiales impresos como libros, revistas, periódicos, folletos, mapas y póster. Ya sea utilizando cualquier medio de impresión y reproducción gráfica o bien medios electrónicos. (6)

1.1.3 Libro digital

Un ebook o e-book es un anglicismo que, traducido al español, significa libro electrónico, libro digital o ciberlibro. Como tal, la palabra es un neologismo del inglés, compuesto por “e”, inicial de electronic, y book, que traduce ‘libro’. De allí que ebook sea el nombre con que han venido denominando los libros que se encuentran en formato digital, es decir, la versión electrónica del libro de papel. (6)

Hoy en día, el ebook es ya una realidad editorial, pero requiere programas especiales para su lectura. En este sentido, para visualizar un ebook en la pantalla de un dispositivo electrónico es necesario instalar una aplicación que permita leer el formato en que el libro se encuentre. (6)

1.1.4 Repositorio

Es un espacio centralizado donde se almacena, organiza, mantiene y difunde información digital, habitualmente archivos informáticos, que pueden contener trabajos científicos, conjuntos de datos o software. (6)

Luego de analizar los principales conceptos se pudo entender con mayor profundidad el proceso de negocio que se quiere desarrollar en función de la problemática identificada y el contexto de uso.

1.2 Análisis de soluciones existentes

En el mundo, principalmente en Estados Unidos, América Latina y Europa, existen diferentes sistemas o herramientas que permiten dar solución a diversos problemas relacionados con la distribución de libros digitales.

El software libre es aquel que ofrece la libertad de utilizar, compartir, estudiar y modificar el código del programa. Al utilizar software libre, los recursos que se dejan de gastar en licencias pueden ser dedicados a personalizar la solución con total libertad y a integrarlo con otras herramientas informáticas. Esta adaptabilidad es una de las mayores ventajas del software libre frente al software privativo, cuyo código, cerrado y rígido por definición, no admite modificaciones significativas. (7)

El coste del software propietario tiende a ser considerablemente mayor que el de las soluciones de código abierto, por lo cual merece la pena intentar buscar opciones libres que además garanticen una gran calidad para los procesos de gestión documental por su capacidad multiplataforma y su sencillo manejo. (8)

Las características tomadas en cuenta para el análisis fueron: formato de los libros, sistema operativo o dispositivo para su lectura, compra de libros o pago por utilizar el sistema y tipo de software.

A continuación, se expone el análisis de los sistemas homólogos a nivel internacional:

1.2.1 Amazon

Es la más conocida y la más popular plataforma de distribución electrónica para libros. No solamente pueden comprar sus libros aquellos que tengan un dispositivo de lectura como los populares Kindle, sino también quienes utilicen las aplicaciones Amazon Kindle, disponibles en cualquier tableta o teléfono inteligente. (9)

Amazon cuenta con las herramientas Kindle Direct Publishing para publicar la versión digital del libro y Create Space para la edición en papel bajo demanda. (9)

Existen dos franjas de precio que determinan las regalías que paga Amazon por la venta de libros digitales.

Si el precio del libro se fija entre 2,60 y 9,70 euros, las regalías son del 70%. Además, para estos libros a los cuales les corresponde este porcentaje de regalías, Amazon aplica una comisión por cada transferencia del libro digital. Esta es igual a 0,12 euros por cada MB de tamaño del libro y se descuenta

antes de calcular las regalías. Es decir que ese 70% se debe calcular sobre el precio de venta, menos el coste por transferencia.

Libros con un precio menor o superior al rango anterior, o ventas realizadas en países no incluidos para el cobro del 70%, corresponden regalías del 35%. En este caso no se aplica la comisión por transferencia.

1.2.2 Bubok

Es una empresa española que se define a sí misma como editorial online. Esta empresa permite a los escritores la publicación de sus obras para ponerlas a la venta tanto en versión libro digital (formato ePub y PDF) como papel en la modalidad de impresión bajo demanda. Además, ofrece un mercado potencial amplio de lectores. (10)

En el caso de los libros impresos bajo demanda al precio de venta al público hay que descontarle los costes de impresión y distribución para determinar la cantidad de beneficio que ese libro ha generado. Sobre ese beneficio, Bubok cobra una comisión del 20%, por lo que las regalías son del 80%. (10)

En el caso del formato digital, se calcula directamente el 80% de regalías sobre el precio de venta.

1.2.3 Google Play

Es la tienda de aplicaciones, música y libros a la cual acceden los millones de usuarios del sistema operativo Android. Este sistema operativo es el que se utiliza mayoritariamente en teléfonos inteligentes y tabletas, lo que el mercado potencial es gigante. Sin embargo, en cuanto a libros digitales no parece ser en la actualidad un referente para los compradores. (11)

Es bastante difícil encontrar la información acerca de las regalías que se pagan por las ventas de libros a través de Google Play. Algunos escritores que han publicado en esta plataforma afirman que las regalías oscilan entre el 52 y 55%. (11)

1.2.4 ibookStore

Se trata de la tienda de venta de libros de Apple por lo cual su mercado potencial son todos los usuarios de iphones, ipods touch, ipads y ordenadores Mac. Además, otro de los requisitos para poder publicar es descargar la aplicación itunes Producer para la cual se requiere un Mac con OS X 10.8 o posterior.

En numerosas publicaciones se menciona que el porcentaje de regalías es del 70%. (12)

1.2.5 NookPress

Es la herramienta de auto-publicación del gigante de la venta de libros, Barnes & Noble. Por lo tanto, el mercado potencial es también de millones de lectores. Los libros podrán comprarse a través de la web de B&N, todos los dispositivos de lectura Nook o la aplicación homónima en tabletas y teléfonos inteligentes. (13)

Al igual que Amazon, las regalías dependen de la franja donde se encuentre el precio de venta del libro:

Entre 2,50 y 9,49 euros las regalías son del 60%.

Fuera de ese rango, regalías del 40%.

1.2.6 Grammata

La empresa española Grammata es el fabricante de los dispositivos de lectura electrónica Papyre. Esta empresa ofrece a los escritores independientes la posibilidad de publicar sus libros y venderlos en Argentina, México, Colombia y España. Este canal se caracteriza por su especialización en el mercado de habla hispana. (14)

Los beneficios que se reparten entre la plataforma y los autores son a partes iguales por lo que las regalías que son del 50%.

A continuación, se muestra una tabla comparativa de los sistemas detallados anteriormente, ver tabla 1:

Tabla 1 Comparación sistemas homólogos

Plataforma	Formato libro	Tipo de software	Pago por libro	Dispositivo o sistema
Amazon	Ebook	Privativo	Pago por consumo	Kindle, las aplicaciones Amazon Kindle.
Google play	Ebook	Privativo	Pago por consumo	Sistema operativo Android
IbookStore	Ebook	Privativo	Pago por consumo	iphones, ipods touch, ipads y ordenadores Mac.
Grammata	Ebook	Privativo	Pago por consumo	Cualquier sistema operativo o dispositivo.

NookPress	Ebook	Privativo	Pago por consumo	Dispositivos de lectura Nook o la aplicación homónima en smartphones y tablets.
Bubok	formato ePub y PDF	Privativo	Pago por consumo	Cualquier sistema operativo o dispositivo.

Luego de analizados los diferentes sistemas se llegó a la conclusión de que el sistema a desarrollar debe ser capaz de utilizarse en cualquier sistema operativo o dispositivo, las herramientas utilizadas para su desarrollo deberán ser de código abierto, lo cual cumple con lo planteado en la Constitución de la República en cuanto a la informatización del país.

1.3 Metodología de desarrollo de software

En el presente trabajo se realizó un estudio para la determinación de la metodología de desarrollo de software a utilizar en el desarrollo del sistema, donde se obtuvo lo siguiente:

Piattini define a la metodología de desarrollo de software como “un conjunto de procedimientos, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software”. (15)

La metodología indica cómo hay que obtener los diferentes productos parciales y finales.

Las metodologías se clasifican en 2 grupos: ágiles y tradicionales. A continuación, se muestran algunas características:

Tabla 2 Comparación metodologías de software (16)

Metodologías ágiles	Metodologías tradicionales
Preparadas para cambios en el proyecto.	Cierta resistencia a los cambios.
No existe un contrato tradicional o al menos es bastante flexible.	Existe un contrato prefijado.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.

Grupos pequeños(menos de 10 integrantes) y trabajando en el mismo sitio.	Grupos grandes y posiblemente distribuidos.
Pocos artefactos.	Muchos artefactos.
Pocos roles.	Muchos roles.

1.3.1 Programación extrema o XP

Es una metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. XP se basa en retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. (17)

Véase en la Ilustración 1 el ciclo de vida de XP.

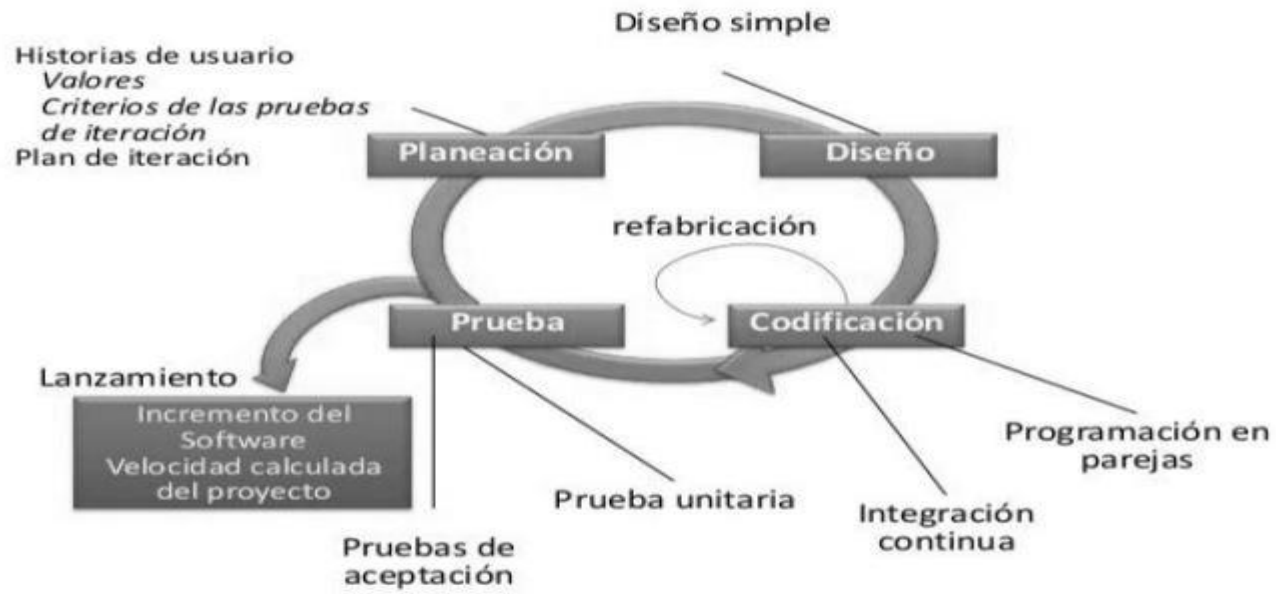

Ilustración 1 Ciclo de vida de XP. *(17)*

El ciclo de vida ideal de XP consiste de seis fases:

Fase I: Exploración

En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. (18)

Fase II: Planificación de la Entrega

En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días. (17)

Fase III: Iteraciones

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura, sin embargo, esto no siempre es posible, ya que es el cliente quien decide qué historias se implementarán en cada iteración (para maximizar el valor de negocio). Al final de la última iteración el sistema estará listo para entrar en producción. (17)

Fase IV: Producción

La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase.

Es posible que se rebaje el tiempo que toma cada iteración, de tres a una semana. Las ideas que han sido propuestas y las sugerencias son documentadas para su posterior implementación (por ejemplo, durante la fase de mantenimiento). (17)

Fase V: Mantenimiento

Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la

puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura. (17)

Fase VI: Muerte del Proyecto

Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo. (17)

1.4 Lenguajes

1.4.1 Lenguaje de modelado

El lenguaje de modelado a utilizar será UML, el cual es un lenguaje diseñado para visualizar, especificar, construir y documentar software orientados a objeto, ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados. Se utiliza para especificar o para describir métodos o procesos, para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. (19)

1.4.2 Marco de trabajo

Symfony es un completo marco de trabajo diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Este marco de trabajo se distingue por separar la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Asimismo, proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. (20)

Symfony está desarrollado completamente con PHP 5 y ha sido probado en numerosos proyectos reales, especialmente en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Entre las principales características de este marco de trabajo destaca su capacidad para ejecutarse tanto en plataformas Unix, como en plataformas Windows. A continuación, se muestran algunos de sus beneficios: (20)

- Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del sistema gestor de bases de datos.

- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Sigue la mayoría de las buenas prácticas y patrones de diseño para la web.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

Symfony 3.4 constituye una perfecta elección para el desarrollo de la solución informática que se desea desarrollar, hecho que responde a las características y buenas prácticas de diseño que hacen de este marco de trabajo líder indiscutible en la implementación de aplicaciones web con PHP. (20)

Symfony provee características y filosofías fundamentales para el desarrollo de una plataforma, de la cual la solución que se desea desarrollar constituye un núcleo base de vital importancia. En ese sentido, este marco de trabajo al poseer gran cantidad de complementos, así como una numerosa y activa comunidad, se convierte en una excelente opción para desarrollar este tipo de aplicaciones.

1.5 Herramientas utilizadas para el desarrolla del sistema

1.5.1 Herramienta de modelado

Las herramientas de modelado de sistemas informáticos, son herramientas que se emplean para la creación de modelos de sistemas que ya existen o que se desarrollarán, permitiendo crear un modelo del sistema a bajo costo y riesgo mínimo.

Las buenas herramientas de modelado cumplen con las siguientes características:

- Permiten una visión descendente del sistema.
- Poseen componentes gráficos con algo de apoyo textual.
- El modelo resultante debe ser transparente (fácil de comprender).
- Poseen mínima redundancia (el aumento de redundancia, disminuye la transparencia del modelo y aumenta las tareas de mantenimiento). (21)

La herramienta de modelado seleccionada fue Visual Paradigm para UML en su versión 8.0, debido a que es una herramienta para el desarrollo de aplicaciones utilizando modelado UML. Esta herramienta es ideal para ingenieros de software, analistas de sistemas y arquitectos de sistemas que están interesados en la construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos. (19)

Visual Paradigm ha sido concebido para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas y constituye una herramienta disponible en varias ediciones. (21)

Se caracteriza por:

- Disponibilidad en múltiples plataformas (Windows, Linux).
- Diseño centrado en casos de uso.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Licencia: gratuita y comercial.
- Varios idiomas.
- Generación de código para Java y exportación como HTML.
- Diagramas de Procesos de Negocio - Proceso, Decisión, Actor de negocio, Documento.
- Generación de código - Modelo a código, diagrama a código.
- Editor de detalles de casos de uso - Entorno todo-en-uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso.
- Soporte ORM - Generación de objetos Java desde bases de datos.
- Generación de bases de datos - Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- Distribución automática de diagramas - Reorganización de las figuras y conectores de los diagramas UML.
- Editor de figuras.

La utilización de esta herramienta permite aumentar la calidad del software, a través de la mejora de la productividad en el desarrollo y mantenimiento del software, así como aumenta el conocimiento informático de una empresa ayudando así a la búsqueda de soluciones para los requisitos. También permite la reutilización del software, portabilidad y estandarización de la documentación, además del uso de las distintas metodologías propias de la Ingeniería de Software. (17)

1.6 IDE

Un entorno de desarrollo integrado (IDE, por sus siglas en inglés), es una aplicación de software, que proporciona servicios integrales para facilitarle al programador de computadora el desarrollo de software. Normalmente, un IDE consiste de un editor de código fuente, herramientas de construcción automáticas y un depurador. La mayoría de los IDEs tienen auto-completado inteligente de código. (22)

Para la implementación del sistema se tuvo en cuenta el Netbeans y PHPStorm.

1.6.1 Netbeans

Es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto libre y gratuito sin restricciones de uso. NetBeans es un proyecto de código abierto una comunidad en constante crecimiento. (23)

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs[1] de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software. (23)

El NetBeansIDE es un IDE de código abierto escrito completamente en Java usando la plataforma NetBeans. Soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Todas las funciones del IDE son provistas por módulos. Cada módulo provee una función bien definida, tales como el soporte de Java, edición, o soporte para el sistema de control de versiones. NetBeans contiene todos los módulos necesarios para el desarrollo de aplicaciones Java en una sola descarga, permitiéndole al usuario comenzar a trabajar inmediatamente. (23)

La plataforma ofrece servicios reusables comunes para las aplicaciones de escritorio, permitiendo a los desarrolladores centrarse en la lógica de sus aplicaciones. Algunas de las características de la aplicación son:

- Gestión de la interfaz de usuario (menús y barras de herramientas).
- Gestión de configuración de usuario.
- Gestión de almacenamiento (guardar o cargar algún tipo de dato).
- Gestión de ventana.
- Marco Asistente.
- Biblioteca de clases.

[1] API: La interfaz de programación de aplicaciones (API, por sus siglas en inglés), es el conjunto de subrutinas, funciones, procedimientos o métodos, en la programación orientada a objetos, que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

- Herramientas de desarrollo integrado.

NetBeans IDE es libre, de código abierto, multiplataforma con soporte integrado para el lenguaje de programación Java.

1.6.2 PhpStorm

PhpStorm es perfecto para trabajar con Symfony, Laravel, Drupal, WordPress y otros marcos de trabajo.
El editor "obtiene" su código y comprende profundamente su estructura, y es compatible con todas las funciones del lenguaje PHP para proyectos modernos y antiguos. Proporciona la mejor terminación de código, refactorizaciones, prevención de errores sobre la marcha y más.
Aprovecha al máximo las tecnologías de vanguardia, como HTML 5, CSS, Sass, Less, Stylus, CoffeeScript, TypeScript, Emmet y JavaScript, con refactorizaciones, depuración y pruebas de unidad disponibles. (24)
Realiza muchas tareas de rutina directamente desde el IDE, gracias a la integración de Version Control Systems, soporte para implementación remota, bases de datos / SQL, herramientas de línea de comandos, Docker, Composer, REST Client y muchas otras herramientas.
Todas las funciones de WebStorm están incluidas en PhpStorm, con soporte completo para PHP y soporte de bases de datos / SQL agregado en la parte superior. (24)

PhpStorm es conocido por su depurador visual de configuración cero, que proporciona una visión extraordinaria de lo que sucede en su aplicación en cada paso. Funciona con Xdebug y Zend Debugger, y puede usarse tanto de forma local como remota. También están disponibles las pruebas unitarias con PHPUnit, BDD con Behat y la integración del generador de perfiles. (24)

1.6.3 Selección del IDE

Para el desarrollo de la plataforma se decidió utilizar el PHPStorm ya que el editor entiende profundamente su estructura, apoya a todas las características del lenguaje PHP para proyectos modernos y antiguos. Proporciona la mejor finalización de su código, refactorizaciones sobre la marcha de la prevención de errores. El soporte PHPDoc, código de (re) arranger y formateo, soluciones rápidas, y otras características contribuyen a escribir un código que será fácil de mantener. Además, realiza muchas tareas rutinarias desde el IDE, gracias a Versión Control System integration, el apoyo a la implementación remota, base de datos/SQL, REST Client. También permite ver los cambios realizados en el front end al instante en el navegador. Otra de las razones por las cuales se utilizará es que permite la realización de pruebas unitarias en el mismo IDE utilizando PhpUnit.

1.7 Herramientas de gestión de base de datos

Un sistema de gestor de bases de datos (DBMS 15), algunas veces llamada simplemente un administrador de base de datos, es un programa que permite a uno o más usuarios de computadoras crear y acceder a los datos en una base de datos. El DBMS gestiona las peticiones del usuario para que estos sean libres de entender que los datos se encuentran físicamente en los medios de almacenamiento. En el manejo de solicitudes de los usuarios, el DBMS asegura la integridad de los datos y seguridad (asegurándose de que sólo aquellos con privilegios de acceso pueden acceder a los datos). El más típico DBMS es un sistema de base de datos relacional. La selección adecuada del DBMS proveerá a la solución informática que se desea desarrollar numerosas herramientas, además de seguridad a los datos que se almacenan y mejorará considerablemente el desempeño de la aplicación final. (25)

1.7.1 MySQL

Es el más popular sistema de gestión de base de datos multiplataforma de código abierto. Se caracteriza fundamentalmente por su velocidad, rendimiento y fiabilidad, motivo por el cual es ampliamente utilizado en sitios web y aplicaciones de escritorio de baja y mediana complejidad. (25)

Además, cumple con las siguientes características:

- MySQL es de código abierto.
- Velocidad al realizar las operaciones, lo que le hace uno de los gestores con mejor rendimiento.
- Bajo costo en requerimientos para la elaboración de bases de datos, ya que debido a su bajo consumo puede ser ejecutado en una máquina con escasos recursos sin ningún problema.
- Facilidad de configuración e instalación.
- Soporta gran variedad de Sistemas Operativos.
- Baja probabilidad de corromper datos, incluso si los errores no se producen en el propio gestor, sino en el sistema en el que se encuentra integrado.
- Ofrece un sistema de contraseñas y privilegios seguros de verificación basada en el host y tráfico de contraseñas encriptado al conectarse a un servidor.
- Soporta gran cantidad de datos, incluso con más de 50 millones de registros.

1.8 Conclusiones del capítulo

Los resultados expuestos en este capítulo permitieron:

- El análisis de los conceptos fundamentales asociados al proceso de distribución electrónica de literatura permitió identificar la visibilidad y el posicionamiento como aspectos fundamentales en

soluciones informáticas para la promoción y comercialización de contenidos literarios. Sin embargo, el contexto de uso continúa siendo el criterio que distingue la efectividad de cada solución estudiada.

- El estudio comparativo de soluciones similares para la distribución electrónica de literatura resaltó la normalización, organización, clasificación, segmentación, disponibilidad e interacción de los contenidos literarios como características más comunes, mientras que el uso de tecnologías privativas y dispositivos a la medida destacan los fines de lucro y expansión de las empresas distribuidoras a nivel global y por tanto, sus principales limitaciones para el desarrollo.
- El análisis de las características que distinguen a los sistemas de distribución de literatura estudiados y su relación con la promoción literaria en el contexto cubano permitió reorientarlas a la industria cubana del libro y las premisas del proceso de informatización de la sociedad cubana.
- La identificación de las características propuestas para la solución, el análisis de las tecnologías que distinguen a soluciones similares para la distribución electrónica de literatura, contribuyó a la selección de las herramientas y tecnologías más adecuadas para desarrollar la propuesta de solución, destacando como metodología de desarrollo XP, como marco de trabajo Symfony y como herramienta de modelado Visual Paradigm usando como lenguaje UML.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA PLATAFORMA SOFIA

En este capítulo se aborda la propuesta de solución a la problemática planteada. Se describen los requisitos funcionales y no funcionales, para dar cumplimiento a los objetivos planteados. Se realizan las historias de usuario, plan de iteraciones, tarjetas CRC y demás artefactos exigidos por la Metodología XP y sus fases.

2.1 Fase 1: Exploración

Se cumple con lo establecido en la **Fase I** de la Metodología XP, donde el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología. (16)

2.1.1 Modelo conceptual

Un modelo conceptual en UML está constituido por un conjunto de conceptos y las relaciones que se establecen entre ellos. Este es un paso muy importante en cualquier metodología de análisis y diseño orientada a objetos, pues de aquí parte el diagrama de clases que modela la aplicación. Los errores que se cometan en la modelación de este diagrama pueden dar al traste con una aplicación incompleta o que no cumpla las expectativas de los clientes. (26)

El sistema cuenta con un perfil para cada rol del sistema (editoriales, autores, lectores), los cuales pueden interactuar con las funcionalidades del mismo acorde a su rol, las principales funcionalidades están enmarcadas, en las editoriales, la publicación de libros, agrupación por géneros literarios, y publicaciones de las editoriales o autores.

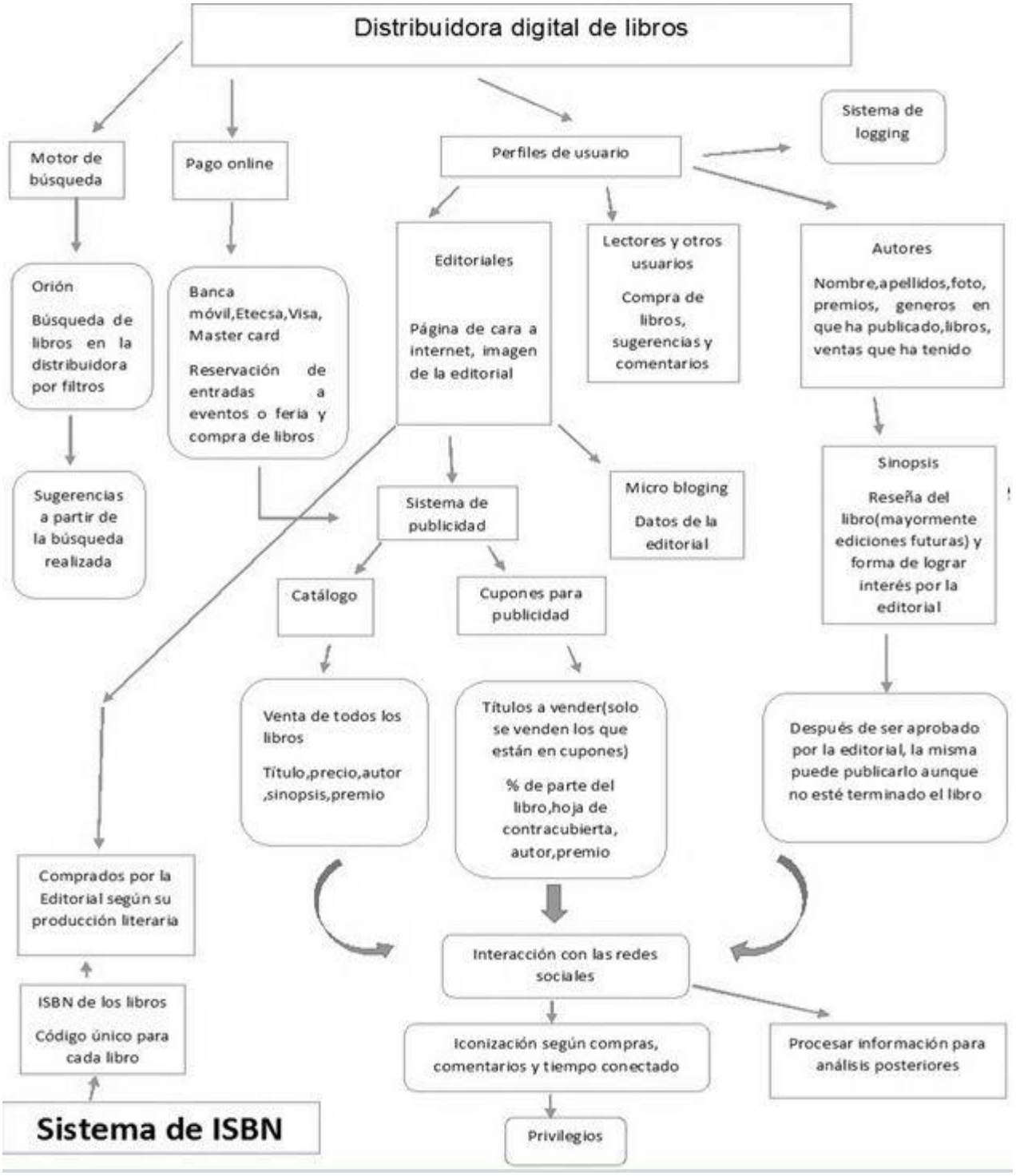

Ilustración 2 Modelo conceptual

2.1.2 Determinación de requisitos

El análisis de requisitos es una de las tareas más importantes en el ciclo de vida del desarrollo de software, puesto que en ella se determinan los "planos" de la nueva aplicación. El análisis de requisitos se puede definir como el proceso del estudio de las necesidades de los usuarios para llegar a una definición de los requisitos del sistema, hardware o software, así como el proceso de estudio y refinamiento de dichos requisitos (27).

2.1.3 Técnica de captura de requisitos

En el proceso de desarrollo de software los analistas emplean varias técnicas para obtener los requisitos del cliente. Históricamente, esto ha incluido técnicas tales como las entrevistas, o talleres con grupos para crear listas de requisitos. Cuando sea necesario, el analista empleará una combinación de estos métodos para establecer los requisitos exactos de las personas implicadas, para producir un sistema que resuelva las necesidades del cliente. A continuación, se muestran algunas de estas técnicas:

2.1.3.1 Entrevista

Las entrevistas son un método común. Por lo general no se entrevista a toda la gente que se relacionará con el sistema, sino a una selección de personas que represente a todos los sectores críticos de la organización, con el énfasis puesto en los sectores más afectados o que harán un uso más frecuente del nuevo sistema.

2.1.3.2 Talleres

2.1.3.2.1 Lluvia de ideas

Todos los participantes pueden aportar distintas ideas en un ambiente libre de prejuicios. Ningún participante debe juzgar negativamente la propuesta de otros, sino que se anotan todas las ideas en una pizarra y serán evaluadas al final de la sesión. El principio básico es no descartar de manera apresurada ningún planteo, de modo que existe la posibilidad de que surjan otras ideas derivadas de la idea original y se generan varios puntos de vista del problema.

2.1.3.2.2 Diseño de aplicación conjunta

Se trabaja directamente sobre los documentos a generar, las temáticas que se tratan durante las reuniones siguen un esquema y se busca que la misma sea ordenada y racional. Se define una agenda con los puntos principales a tratar durante la jornada. Este tipo de taller tiene el inconveniente de que es muy difícil poder reunir a todas los participantes, es costoso y generalmente es necesaria más de una reunión para establecer los requisitos del sistema. (28)

2.1.3.2.3 Forma de contrato

En lugar de una entrevista, se pueden llenar formularios o contratos indicando los requisitos. En sistemas muy complejos éstos pueden tener centenares de páginas. (28)

2.1.3.2.4 Objetivos medibles

Los requisitos formulados por los usuarios se toman como objetivos generales, a largo plazo, y en cambio se los debe analizar una y otra vez desde el punto de vista del sistema hasta determinar los objetivos críticos del funcionamiento interno que luego darán forma a los comportamientos apreciables por el usuario. Luego, se establecen formas de medir el progreso en la construcción, para evaluar en cualquier momento qué tan avanzado se encuentra el proyecto. (28)

2.1.4 Selección de la técnica

Luego de analizadas las posibles técnicas a utilizar se decidió emplear la entrevista y las lluvias de ideas, de conjunto con los objetivos medibles debido a la dinámica de desarrollo del software y la constante interacción del equipo con Ediciones Futuro.

2.1.5 Requisitos funcionales

Los requisitos funcionales describen la interacción entre el sistema y su ambiente independientemente de su implementación. El ambiente incluye al usuario y cualquier otro sistema externo que interactúa con el sistema. Los requisitos definidos por el cliente se listan a continuación:

Los requisitos funcionales del sistema son los siguientes:

1. Gestionar usuario
2. Gestionar libro
3. Gestionar editorial
4. Gestionar publicación
5. Gestionar género
6. Gestionar comentario

2.1.6 Requisitos no funcionales

Los requisitos no funcionales describen aspectos del sistema que son visibles por el usuario que no incluyen una relación directa con el comportamiento funcional del sistema. Los requerimientos no funcionales incluyen restricciones como el tiempo de respuesta (desempeño), la precisión, recursos consumidos, seguridad, etc.

Usabilidad:

- El sistema será una aplicación web con una curva de aprendizaje baja que pueda ser usada por cualquier persona que posea un nivel básico de conocimientos de computación.
- Deberá utilizar nombres sugerentes para lograr que el usuario encuentre lo que busca en el menor tiempo posible, las acciones a realizar serán fáciles de acceder.
- El sistema debe proporcionar mensajes de error que sean informativos y orientados a usuario final.
- La aplicación web debe poseer un diseño adaptable con el fin de garantizar la adecuada visualización en múltiples computadores personales, dispositivos tableta y teléfonos inteligentes.
- El sistema debe poseer interfaces gráficas bien formadas.

Seguridad:

- El sistema debe usar roles para especificar los privilegios de cada usuario.
- Los permisos de acceso al sistema podrán ser cambiados solamente por el administrador de acceso a datos.
- Todos los sistemas deben respaldarse cada 24 horas. Los respaldos deben ser almacenados en una localidad segura.

Hardware:

- Luego de instalar la solución en hardware con microprocesadores distintos (1era – Intel Pentium, 2da – AMD A8, 3era – Core I3) el sistema respondió más favorable a las peticiones en un servidor de aplicaciones web y de base de datos que cuente como mínimo con microprocesador Core i3 a 2,4 GHz o superior.
- Teniendo en cuenta el peso de la distribución de PHPStorm, el gestor de base de datos de MySQL y de la aplicación se debe disponer de aproximadamente 4GB de espacio de almacenamiento en disco como mínimo en el hardware de despliegue.
- El uso en las máquinas clientes los requerimientos serán menores. Es necesario que estas posean al menos 1GB de memoria RAM y podrán visualizarse desde cualquier sistema de cómputo (móvil o de escritorio) con el software necesario.

- Los hardware clientes y servidor deben poseer una tarjeta de interconexión de red que permita acceder y responder a las peticiones respectivamente.

2.2 Fase 2: Planificación de las entregas

Se cumple con la **Fase II** de la Metodología XP donde el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres semanas.

2.2.1 Historias de usuario

Las historias de usuario son la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

Se realiza la selección del RF_2 por el nivel de prioridad para ser desarrollados a continuación:

Tabla 3 Historia de usuario

Número: 2	**Nombre del requisito:** Autenticar usuario
Programador: Ramón Emanuel Escalona Mayo	**Iteración Asignada:** 2
Prioridad: Alta	**Tiempo Estimado:**
Riesgo en Desarrollo: N/A	**Tiempo Real:**
Descripción: Una vez creado el usuario (ver HU # 2), le permite iniciar sesión y acceder a las funcionalidades que le fueron asignadas según el rol que desempeñe el mismo.	
Observaciones: N/A	
Prototipo elemental de interfaz gráfica de usuario:	

2.2.2 Estimación de tiempo por historia de usuario

La programación bajo la metodología XP (programación extrema) basa sus procesos de planificación en estimaciones temporales de las historias de usuario, las cuáles deben ser realizadas por los desarrolladores durante las diversas reuniones de planificación.

Una historia de usuario es lo suficientemente pequeña como para que el equipo la desarrolle durante una entrega de una a tres semanas; más de tres semanas implica que se debe señalar al cliente que debe dividir una historia de usuario y menos de una semana implica que la historia es demasiado sencilla y por lo que se deben unir dos o más de ellas para su mejor interpretación.

Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto de estimación. Un punto de estimación, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos de estimación. Por otra parte, el equipo de desarrollo mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración.

En la siguiente tabla de estimación del tiempo de las historias de usuario, se usó la denotación de punto de estimación por semana y la denotación de puntos flotantes para indicar días.

Ejemplo:

Tabla 4 Estimación de tiempo

Denotación de punto de estimación	**Interpretación**
1 punto	1 día(8 horas laborales)
1.5 puntos	1 día y 4 horas

Tabla 5 Estimación del tiempo de HU

Historia de usuario	**Punto estimado**
Crear Usuario	8
Eliminar Usuario	3
Modificar Usuario	2
Buscar Usuario	2
Listar/Mostrar Usuario	2
Agregar libro	5
Eliminar libro	2
Modificar libro	3
Buscar libro	2
Listar/Mostrar libro	2
Autenticar usuario	12
Agregar publicación	6
Eliminar publicación	5
Modificar publicación	2
Mostrar publicación	3
Buscar publicación	2
Agregar catálogo	9

Eliminar catálogo	4
Modificar catálogo	7
Mostrar catálogo	3
Agregar editorial	7
Eliminar editorial	5
Modificar editorial	6
Buscar editorial	3
Mostrar/Listar editorial	2
Agregar géneros	6
Eliminar géneros	2
Mostrar géneros	2
Listar géneros	2
Añadir comentario	8

Las estimaciones realizadas permitieron confeccionar una evaluación puntual del tiempo de implementación de cada historia de usuario para la posterior elaboración del plan de iteración. Una vez realizadas las estimaciones fue preciso construir un plan de iteración donde se pueden agrupar estas historias y dar su cumplimiento de manera paulatina.

2.2.3 Plan de iteraciones

Un plan de iteración está constituido por un conjunto secuencial de actividades y tareas, cada una tiene recursos asignados y pueden depender a su vez de otras tareas. El plan de iteración se realiza una vez por cada iteración.

El contenido de la iteración puede variar, dependiendo de la posición dentro del ciclo de vida y de la naturaleza del proyecto. El plan de iteración incluye porciones relevantes de todas las disciplinas para cada iteración en particular, la prioridad de implementación se evalúa en base al cliente y el equipo de desarrollo, y el impacto del riesgo en base al juicio de experto. Los planes de iteración por lo general muestran un planeamiento detallado de quién va a realizar una tarea/actividad de acuerdo en conformidad a que criterios de evaluación.

Tabla 6 Plan de iteraciones

No. Iteración	Historia de Usuario	Prioridad	Esfuerzo Estimado	
1	Crear Usuario	Alta	8	31
	Eliminar Usuario	Media	3	
	Modificar Usuario	Baja	2	
	Buscar Usuario	Media	2	
	Listar/Mostrar Usuario	Baja	2	
	Agregar libro	Alta	5	
	Eliminar libro	Alta	2	
	Modificar libro	Media	3	
	Buscar libro	Alta	2	
	Listar/Mostrar libro	Alta	2	
	Autenticar usuario	Alta	12	45

2	Agregar publicación	Media	8	
	Eliminar publicación	Baja	5	
	Modificar publicación	Media	2	
	Mostrar publicación	Media	3	
	Buscar publicación	Media	2	
	Agregar catálogo	Alta	9	
	Eliminar catálogo	Alta	4	
	Modificar catálogo	Media	7	
	Mostrar catálogo	Media	3	
3	Agregar editorial	Alta	7	43
	Eliminar editorial	Media	5	
	Modificar editorial	Media	6	
	Buscar editorial	Media	3	
	Mostrar/Listar editorial	Media	2	
	Agregar géneros	Alta	6	

	Eliminar géneros	Media	2	
	Mostrar géneros	Media	2	
	Listar géneros	Media	2	
	Añadir comentario	Baja	8	

Una vez realizado el plan de iteración se pudo agrupar las diferentes historias de usuario en 3 iteraciones teniendo en cuenta las características que rigen la metodología XP. Con este plan de iteraciones ya es posible realizar un plan de entrega que será entregado al cliente y que el grupo de desarrollo está obligado a hacer cumplir.

2.2.4 Plan de entrega

Determinada la duración de cada iteración se presenta el plan de entrega elaborado para la fase de implementación teniendo en cuenta que el desarrollo del proyecto inicia el 1/02/2019 y concluirá el 10/06/2019:

Tabla 7 Plan de entrega

	Iteración No.1	**Iteración No.2**	**Iteración No.3**
Cantidad de HU	10	10	10
Fecha de inicio	01/02/2019	02/03/2019	27/04/2019
Fecha de entrega	01/03/2019	26/04/2019	10/06/2019

Una vez realizado el plan de entrega se puede confirmar con el cliente que el proyecto durará 16 semanas aproximadamente, lo que significa que tendrá un tiempo de desarrollo de 4 aproximadamente.

2.2.5 Reuniones diarias de seguimiento

El planeamiento es esencial para cualquier tipo de metodología, es por ello que XP requiere de una revisión continua del plan de trabajo. A pesar de ser una metodología que evita la documentación exagerada, es muy estricta en la organización del trabajo. Para la misma, el equipo de desarrollo definió desde el inicio aquellos espacios en los que el equipo y el cliente realizarían encuentros semanales para la evaluación del cumplimiento de los resultados de las diferentes iteraciones, así como el cumplimiento parcial, total o nuevas redefiniciones a los conceptos establecidos. Las pequeñas entregas facilitarán el trabajo de los desarrolladores, haciendo posible suplir las necesidades del cliente.

Tabla 8 Reuniones de seguimiento

Reunión de / Fecha		**Descripción**
Plan de entregas	24/10/2018	Se realiza entre el equipo de trabajo y los clientes y se define el marco temporal de la realización del sistema. El cliente expone las historias de usuario a los integrantes de grupo, quienes estimarán el grado de dificultad de la implementación de cada historia. A partir de las historias de usuario, el cliente plantea las pruebas de aceptación con las cuales se comprueba que cada una de éstas ha sido correctamente implementada.
Inicial de Iteración	01/02/2019	Esta reunión es realizada previo a iniciar una iteración donde se organizan las actividades de programación a realizar. Las historias de usuario son traducidas a tareas y asignadas a los desarrolladores.
Diarias	Al inicio de cada jornada de trabajo	Estas reuniones se realizan al comenzar la jornada laboral, donde todo el equipo de desarrollo se reúne para exponer los problemas e ideas que se estén presentando. Es de vital importancia evitar las discusiones largas, ya que se está utilizando tiempo laboral que puede ser destinado a la construcción del sistema

Fin de iteración	01/03/2019 26/04/2019 10/06/2019	Estas reuniones se realizan al finalizar cada iteración en conjunto con los clientes para presentar los avances en cada iteración y demostrar la aceptación por los mismos como muestra de una correcta implementación.

Con el plan de reuniones queda determinado la secuencia de las distintas actividades a realizar dentro del proceso de desarrollo de la solución, contribuyendo a la calidad del producto deseado. Una vez determinado los artefactos organizativos, se procede a la fase de diseño de la solución propuesta.

2.3 Fase 3: Etapa de diseño

Teniendo en cuenta lo que plantea la metodología XP, en esta fase se confeccionan las tarjetas *Clase-Responsabilidad-Colaborador* (CRC) para la descripción de las principales clases de los módulos desarrollados. Se define la arquitectura del sistema y los estándares de codificación, así como los patrones de diseño utilizados en el desarrollo de la propuesta.

2.3.1 Tarjetas CRC

Las tarjetas CRC se elaboran durante la fase de diseño de la metodología XP para describir las entidades existentes en la aplicación. El uso de este tipo de tarjetas es una técnica de modelado que permite identificar las clases, responsabilidades y colaboraciones. El objetivo es obtener un diseño simple, elegante y fácil de comprender por parte de los programadores.

Tabla 9 Tarjeta CRC

Nombre clase: AutorController.php
Superclase: Controller.php

Responsabilidad:	Colaborador:
Crear autor	Autor.php
Modificar autor	
Eliminar autor	
Mostrar autor	

Con la realización de las tarjetas **CRC** se evidencia la interrelación existente entre cada clase de la solución y sus diferentes funcionalidades. Teniendo en cuenta la interrelación de las entidades se hace necesario realizar prototipos que evidencien al cliente y al equipo de desarrollo el resultado que se obtendría con la solución.

2.3.2 Arquitectura Modelo-Vista-Controlador

Symfony está basado en un patrón clásico del diseño web conocido como arquitectura MVC, que está formado por tres niveles

- La capa del modelo define la lógica de negocio (la base de datos pertenece a esta capa). Symfony guarda todas las clases y archivos relacionados con el modelo en el directorio lib/model/.
- La vista es lo que utilizan los usuarios para interactuar con la aplicación (los gestores de plantillas pertenecen a esta capa). En Symfony la capa de la vista está formada principalmente por plantillas en PHP. Estas plantillas se guardan en varios directorios llamados templates/ repartidos por todo el proyecto.
- El controlador es un bloque de código que realiza llamadas al modelo para obtener los datos y se los pasa a la vista para que los muestre al usuario.

La arquitectura MVC separa la lógica de negocio (el modelo) y la presentación (la vista) por lo que se consigue un mantenimiento más sencillo de las aplicaciones. Si por ejemplo una misma aplicación debe ejecutarse tanto en un navegador estándar como un navegador de un dispositivo móvil, solamente es necesario crear una vista nueva para cada dispositivo; manteniendo el controlador y el modelo original. El controlador se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comandos, email, etc.). El modelo se encarga de la abstracción de la

lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación.

Ilustración 3 Arquitectura Modelo-Vista-Controlador

Modelo

Ilustración 4 Capa Modelo

Vista

Ilustración 5 Capa Vista

Controlador

Ilustración 6 Capa Controlador

2.3.3 Prototipo de interfaz de usuario

Los prototipos ayudan a identificar, comunicar y probar un producto antes de crearlo. Al tratarse de una plataforma para la distribución electrónica de libros, se ha de conseguir que la atención del usuario que entra en la web se centre mayoritariamente en la información que se presenta dependiendo de intereses personales.

Ilustración 7 Prototipo interfaz página principal

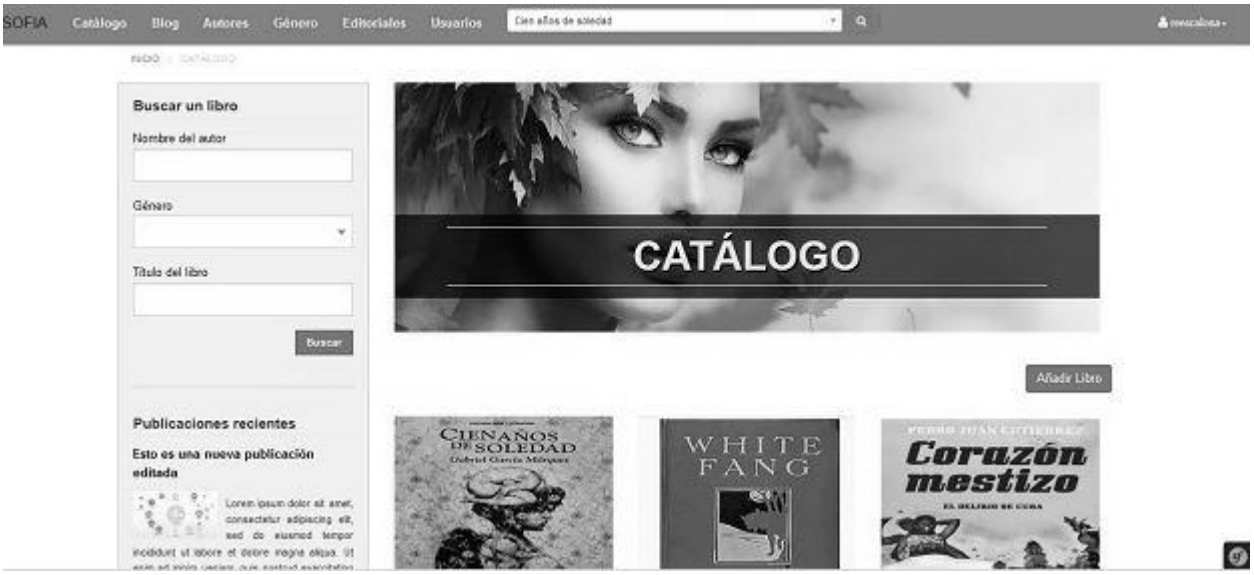

Ilustración 8 Prototipo interfaz catálogo

Ilustración 9 Prototipo interfaz blog

2.4 Fase 4: Implementación

El desarrollo o la codificación es un proceso que se realiza en forma paralela con el diseño y la cual está sujeta a varias observaciones por parte de XP. Dentro de esta fase se describen los estándares de codificación, así como los patrones de diseño que se emplearon dentro del sistema.

2.4.1 Estándares de codificación

En el proceso de desarrollo de un software siguiendo la metodología XP, es necesario que exista una adecuada comunicación entre los programadores del equipo de desarrollo. Para lograr esto, se establecen un conjunto de estándares definidos por las convenciones de escritura de código Php abarcando el marco de trabajo Symfony.

El código fuente de Symfony sigue las recomendaciones de los estándares PSR-1 y PSR-2 definidos por la comunidad PHP.

PSR-1 Norma Básica de Codificación:

Esta sección de la norma comprende lo que se debe considerar como los elementos de codificación estándar que se requieren para garantizar un alto nivel de interoperabilidad técnica entre el código PHP compartido. (20)

- Los archivos deben utilizar solo las etiquetas <? Php y <? =.

```
<?php
namespace DistribuidoraBundle\Controller;
use DistribuidoraBundle\Entity\Autor;
```

- Los nombres de los métodos deben ser declarados en camelCase.

```
public function newAction(Request $request)
{
$autor = new Autor();
$em = $this->getDoctrine();
$editoriales = $em->getRepository(Editorial::class)->findAll();
```

PRS-2 Guía para el estilo de código

Esta guía se extiende y expande en PSR-1, el estándar de codificación básico.
El objetivo de esta guía es reducir la fricción cognitiva al escanear códigos de diferentes autores. Lo hace enumerando un conjunto compartido de reglas y expectativas sobre cómo formatear el código PHP. (20)

- Las llaves de apertura para las clases deben ir a la siguiente línea, y las llaves de cierre deben ir a la siguiente línea después del cuerpo.

```
class Autor implements \JsonSerializable
{
@var int
```

- Las llaves de apertura para los métodos deben ir a la siguiente línea, y las llaves de cierre deben ir a la siguiente línea después del cuerpo.

```
public function newAction(Request $request)
{
$autor = new Autor();
$em = $this->getDoctrine();
```

- Las llaves de apertura para las estructuras de control deben ir en la misma línea, y las llaves de cierre deben ir en la siguiente línea después del cuerpo.

```
for ($i = 0 ; $i < count($libro_all);$i++){
for ($j = 0;$j<count($libro_all[$i]->getAutor());$j++){
if (in_array($libro_all[$i]->getAutor()[$j]->getNombreCompleto() ,$autores)){
```

2.4.2 Patrones de diseño

Si bien la elaboración de un buen diseño del software contribuye directamente a la calidad del producto final, gran parte de esta yace en la utilización adecuada de los patrones de diseño y arquitectónicos existentes en la actualidad. Los patrones son soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos. Son soluciones basadas en la experiencia y que se ha demostrado que funcionan. (29)

Con empleo del marco de trabajo Symfony se garantiza la utilización de varios de estos patrones, permitiendo a los desarrolladores utilizar buenas prácticas de programación y ahorrar tiempo y recursos. A continuación, se describen los principales patrones empleados en el diseño e implementación de la solución.

2.4.2.1 Patrón Modelo-Vista-Controlador

El patrón MVC obliga a dividir y organizar el código de acuerdo a las convenciones establecidas por el marco de trabajo, la interfaz de usuario se guarda en la vista, la manipulación de datos se guarda en el modelo y el procesamiento de las peticiones constituye el controlador. El empleo del patrón MVC en un sistema resulta bastante útil además de restrictivo. Symfony está basado en el patrón de arquitectura conocido MVC, formado por tres niveles: el Modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio, la Vista es la encargada de originar las páginas que son mostradas como resultado de las acciones y el Controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista. Symfony toma lo mejor de la

arquitectura MVC y la implementa de forma que el desarrollo de aplicaciones sea rápido y sencillo. La implementación que realiza Symfony de la arquitectura MVC incluye varias clases como son:

- Controller: es la clase del controlador y se encarga de decodificar la petición y transferirla a la acción correspondiente.
- Request: guarda todos los elementos que integran la petición (parámetros, cookies, cabeceras, entre otros).
- Response: posee las cabeceras de la respuesta y los contenidos. El contenido de este objeto se convierte en la respuesta HTML que se remite al usuario. (cabecera, etiquetas html).

2.4.2.2 Patrones GRASP

- Controlador: Symfony implementa un controlador frontal para atender todas las peticiones web, es el único punto de entrada de toda la aplicación en un determinado entorno. Cuando recibe una petición, utiliza el sistema de enrutamiento para enviar la acción al controlador responsable de la solución y se encarga de manejar la seguridad y ejecución de los filtros. Este patrón se evidencia en las clases FrontController, WebFrontController, Contex, los “actions” y el index.php del ambiente.
- Alta Cohesión: Symfony permite la organización del trabajo en cuanto a la estructura del proyecto y la asignación de responsabilidades con una alta cohesión. Un ejemplo de ello son las clases actions, las cuales son responsables de definir las acciones para las plantillas y colaborar con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades.
- Bajo Acoplamiento: las clases relacionadas con el controlador frontal y que se encuentran en el paquete actions heredan únicamente de Actions para alcanzar un bajo acoplamiento entre clases. Las clases que implementan la lógica del negocio y de acceso a datos se encuentran en el modelo, las cuáles no se asocian directamente con las clases de la vista o el controlador, lo que proporciona que la dependencia en este caso sea baja.
- Experto: Symfony utiliza Propel o Doctrine para realizar su capa de abstracción en el modelo, encapsular toda la lógica de los datos y generar las clases con todas las funcionalidades comunes de las entidades, las clases de abstracción de datos poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan y contienen la información necesaria de la tabla que representan.

2.4.3 Modelo de datos

EL modelo de datos es un lenguaje orientado a hablar de una base de datos. Típicamente un modelo de datos permite describir:

- Las estructuras de datos de la base: El tipo de los datos que hay en la base y la forma en que se relacionan.
- Las restricciones de integridad: Un conjunto de condiciones que deben cumplir los datos para reflejar la realidad deseada.
- Operaciones de manipulación de los datos: típicamente, operaciones de agregado, borrado, modificación y recuperación de los datos de la base.

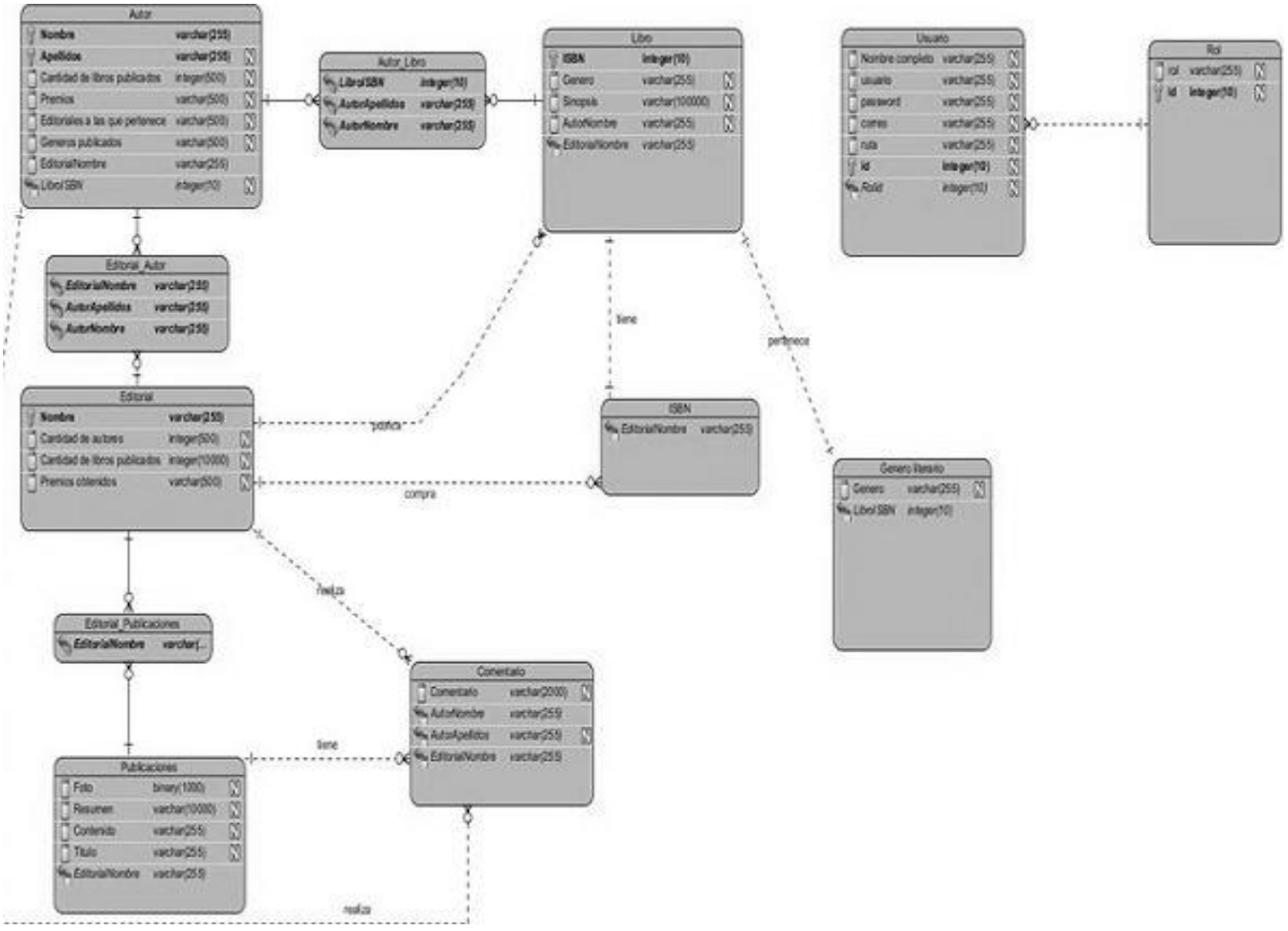

Ilustración 10 Modelo de datos

2.5 Conclusiones del capítulo

- La generación de los artefactos en la etapa de implementación de la plataforma siguiendo la Metodología XP, facilitaron las tareas de implementación, la reducción de errores y riesgos.
- El uso del modelo conceptual permitió describir el contexto para la distribución electrónica de libros digitales en Cuba e identificar la complejidad de las características para la propuesta de solución
- Se realizó la captura de los principales requerimientos que contribuyó a la confiabilidad de los datos recolectados y a satisfacer las expectativas de los usuarios mediante las características de la aplicación.
- Se realizaron las tarjetas CRC para describir las entidades existentes y tener un mayor dominio de las diferentes clases a desarrollar dentro de la solución.

.

CAPÍTULO 3: EVALUACIÓN DE LA SOLUCIÓN PROPUESTA

En este capítulo se aborda las diferentes pruebas realizadas al software y que son exigidas por la metodología XP para el buen funcionamiento y calidad de la propuesta desarrollada. Se detalla con exactitud los diferentes resultados alojados por cada prueba y la respuesta a dichas no conformidades.

3.1 Diagrama de despliegue

En el diagrama de despliegue demuestra cómo y dónde se despliega el sistema. Las máquinas físicas y los procesadores se representan como nodos, y la construcción interna puede ser representada por nodos o artefactos embebidos. Los estereotipos permiten la naturaleza del equipo: dispositivos, procesadores y memoria.

Los elementos usados por este tipo de diagrama son nodos (representados como un prisma), componentes (representados como una caja rectangular con dos protuberancias del lado izquierdo) y asociaciones.

A continuación, se representa el ambiente donde debe ser desplegada la solución:

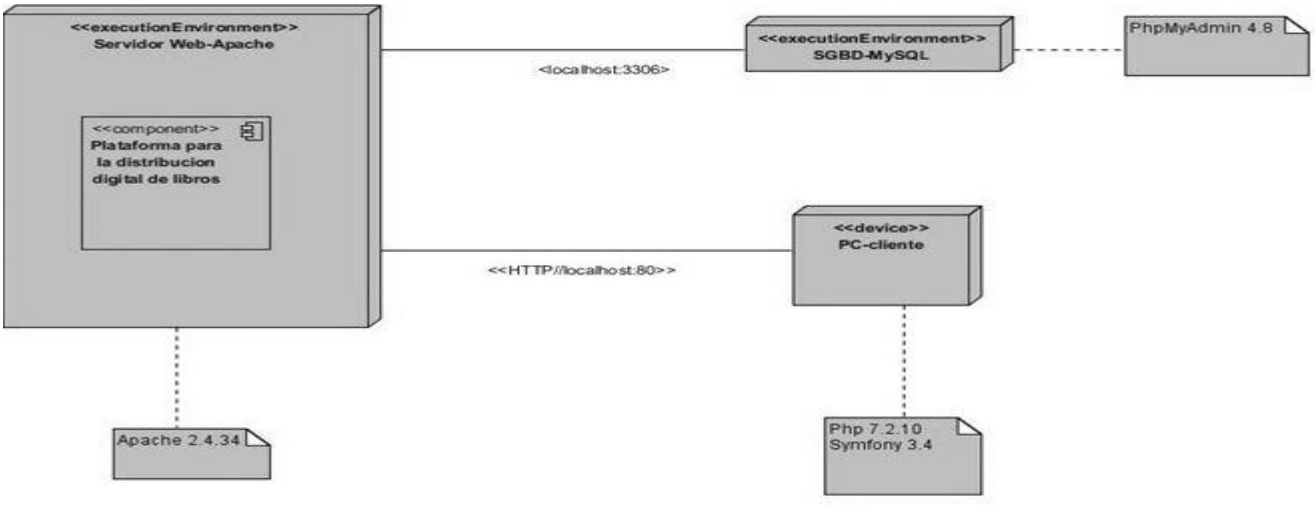

Ilustración 11 Diagrama de despliegue

3.2 Fase 5: Pruebas

Las pruebas de software forman parte de la última fase que propone XP, con el objetivo de lograr una herramienta que cumpla con los requisitos previamente identificados. En el presente capítulo se describen las etapas de codificación y de pruebas de la propuesta de solución

Las pruebas de software son una investigación orientada a proporcionar a los interesados información sobre la calidad del producto o servicio bajo prueba. Las pruebas de software también pueden proporcionar una visión objetiva e independiente del software para permitir a los desarrolladores apreciar y entender los riesgos del software puesto en práctica. (30)

Por lo general, las pruebas involucran operaciones del sistema evaluando los resultados bajo condiciones controladas, lo que hace que la realización de pruebas al software sea un factor de vital importancia. En pos de comprobar el estado de la calidad del software, se siguen las siguientes etapas:

- Verificación de la interacción de componentes.
- Verificación de la integración adecuada de los componentes.
- Verificación de que todos los requisitos se han implementado correctamente.
- Identificación de los defectos encontrados y aseguramiento de su corrección antes de entregar el software.
- Diseño de pruebas que sistemáticamente identifiquen diferentes clases de errores, con la menor cantidad de tiempo y esfuerzo.

3.2.1 Estrategia de pruebas

La Metodología XP propone que las pruebas de software sean realizadas al término de cada iteración, garantizando el funcionamiento deseado y la aceptabilidad por el cliente para realizar una entrega funcional y acorde a las exigencias de un producto con calidad. Las dos pruebas exigidas por la metodología, por su importancia y agilidad en el proceso; son las pruebas unitarias y de aceptación. Se hicieron las pruebas unitarias al código al finalizar cada iteración e igualmente se realizaron las pruebas de aceptación.

3.2.2 Pruebas unitarias

Las pruebas unitarias son **pruebas de caja blanca** donde los componentes individuales del software se someten a pruebas. El propósito de estas es asegurar que cada unidad de trabajo funciona individualmente bien, responde como se espera que deba responder, o falle como y cuando se supone que debe fallar. Se supone que las pruebas unitarias deben probar la unidad mínima de trabajo de un

programa que es aquella que devuelve un valor o produce un cambio en el estado del programa, generalmente hablaríamos de un solo método o una función.

Contrariamente a las pruebas de caja negra, en las de caja blanca el software se ve como un cuadro blanco, o de vidrio, que permite ver la estructura y el flujo del software bajo prueba. Los planes de pruebas se hacen de acuerdo a los detalles de la implementación del software, tales como lenguaje de programación, la lógica y estilos. Los casos de prueba se derivan de la estructura del programa.

Estas pruebas se basan en el minucioso examen de los detalles procedimentales, donde se comprueban los caminos lógicos del software proponiendo casos de prueba que examinen que todas las condiciones y/o bucles estén correctas para determinar si el estado real coincide con el esperado o afirmado. (31)

Los objetivos fundamentales de estas pruebas son:

- Garantizar que se ejerciten por lo menos una vez todos los caminos independientes de cada módulo, programa o método.
- Ejercitar todas las decisiones lógicas en las vertientes verdadera y falsa.
- Ejecutar todos los bucles en sus límites operacionales.
- Ejercitar las estructuras internas de datos para asegurar su validez.

Debido a la disminución en un alto por ciento del número de errores existentes en los sistemas gracias a la aplicación de las pruebas de caja blanca, estas son consideradas como uno de los tipos de pruebas más importantes, por mejorar la calidad y confiabilidad del software analizado.

Luego de realizar la prueba de caja blanca a la función guardar_autorAction(Request $request) se determinó que la complejidad ciclomática era 3 por lo que el método es sencillo y de poco riesgo para el sistema. (31)

v(G) = e – n + 2, donde *e* representa el número de aristas y *n* el número de nodos.

e= 10 n=9

v(G) = número de regiones cerradas en el grafo + 1

numero de regiones cerradas del grafo=2

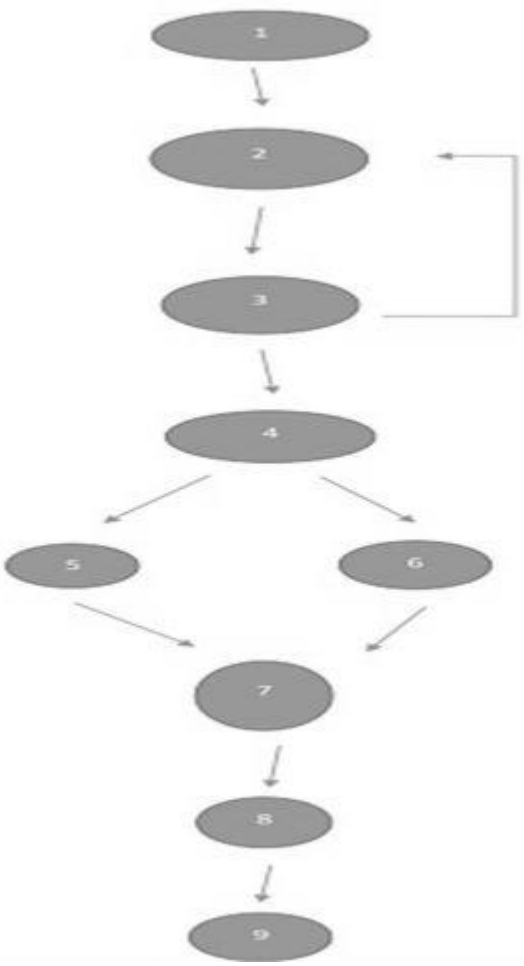

Ilustración 12 Camino básico

Caminos:

1,2,3,2,3,4,5 7,8,9

1,2,3,2,3,4,6,7,8,9

1,2,3,4,5,7.8.9

3.2.3 Pruebas de aceptación

Las pruebas de aceptación son **pruebas de caja negra** definidas por el cliente para cada historia de usuario, y tienen como objetivo asegurar que las funcionalidades del sistema cumplen con lo que se espera de ellas. En efecto, las pruebas de aceptación corresponden a una especie de documento de requerimientos en XP, ya que marcan el camino a seguir en cada iteración, indicándole al equipo de

desarrollo hacia donde tiene que ir y en qué puntos o funcionalidades debe poner el mayor esfuerzo y atención (32).

Las pruebas de caja negra negro son un método de ensayo en el que los datos de prueba se derivan de los requisitos funcionales especificados sin tener en cuenta la estructura final del programa. Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos. (31)

Mientras más amplio sea el espectro de elementos de entrada para realizar la prueba, las probabilidades de encontrar problemas en el software aumentan y, por lo tanto, será más confiable la calidad del software.

Este tipo de pruebas permite encontrar:

- Funciones incorrectas o ausentes
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

Para preparar los casos de prueba es necesario un número de datos que ayuden a la ejecución de estos casos y que permitan que el sistema se ejecute en todas sus variantes. Estos datos pueden ser válidos o no para el software y son seleccionados atendiendo a las especificidades de la funcionalidad a probar. Para realiza estas pruebas existen varias técnicas:

- **Partición de equivalencia:** divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. Esta técnica es muy efectiva puesto que permite examinar los valores válidos e inválidos de las entradas existentes en el producto, descubre de forma inmediata una clase de errores que, de otro modo, requerirían la realización de muchos casos antes de detectar el error genérico. La partición de equivalente se dirige a la definición de casos de pruebas que descubran clases de errores, reduciendo de este modo el número de pruebas a realizar.

Se realizaron 6 casos de prueba a continuación se muestra un ejemplo de los mismos:

Tabla 10 Clases de equivalencia

Identificador	Entrada(clase)	Clases válidas	Clases inválidas

Nombre	Nombre complete	Nombre=a-z	Nombre=0-9,caracteres especiales
Foto	Foto	Foto=.jpg, .png, “ ”	Foto=otro format
Correo	Correo	Correo=	Correo=””,
Editorial	Editoril	Editorial=a-z	Editorial=0-9,caracteres especiales

Id del escenario	Escenario	Nombre	Foto	Correo	Editorial	Respuesta del sistema	Resultado de la prueba
EC1	Guardar autor	Luis Ruben Lima Mateo V	Foto.jpg V	lrlima@uci.cu V	Ediciones Futuro V	El autor se adiciono correctamente	Satisfactorio
		Ramon Escalona Mayo V	Foto.gif I	reescalona@uci.cu V	Ediciones Abril V	Debe elegir otra foto	No satisfactorio
		Yandri H3rn4nd3z i	Foto.png V	yinerarity@uci.cu V	Ediciones Futuro V	El nombre no cumple los requisitos	No satisfactorio
		Ramon Escalona Mayo V	Foto.jpg V	reescalona@uci.cu V	Ediciones 66765 I	El nombre de la editorial es incorrecta	No satisfactorio
		Luis Ruben Lima V	Foto.png V	lrlima.uci.cu I	Ediciones Abril V	El correo es incorrecto	No satisfactorio

		Ramon Escal456a Mayo I	Foto.gif I	reescalona@uci.cu V	Ediciones Abril V	Los datos son incorrectos	No satisfactorio
		Ramon Escalona Mayo V	Foto.gif I	reescalonauci.cu I	Ediciones Abril V	Los datos son incorrectos	No satisfactorio
		Ramon Escalona Mayo V	Foto.jpg V	reescalona@i.cu I	Ediciones I	Los datos son incorrectos	No satisfactorio

3.2.4 Pruebas de rendimiento

Las pruebas de performance o rendimiento permiten conocer y mitigar los riesgos relacionados con el mal desempeño de las aplicaciones en los entornos de producción y realizar las correcciones necesarias antes de salir al mercado.

Entre las acciones realizadas están:

- Cuantificación de la capacidad de la infraestructura.
- Validación de los requerimientos de rendimiento, la escalabilidad de las plataformas y del sistema a probar.

De esta manera, los desarrolladores pudieron conocer qué cantidad de clientes simultáneos soporta su producto, con tiempos y datos razonables sobre la infraestructura y las plataformas propuestas. Asimismo, puede saber si es suficiente el hardware para soportar el nivel propuesto de transacciones y qué expectativa de crecimiento soporta. Es relevante la selección de la herramienta adecuada, la definición de los escenarios de prueba y la configuración del entorno.

Una vez ejecutadas las pruebas, es importante la evaluación de resultados para identificar los posibles problemas y las posibilidades de mejora sobre el desempeño del sistema. La prueba de rendimiento se da durante todos los pasos del proceso de la prueba. Incluso al nivel de unidad, se debe asegurar el rendimiento de los módulos individuales a medida que se llevan a cabo las pruebas de caja blanca. Sin embargo, hasta que no estén completamente integrados todos los elementos del sistema no se puede asegurar realmente el rendimiento del mismo. (31)

3.2.5 Pruebas de resistencia

Durante los pasos de prueba anteriores, las técnicas de caja blanca y de caja negra daban como resultado la evaluación del funcionamiento y del rendimiento normal del programa. Las pruebas de resistencia están diseñadas para enfrentar a los programas con situaciones anormales. (31)

Una variante de la prueba de resistencia es una técnica denominada prueba de sensibilidad. En algunas situaciones (la más común se da con algoritmos matemáticos), un rango de datos muy pequeño dentro de los límites de una entrada válida para un programa puede producir un proceso exagerado e incluso erróneo o una profunda degradación del rendimiento. Esta situación es análoga a una singularidad en una función matemática. La prueba de sensibilidad intenta descubrir combinaciones de datos dentro de una clase de entrada válida que pueda producir inestabilidad o un proceso incorrecto.

Para el desarrollo de las pruebas de carga y estrés se utilizó como herramienta el ApacheBench, el cual tiene como características principales las siguientes:

- Al ser un software de código abierto, está disponible de manera gratuita.
- Es un programa que podemos utilizar desde la línea de comandos de manera simple.
- Es una herramienta independiente de la plataforma que utilicemos. Esto significa que vamos a poder utilizarla igualmente en Gnu/Linux o en servidores Windows.
- El programa puede realizar pruebas de carga y rendimiento únicamente para el servidor web: HTTP o HTTPS.

Luego de realizadas las pruebas de Carga y estrés arrojaron el siguiente resultado:

Time taken for tests: 683.976 seconds
Complete requests: 1000
Failed requests: 0
Total transferred: 82762000 bytes
HTML transferred: 82405000 bytes
Requests per second: 1.46 [#/sec] (mean)
Time per request: 6839.760 [ms] (mean)
Time per request: 683.976 [ms] (mean, across all concurrent requests)
Transfer rate: 118.17 [Kbytes/sec] received

Connection Times (ms)

	min	mean[+/-sd]		median	max
Connect:	0	0	0.4	0	1

Processing:	643	6746	1162.7	6439	13755
Waiting:	542	66661151.3		6359	13640
Total:	644	6746	1162.7	6439	13755

Porcentaje de peticiones en un tiempo determinado:

50% 6439
66% 6855
75% 7199
80% 7506
90% 8227
95% 8844
98% 9567
99% 11121
100% 13755 (mayor peticion)

Después de realizadas las pruebas de carga y estrés en un ambiente controlado se llegó a la conclusión que el sistema es capaz de funcionar adecuadamente hasta con 13755 peticiones, que el tiempo de respuesta es menor que 1 segundo y sin fallo alguno.

3.2.6 Pruebas de seguridad

Las pruebas de seguridad abarcan más allá de lo que es el simple escaneo de puerto, los probadores deben utilizar enfoques basados en el riesgo, basados tanto en la realidad arquitectónica del sistema como en la mentalidad del atacante, para evaluar adecuadamente la seguridad del software. (33)

Para el desarrollo de las mismas se utilizó la herramienta Acunetix que ha sido pionero en la tecnología de análisis de seguridad de aplicaciones. Incluye varias características innovadoras:

- Un analizador automático de JavaScript que permite realizar pruebas de seguridad de Ajax y aplicaciones Web 2.0
- Dispone de los test más avanzados y profundos de análisis y pruebas de Inyección de SQL y Cross Site Scripting.
- El grabador de macros hace que las pruebas sobre los formularios de las áreas protegidas de la Web sean más fáciles.
- Diversos tipos de informes, incluidos informes de conformidad VISA PCI.
- El analizador de vulnerabilidades, rapidísimo y multitarea, rastrea cientos de miles de páginas con facilidad.

- Pruebas de vulnerabilidad de carga automatizada de informes.
- Acunetix rastrea y analiza los sitios Web incluyendo el contenido de Flash, AJAX y SOAP
- Innovadora Tecnología AcuSensor que permite la exploración precisa de muchas vulnerabilidades.
- Análisis de puertos red y alertas contra el servidor Web para complejos controles de seguridad.

Se realizó un análisis del sistema con la herramienta, los resultados del mismo están reflejados en el informe emitido por la misma herramienta.

3.3 Resultados de las pruebas

A continuación, se muestra un gráfico final con los resultados arrojados en las cuatro iteraciones de pruebas donde se obtuvo un total de 15 no conformidades (NC) significativas: 8 con prioridad normal, 3 con prioridad baja y 4 con prioridad alta. Los resultados obtenidos de las pruebas de aceptación por cada iteración pueden verse en la siguiente figura de este documento.

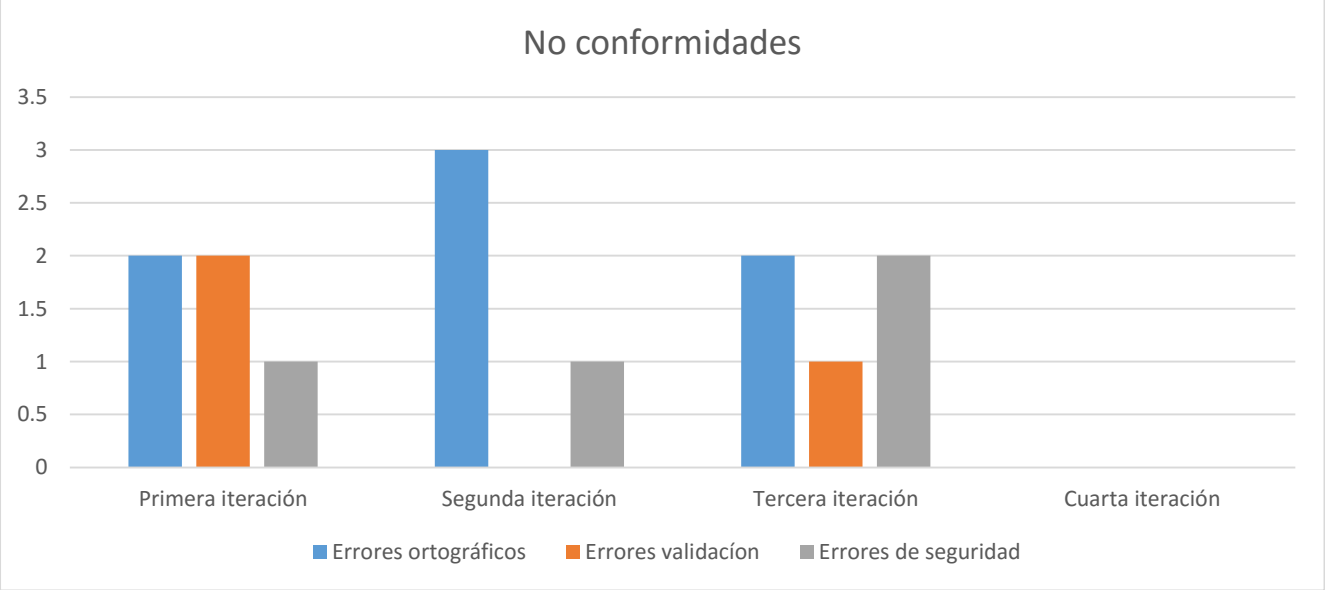

Ilustración 13 Gráfico de no conformidades

Figura 1: Resumen del resultado de las pruebas de aceptación.

Las principales NC no significativas encontradas fueron errores ortográficos, omisiones de tildes y cambio de mayúsculas por minúsculas. También se mostraban algunos mensajes innecesarios en pantalla y elementos de interfaz no sugerentes para un usuario con poca experiencia. Las principales NC significativas encontradas fueron errores de validación, errores en validaciones de los campos de los formularios. Después de concluida cada iteración se resolvieron las NC arrojadas quedando en un estado cerrado.

3.4 Validación de las variables

Luego del análisis, diseño e implementación de la solución se puede demostrar que:

Tabla 11 Validación de variables

Características	Antes de la solución	Después de la solución
Editoriales con páginas de cara a internet	Solo 22 sellos editoriales	180 sellos editoriales
Cantidad de libros a los que pueden acceder las personas	Solo los que son impresos	Todos los libros registrados en la Agencia Cubana de ISBN (más de 15 000)
Cantidad de autores con páginas en internet	Muy pocos	Más de 3 500 autores
Utilización de la conectividad para promoción y consumo de contenido	No se utilizaba	En Cuba existen más de 5 millones de líneas celulares, más de 2 millones de conexiones diarias, y todo lo expuesto está integrado en un solo sistema, dominio, permitiendo su rápida indexación, visibilidad, tráfico y posicionamiento en motores de búsqueda.

Por tanto, se puede afirmar que con el desarrollo de la plataforma SOFIA contribuirá a la visibilidad y posicionamiento de los contenidos de las obras literarias cubanas.

3.5 Conclusiones del capítulo

- Los diferentes métodos de pruebas aplicados a la solución durante el ciclo de vida del software permitieron comprobar los errores existentes y mejorar la calidad de los resultados.

- Se realizaron prueba de aceptación que pudo evidenciar el correcto funcionamiento y la aceptabilidad de los requisitos descritos de conjunto con el cliente en cada iteración en el proceso de desarrollo de la metodología XP.
- Los resultados de estas pruebas a la plataforma demostraron su efectividad para ser desplegada.

CONCLUSIONES GENERALES

Considerando los resultados descritos en este informe, la necesidad y el objetivo planteado por la investigación se arriban a las siguientes conclusiones:

- El análisis de las diferentes fuentes, herramientas y soluciones tecnológicas para la distribución electrónica de libros permitió identificar las características claves que distinguen la plataforma propuesta y su adaptación al contexto nacional.
- El empleo de las herramientas y tecnologías seleccionadas para la implementación de la solución propició la correspondencia entre los resultados obtenidos y los esperados, lo cual pudo asegurar el nivel de precisión en el análisis y diseño de la plataforma.
- La aplicación de diversas pruebas de software arrojó resultados satisfactorios en relación al código y el conjunto de interfaces implementadas, las cuales demostraron, además, la invulnerabilidad en el código y su capacidad técnica en función del entorno para el cual fue desarrollado.
- Los resultados de la investigación que describe este informe demostraron la viabilidad de la solución propuesta para la distribución electrónica de libros digitales en Cuba, cuyo desarrollo contribuye a la visibilidad y posicionamiento de la producción literaria cubana.

RECOMENDACIONES

Los resultados obtenidos en este trabajo sientan las bases para la realización de proyectos futuros que permitan enriquecer y escalar la solución propuesta. Por lo que se recomienda que:

- Integrar la plataforma SOFIA con una pasarela nacional de pago apenas se encuentre disponible.
- Desarrollar complemento de analítica web para monitorear el tráfico y comportamiento de los usuarios.
- Consumir el servicio del registro ISBN a la plataforma apenas se encuentre disponible.
- Desarrollar complemento que permita a través del Centro Nacional de Derecho de Autor la validación de la autoría de los títulos literarios.
- Integrar la plataforma SOFIA con repositorios universitarios y el sistema de catálogos de las bibliotecas.

REFERENCIAS BIBLIOGRÁFICAS

1. **Cubadebate.** Cubadebate. *http://cubadebate.cu.* [En línea] 2018. [Citado el: 10 de 5 de 2018.]

2. **Constitución de la República de Cuba.** *Constitución de la República de Cuba.* 2019.

3. **Oficina Nacional de Estadística e Información.** *Tecnología de la Información y las comunicaciones indicadores seleccionados.* La Habana : s.n., 2018.

4. **PeriodicoVictoria.** PeriodicoVictoria. *PeriodicoVictoria.* [En línea] 2018. http://www.periodicovictoria.cu/cierra-la-feria-del-libro-2018-en-cuba-con-mas-de-1-millon-290-mil-ejemplares-vendidos/.

5. **Abrew, Karl de.** *eBook are Here to stay.* 2013.

6. **Editorial.** Editorial. *Editorial.* [En línea] 2009. https://definicion.de/editorial/.

7. **Software Libre.** Software Libre. *Software Libre.* [En línea] 2019. https://www.significados.com/software-libre/.

8. **Software Propietario.** Software Propietario. *Software Propietario.* [En línea] 2015. https://okhosting.com/blog/software-propietario/.

9. **Amazon.** Amazon. *http://www.amazon.com/Books/b?ie=UTF8&node=301731.* [En línea] Amazon, 2018. [Citado el: 10 de 8 de 2018.] http://www.amazon.com/Books/b?ie=UTF8&node=301731.

10. **Bubok.** Bubok. *Bubok.* [En línea] 2019. https://www.bubok.es/.

11. **Google Play.** Google Play. *Google Play.* [En línea] 2019. https://play.google.com/store/apps.

12. **Apple.** Apple. [En línea] 2018. http://www.apple.com.

13. **Marina Borrell, Jose Antonio Cordon Garcia.** *Plataformas de comercialización de libros electtronicos y oferta de titulos en castellano.* Salamanca : s.n., 2014.

14. **Grammata.** Grammata. *Grammata.* [En línea] 2019. https://libreriagrammata.com/.

15. **Piattini.** *Análisis y diseño de aplicaciones informáticas de gestión.* 2018.

16. **Montero, Bryan Molina.** *Metodologias agiles frente a las tradicionales enel proceso de desarrollo de software.* s.l. : Espirales, 2018. ISSN: 2550-6862.

17. **Software Development Methodologies.** Software Development Methodologies. *Software Development Methodologies.* [En línea] http://softwaredevelopmentmethodologies.org.

18. **Fases XP.** Fases XP. *Fases XP.* [En línea] 2016. http://oness.sourceforge.net/proyecto/html/ch05s02.html.

19. **UML.** UML. *UML.* [En línea] 2018. https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml.

20. **Symfony.** Symfony. *Symfony.* [En línea] 2018. https://symfony.com/doc/current/index.html.

21. **Visual Paradigm.** Visual Paradigm. *Visual Paradigm.* [En línea] 2018. https://online.visual-paradigm.com/es/solutions/diagramming-tool/.

22. **Cubava.** Cubava. *Cubava.* [En línea] 2016. https://deprogramacion.cubava.cu/2016/02/01/que-es-un-ide/.

23. **Netbeans IDE.Features.** Netbeans IDE.Features. *Netbeans IDE.Features.* [En línea] 2016. http://netbeans.org/features/index.html.

24. **Jetbrains.** Jetbrains. *Jetbrains.* [En línea] 2018. https://www.jetbrains.com/phpstorm/.

25. **Oracle Corporation.** Oracle Corporation. *Oracle Corporation.* [En línea] 2018. http://www.mysql.com/why-mysql/.

26. **Sevilla, Universidad de.** 2012.

27. **Agusto, Raúl Monferrer.** *Especificación de Requisitos Software según el estándar de IEEE 830.* s.l. : Departament d'Informàtica - Universitat Jaume I, 2001.

28. **Mesias, Fernando Torres.** *INGENIERÍA DE REQUERIMIENTOS: MÉTODOS Y TÉCNICAS.* Universidad Autonoma Aguascalientes : s.n., 2014.

29. **Blancarte, Oscar.** *Introducción a los parones de diseño.* 2016.

30. **Pruebas.** Testing. *Testing.* [En línea] 2014. www.testingeducation.org.

31. **Pressman, Roger S.** *Ingeniería de Software - Un enfoque práctico 7ma.* MEXICO : s.n., 2011.

32. **Malfará, Dayvis, y otros.** *Testing en eXtreme Programming.* 2006.

33. **Acunetix.** Acunetix. *Acunetix.* [En línea] 2018. https://www.acunetix.com/.

ANEXO 1. GLOSARIO DE TÉRMINOS

Plataforma: En informática, una plataforma es un sistema que sirve como base para hacer funcionar determinados módulos de hardware o de software con los que es compatible. Dicho sistema está definido por un estándar alrededor del cual se determina una arquitectura de hardware y una plataforma de software (incluyendo entornos de aplicaciones). Al definir plataformas se establecen los tipos de arquitectura, sistema operativo, lenguaje de programación o interfaz de usuario compatibles.

Lenguaje de modelado: Se entiende por lenguaje de modelado cualquier lenguaje artificial que puede ser utilizado para expresar la información, el conocimiento o sistemas en una estructura que está definida por un conjunto coherente de reglas.

Lenguaje de programación: Un Lenguaje de Programación es un conjunto de reglas, notaciones, símbolos y/o caracteres que permiten a un programador poder expresar el procesamiento de datos y sus estructuras en la computadora. Cada lenguaje posee sus propias sintaxis. También se puede decir que un programa es un conjunto de órdenes o instrucciones que resuelven un problema específico basado en un Lenguaje de Programación (34)

API: Una API es una interfaz de programación de aplicaciones (del inglés API: Application Programming Interface). Es un conjunto de rutinas que provee acceso a funciones de un determinado software.

MVC: Arquitectura Modelo-Vista-Controlador.

ANEXO 2. ENTREVISTA

Entrevista con el cliente

1-Como se registran los libros y como le notifican al autor?

2-Como se lleva a cabo las publicaciones de los libros?

3-Cuales considera que sean los datos más importantes de un libro para que sea de interés al lector?

4-Como la editorial adquiere nuevos libros, ¿es decir ella busca a los autores o los autores van a la editorial?

5-Cuales son los datos más significativos de una editorial?

6-Cuales son los datos más significativos de un autor?

7-Se puede clasificar un autor según sus publicaciones o escritos en alguna categoría?

8-Quien es el encargado de registrar los libros, ¿sin que se viole el derecho de autor?

9-Como o cuales son los principales eventos literarios en Cuba?

10-Existe la manera de saber dónde encontrar un libro?

11-Existe la manera de saber las publicaciones de un autor?

12-Que datos de interés podría brindarnos que puedan atraer a los lectores como experiencia?

13-Tienen las editoriales en Cuba páginas en internet o en redes sociales?

14-Que privilegios podría otorgársele a un autor o editorial por su buen trabajo o desempeño?