Universidad de las Ciencias Informáticas

Facultad 1

Título: “Sistema para la aplicación de pautas de accesibilidad Web al Sistema Integrado de Gestión Bibliotecaria de la Universidad de las Ciencias Informáticas”

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

Autor:

Hilario Mastrapa Montero

Tutores:

Ing. Edisnel Carrazana Castro

Ing. Ramón Ernesto de Ávila León

Ciudad de la Habana, Cuba

Junio 2013

Declaración de autoría

Declaro que soy el único autor de esta investigación y autorizo a la Facultad 1 de la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio.

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

Autor:

_____

Hilario Mastrapa Montero

Tutores:

_____ _____

Ing. Edisnel Carrazana Castro Ing. Ramón Ernesto de Ávila León

Datos del contacto

Ing. Edisnel Carrazana Castro.

Es Ingeniero en Ciencias Informáticas desde su graduación en 2007, con categoría docente de Instructor, labora como profesor de la Facultad 2 de la Universidad de las Ciencias Informáticas (UCI), ha impartido durante varios cursos las asignaturas Programación II e Inteligencia Artificial. Durante 5 años se desempeñó como Jefe de la línea de producción Soluciones para la gestión de bibliotecas en el centro CENIA, en la cual dirigió varios proyectos de desarrollo de software para bibliotecas y centros de documentación. Actualmente es arquitecto del proyecto Sistema integrado de gestión bibliotecaria del centro ISEC. Sus trabajos más relevantes en el área investigativa se encuentran en la automatización de los procesos bibliotecarios y la gestión de la información y el conocimiento.

Email: ecarrazana@uci.cu

Ing. Ramón Ernesto de Ávila León.

Ocupa la plaza de especialista general, con 3 años de experiencia, graduado de Ingeniería en Ciencias Informáticas, actualmente se desempeña como líder de proyecto del Sistema de Gestión de Asignación de Gas Licuado para la Universidad de las Ciencias Informáticas.

Email: redeavila@uci.cu

Agradecimientos

La realización del presente trabajo implicó mucho esfuerzo, empeño, interés, dedicación, gracias al apoyo incondicional de muchas personas fue posible la realización del mismo. Agradecer a la UCI y a nuestros profesores, por haber hecho de nosotros profesionales y hacer que nuestros sueños se hayan convertido en realidad, al formarnos como ingenieros. Agradecer también a nuestros tutores, Ing. Edisnel Carrazana Castro y el Ing. Ramón Ernesto de Ávila León, por toda la ayuda brindada en el desarrollo de este trabajo, tan complicado. A todos los buenos amigos que de una forma u otra dieron su apoyo incondicional cuando todo se tornaba difícil y parecía imposible poder terminar esta dura tarea, gracias a todos por el gran apoyo brindado.

Dedicatoria

Este trabajo va dedicado a todas las personas que siempre confiaron en mí y dieron un apoyo incondicional en todas las decisiones tomadas y siempre estuvieron presentes en los momentos difíciles para brindarme una mano firme con la cual pudiera levantarme con la frente en alto.

Dedicárselo a mis padres, Hilario Mastrapa Tamayo e Iliana Montero Peña por todo el amor, cariño, apoyo, por educarme y guiarme siempre por un buen camino, por enseñarme a ser un hombre de bien, gracias por confiar siempre en mí y apoyarme incondicionalmente. Agradecerle también a mi hermanita Iliamnis, por apoyarme, siempre entenderme y estar presente, por ser mi hermanita querida, a mis abuelos, Ernesto y Fortuna, por el apoyo incondicional que siempre me han brindado, por su eterno amor, cariño, respeto y siempre estar presente cuando los necesité, y ser como mis padres. A mi tía Damarys por ser otra madre para mí y siempre estar presente, apoyándome y dándome fuerzas y esperanza en todo, a mi familia en general que siempre me ha apoyado, gracias.

También quiero dedicarle este trabajo a Arianna, por haber confiado siempre en mí y apoyarme cuando todo parecía imposible, una parte de este triunfo te pertenece, dedicárselo a Martha y Raicel, por brindarme un apoyo incondicional y aconsejarme cuando lo necesitaba, gracias por todo.

A mis amigos de Holguín, que siempre me apoyaron y confiaron en mí, gracias.

A mis compañeros de aula y mis compañeros de cuarto, que siempre estuvimos unidos y ayudándonos, gracias a todos.

A todos mi buenos amigos, a los que pudieron graduarse y los que no, a todos ellos dedicárselo también.

Resumen

Hoy en día la accesibilidad web es un tema que toma fuerza a nivel mundial, ya que cada vez son más los sitios que cumplen con estos estándares de accesibilidad, como también ha ido en aumento el interés de los desarrolladores por realizar sitios web accesibles para todos. Existe a nivel internacional un grupo de trabajo nombrado Iniciativa de Accesibilidad Web[1] , siendo los responsables de crear las normas de accesibilidad.

El objetivo principal de este trabajo es facilitar la corrección de las pautas de accesibilidad web en el catálogo en línea de la biblioteca de la UCI[2], ya que en su elaboración no se tuvo en cuenta la aplicación de las mismas, y sería muy tedioso si se corrigieran de forma manual. La confección de una herramienta que facilite el trabajo de corrección de las pautas de accesibilidad mejoraría el trabajo de corrección de los desarrolladores en el catálogo en línea.

Palabras clave

Accesibilidad web, catálogo en línea, pautas de accesibilidad web.

1 (WAI) *Web* Accessibility Initiative

2 Universidad de las Ciencias Informáticas

Índice general

INTRODUCCIÓN 1

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA 4

1.1 INTRODUCCIÓN 4

1.2 CONCEPTUALIZACIÓN DE LA ACCESIBILIDAD WEB 4

1.2.1 PAUTAS DE ACCESIBILIDAD AL CONTENIDO EN LA WEB (WCAG) 4

1.2.2 VERSIONES DE LAS WCAG 5

1.2.3 EVALUACIÓN DE LA ACCESIBILIDAD EN SITIOS WEB 6

1.3 SISTEMAS INTEGRADOS DE GESTIÓN BIBLIOTECARIA 8

1.3.1 ANTECEDENTES 8

1.3.2 CONCEPTUALIZACIÓN DE LOS SISTEMAS INTEGRADOS DE GESTIÓN BIBLIOTECARIA 9

1.3.3 CATÁLOGO DE ACCESO PÚBLICO EN LÍNEA 9

1.4 ANÁLISIS DE SOLUCIONES EXISTENTES O SEMEJANTES 10

1.4.1 HERRAMIENTAS DE EVALUACIÓN AUTOMÁTICA 11

1.4.1.1 HERRAMIENTAS DE VALIDACIÓN DE GRAMÁTICA, PROPORCIONADAS POR EL W3C 11

1.4.1.2 HERRAMIENTAS DE EVALUACIÓN DE ACCESIBILIDAD 11

1.4.2 EVALUACIÓN MANUAL 11

1.4.2.1 HERRAMIENTAS DE EVALUACIÓN DE COLOR Y CONTRASTE 12

1.4.3 CONCLUSIONES DEL ANÁLISIS DE SOLUCIONES SEMEJANTES 12

1.5 LENGUAJES, TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS 13

1.5.1 LENGUAJE JAVA 13

1.5.2 LENGUAJE DE MARCADO DE HIPERTEXTO 14

1.5.3 HOJAS DE ESTILO EN CASCADA 14

1.5.4 LENGUAJE DE MODELADO 14

1.5.5 HERRAMIENTA DE MODELADO 15

1.5.6 ENTORNO DE DESARROLLO INTEGRADO 16

1.6 METODOLOGÍA DE DESARROLLO 16

1.7 FUNDAMENTACIÓN DE LA PROPUESTA DE SOLUCIÓN 17

1.8 CONCLUSIONES PARCIALES 17

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA 18

2.1 INTRODUCCIÓN 18

2.2 OBJETO DE AUTOMATIZACIÓN 18

2.2.1 FLUJO DE EVENTOS AUTOMATIZADOS 18

2.3 MODELO DE DOMINIO 19

2.3.1 GLOSARIO DE TÉRMINOS DEL DOMINIO 19

2.4 REQUISITOS FUNCIONALES DEL SISTEMA 20
2.5 REQUISITOS NO FUNCIONALES DEL SISTEMA 20
2.6 MODELO DE CASOS DE USO DEL SISTEMA 21
2.6.1 ACTORES DEL SISTEMA 22
2.6.2 DIAGRAMA DE CASOS DE USO DEL SISTEMA 22
2.6.3 DESCRIPCIONES DE CASOS DE USO 23
2.7 CONCLUSIONES PARCIALES 26
CAPÍTULO 3. ANÁLISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN 28
3.1 INTRODUCCIÓN 28
3.2 ANÁLISIS 28
3.2.1 MODELO DE ANÁLISIS 28
3.2.2 DIAGRAMA DE CLASES DEL ANÁLISIS 29
3.3 DISEÑO 30
3.3.1 DIAGRAMA DE COLABORACIÓN 30
3.3.2 DIAGRAMA DE CLASES DEL DISEÑO 32
3.3.3 ARQUITECTURA DEL SISTEMA 33
3.3.3.1 PATRONES DE DISEÑO GRASP Y GOF 34
3.4 CONCLUSIONES PARCIALES 35
CAPÍTULO 4. IMPLEMENTACIÓN Y PRUEBAS 36
4.1. INTRODUCCIÓN 36
4.2. MODELO DE IMPLEMENTACIÓN 36
4.2.1. DIAGRAMA DE PAQUETES 36
4.2.2. DIAGRAMA DE COMPONENTES 37
4.3. DIAGRAMA DE DESPLIEGUE 37
4.4. ANÁLISIS DE LOS REQUISITOS FUNCIONALES IMPLEMENTADOS EN LA HERRAMIENTA 38
4.5. ANÁLISIS DE REQUISITOS FUNCIONALES NO IMPLEMENTADOS EN LA HERRAMIENTA 39
4.6. PRUEBAS 45
4.6.1. PRUEBAS DE FUNCIONALIDAD 46
4.6.1.1 PRUEBAS DE CAJA NEGRA 46
4.6.1.2 DISEÑO DE CASOS DE PRUEBA 46
4.6.2. RESULTADOS DE LAS PRUEBAS 47
4.7. CONCLUSIONES PARCIALES 47
CONCLUSIONES GENERALES 48
RECOMENDACIONES 49
REFERENCIAS BIBLIOGRÁFICAS 51

BIBLIOGRAFÍA CONSULTADA 54

ANEXOS 57

ANEXO_1: VISTAS DE LA HERRAMIENTA 57

Índice de ilustraciones

Ilustración 1: Diagrama de clases del modelo de dominio 19

Ilustración 2 Diagrama de casos de uso del sistema 23

Ilustración 3: Diagrama de clases del análisis correspondiente al CU Corregir imagen 29

Ilustración 4: Diagrama de clases del análisis correspondiente al CU Corregir color y contraste 29

Ilustración 5: Diagrama de clases del análisis correspondiente al CU Especificar idioma 29

Ilustración 6: Diagrama de clases del análisis correspondiente al CU Corregir aviso de aparición de ventanas emergentes 29

Ilustración 7: Diagrama de clases del análisis correspondiente al CU Establecer unidades de medida 30

Ilustración 8: Diagrama de clases del análisis correspondiente al CU Describir acrónimos 30

Ilustración 9: Diagrama de colaboración correspondiente al CU Corregir imagen 31

Ilustración 10: Diagrama de colaboración correspondiente al CU Corregir color y contraste 31

Ilustración 11: Diagrama de colaboración correspondiente al CU Especificar idioma 31

Ilustración 12: Diagrama de colaboración correspondiente al CU Corregir aviso de aparición de ventanas emergentes 32

Ilustración 13: Diagrama de colaboración correspondiente al CU Establecer unidades de medida 32

Ilustración 14: Diagrama de colaboración correspondiente al CU Describir acrónimos 32

Ilustración 15: Diagrama de clases del diseño 33

Ilustración 16: Niveles de capas 34

Ilustración 17: Diagrama de paquetes del sistema 36

Ilustración 18 Diagrama de paquete "Componente" 37

Ilustración 19 Diagrama de despliegue 38

Ilustración 20 Enlace que no tiene especificado el idioma 38

Ilustración 21 Enlace con el idioma especificado 38

Ilustración 22 Acrónimo sin ser corregido en el código fuente 39

Ilustración 23 Vista de la ventana en la que se visualizan los acrónimos detectados 39

Ilustración 24 Acrónimo corregido en el código fuente 39

Ilustración 25 Diagrama de componentes Visión de la Arquitectura (Valdez, 2012) 40

Ilustración 26 Imagen tratada con jquery tomado de la plantilla doc-head-close.inc del catálogo 41

Ilustración 27 Imagen establecida como fondo en CSS 41

Ilustración 28 Imagen con atributos establecidos con valores en base de datos, tomada de la plantilla opac-advsearch.tmpl del catálogo 42

Ilustración 29 Administración de tipos de ejemplares en el módulo de administración del SIGB 43

Ilustración 30 Color y contraste tomado de Búsqueda simple en el catálogo en línea (http://catalogoenlinea.uci.cu/cgi-bin/koha/opac-main.pl) 44

Ilustración 31 Modificar unidades de medida..... 45

Ilustración 32 Vista de la interfaz principal..... 57

Ilustración 33 Vista de la interfaz principal con el menú desplegado..... 57

Ilustración 34 Vista de la interfaz principal con las opciones que brinda Acciones, desplegado 57

Ilustración 35 Vista del mensaje lanzado después de ser corregido el idioma en los enlaces 58

Ilustración 36 Vista de la interfaz, donde se mostrarán los acrónimos encontrados..... 58

Índice de tablas

Tabla 1: Actores del sistema 22

Tabla 2: Descripción del CU_Corregir imagen 23

Tabla 3: Descripción del CU_Corregir color y contraste 24

Tabla 4: Descripción del CU_Especificar idioma 24

Tabla 5: Descripción del CU_Describir acrónimos 25

Tabla 6: Descripción del CU_Establecer unidades de medida 25

Tabla 7: Descripción del CU_Corregir aviso de aparición de ventanas emergentes 26

Tabla 8: Patrones GRASP 35

Tabla 9: Caso de prueba #1 Prueba de funcionalidad para especificar idioma en los recursos enlazados 46

Tabla 10: Caso de prueba #2 Prueba de funcionalidad para buscar acrónimos sin descripción 47

INTRODUCCIÓN

La accesibilidad web es un tema que ha adquirido gran importancia en el mundo de Internet, cada vez es mayor el número de aplicaciones que cumplen estándares de accesibilidad reconocidos a nivel mundial. Gracias a los esfuerzos de diversas organizaciones internacionales, se ha logrado una mayor difusión y concientización acerca de los beneficios de poseer sitios web accesibles. Sin embargo, aún quedan escépticos que creen que desarrollar un sitio con estas características resulta más costoso y complicado.

Los sistemas de información deben ser útiles para todos, es por eso que, se deben hacer sitios web que sean accesibles para aquellas personas que presentan discapacidades visuales.

Existen millones de personas con discapacidad que no pueden utilizar la web. Actualmente, la mayoría de los sitios web, presentan barreras de accesibilidad, lo que dificulta o imposibilita la utilización de los mismos por personas con discapacidades visuales. Cuanto más software y sitios web accesibles estén disponibles, más personas con discapacidad podrán utilizar la web y contribuir de forma más eficiente (Ruiz, 2004).

Hoy en día se han realizado diversos estudios respecto a la accesibilidad web para débiles visuales, existen diversas herramientas que posibilitan una mayor interacción e integración de la sociedad en el mundo de la informática. Es necesario que personas discapacitadas puedan acceder a las diferentes tecnologías y a la información que nos brinda la web, y que puedan participar de forma activa en el disfrute y consumo de la información teniendo una navegabilidad plena.

En la UCI se han realizado varios trabajos sobre la accesibilidad web, entre ellos se encuentra el trabajo para optar por el título de Ingeniero en Ciencias Informáticas, que lleva por título “Propuesta de accesibilidad en el catálogo en línea para discapacitados visuales”, de Susana Castillo Hernández (Hernández, 2012). Este trabajo permitió obtener los problemas de accesibilidad que presenta el catálogo en línea de la biblioteca de la UCI en el cual no se han tenido en cuenta las pautas de accesibilidad web para débiles visuales. Cuando las personas con discapacidad visual utilizan el SIGB[3] no pueden tener una navegación plena y en ocasiones no obtienen la información buscada.

El catálogo no está en condiciones de ser utilizado por personas con discapacidades visuales, debido a que hay presencia de imágenes sin una descripción detallada, utilización de colores similares en el primer plano y el fondo de las páginas, tamaños de letras con medidas absolutas que no permiten redefinirlo, tampoco se especifica la expansión de cada acrónimo cuando aparece por primera vez. Además cuando se incluye un enlace a otra página, el idioma en el que la misma se visualizará no se especifica y hay

[3] Sistema Integrado de Gestión Bibliotecaria

aparición de ventanas emergentes sin previo aviso. Si las pautas de accesibilidad web para débiles visuales no están bien aplicadas en el catálogo, las herramientas asistivas[4] no pueden ser utilizadas.

Constituye una necesidad la corrección de estos problemas en el catálogo en línea para facilitar la navegación plena a los usuarios con discapacidad visual. Para ello el proyecto encargado del desarrollo del SIGB debe hacer una revisión exhaustiva de cada una de las páginas web del catálogo en línea, para detectar y corregir los errores asociados a la no aplicación de las pautas de accesibilidad. De hacerse manualmente, este trabajo consumiría un tiempo considerable, teniendo en cuenta que el catálogo cuenta con 1917 elementos, entre ellos: páginas HTML, archivos CSS, JavaScript, imágenes, los cuales hay que considerar para la revisión y aplicación de las pautas.

De acuerdo con la situación problemática descrita anteriormente, se plantea la siguiente interrogante como **problema de investigación**: ¿cómo facilitar la aplicación de las pautas de accesibilidad web en el catálogo en línea de la biblioteca de la UCI?

Se define como **objeto de estudio** la accesibilidad web para discapacitados visuales.

Teniendo el **campo de acción** la accesibilidad web para discapacitados visuales en el catálogo en línea de la biblioteca de la UCI.

Como **objetivo general** desarrollar una herramienta informática que facilite la aplicación de las pautas de accesibilidad web en el catálogo en línea de la biblioteca de la UCI.

Como **objetivos específicos** se definieron los siguientes:

- Realizar el marco teórico relacionado con los conceptos de catálogo en línea y accesibilidad web para identificar posibles soluciones al problema planteado.
- Identificar los requisitos funcionales y no funcionales para describir las características de la herramienta a desarrollar.
- Realizar análisis y diseño de la herramienta para traducir los requisitos a una especificación que describa cómo implementar el sistema.
- Implementar las funcionalidades a la herramienta para dar cumplimiento al diseño elaborado previamente.
- Realizar pruebas de calidad a la herramienta desarrollada para validar los resultados obtenidos.

[4] Herramientas, tecnologías, que las personas con discapacidad utilizan para facilitar la ejecución de actividades, y la interacción con el entorno.

Para un mejor desarrollo de la investigación se utilizaron los siguientes métodos científicos:

Métodos Teóricos:

Analítico-Sintético: Se utilizará este método en la identificación de conceptos y definiciones relevantes relacionados con la accesibilidad web, específicamente para débiles visuales, para luego generar una propuesta adecuada a la situación planteada de acuerdo a las exigencias del objeto de estudio y la tecnología estudiada.

Histórico-Lógico: Se realizará un estudio del desarrollo y evolución de herramientas que permitan aplicar pautas de accesibilidad web para débiles visuales, analizando cada una de las principales etapas de su desenvolvimiento y las conexiones históricas fundamentales, permitiendo tener una visión tanto de los principales problemas como de los avances obtenidos en la materia.

Método empírico:

Observación: Se utilizará este método en el análisis del funcionamiento de las distintas herramientas y los resultados que estas muestran, para comprender el funcionamiento de las mismas.

El presente trabajo de diploma se encuentra estructurado en cuatro capítulos, donde cada uno consta de introducción, desarrollo y conclusiones.

Capítulo 1. Fundamentación Teórica. En este capítulo se hace referencia a los principales conceptos relacionados con la accesibilidad web para débiles visuales, los Sistemas Integrados de Gestión Bibliotecaria, los OPAC. Se realiza un estudio del estado del arte de herramientas que permitan aplicar pautas de accesibilidad web para débiles visuales, y se analiza el catálogo en línea de la biblioteca de la UCI. Son expuestas además, la metodología, las tecnologías, los lenguajes y las herramientas utilizadas para desarrollar la solución propuesta.

Capítulo 2. Características del Sistema. En este capítulo se realiza el modelo de dominio, se describen los requisitos funcionales y no funcionales de la herramienta, se definen los casos de uso del sistema.

Capítulo 3. Análisis y diseño de la solución propuesta. En este capítulo se presenta la propuesta de solución mostrando elementos del diseño y la arquitectura del sistema.

Capítulo 4. Implementación y Pruebas. En este capítulo se muestran los diagrama de despliegue y de componentes como elementos fundamentales a modelar en la implementación. Se describen los casos de prueba para cada caso de uso.

Capítulo 1. Fundamentación teórica

1.1 Introducción

En este capítulo se abordarán y describirán las herramientas, tecnologías, así como los conceptos relacionados con estas, para lograr construir la solución propuesta. Se abordarán, además, los lenguajes de programación utilizados y los elementos referentes a la accesibilidad web específicamente para débiles visuales.

1.2 Conceptualización de la accesibilidad web

La accesibilidad web significa que personas con algún tipo de discapacidad van a poder hacer uso de la misma. En concreto, al hablar de accesibilidad web se está haciendo referencia a un diseño web que va a permitir que estas personas puedan percibir, entender, navegar e interactuar con esta, aportando a su vez información.

La accesibilidad web engloba muchos tipos de discapacidades, incluyendo problemas visuales, auditivos, físicos, cognitivos, neurológicos y del habla (Ruiz, 2004).

1.2.1 Pautas de Accesibilidad al Contenido en la Web (WCAG)

Los documentos denominados Pautas de Accesibilidad al Contenido en la Web (por sus siglas en inglés WACG[5]), explican cómo hacer que el contenido web sea accesible para personas con discapacidad. El término "contenido" web normalmente hace referencia a la información contenida en una página web o en una aplicación web, incluyendo texto, imágenes, formularios, sonido, etc.

El W3C[6] es un grupo internacional e independiente que define los protocolos y estándares para la web. Ellos crean las especificaciones de HTML, CSS, etc. Una de las principales iniciativas del W3C es el desarrollo de normas de accesibilidad. El objetivo de la Iniciativa para la Accesibilidad Web (por sus siglas en inglés WAI[7]), es desarrollar los estándares de accesibilidad. Los grupos de trabajo del WAI desarrollan las normas de accesibilidad para los navegadores web, para las herramientas de autor, para las herramientas de evaluación, y para el contenido web, por nombrar algunas. Las normas del Grupo de Trabajo para el Contenido Web se llaman Pautas de Accesibilidad al Contenido en la Web (Mora, 2006).

[5] *Web Content Accessibility Guidelines*

[6] *World Wide Web Consortium*

[7] *Web Accessibility Initiative*

- Pautas:

1. Proporcione alternativas equivalentes para el contenido sonoro y visual.
2. No se base sólo en el color.
3. Utilice marcadores y hojas de estilo y hágalo apropiadamente.
4. Identifique el idioma usado.
5. Cree tablas que se transformen correctamente.
6. Asegúrese de que las páginas que incorporan nuevas tecnologías se transformen correctamente.
7. Asegure al usuario el control sobre los cambios de los contenidos tempo-dependientes.
8. Asegure la accesibilidad directa de las interfaces de usuario incrustadas.
9. Diseñe para la independencia del dispositivo.
10. Utilice soluciones provisionales.
11. Utilice las tecnologías y pautas W3C.
12. Proporcione información de contexto y orientación.
13. Proporcione mecanismos claros de navegación.
14. Asegúrese de que los documentos sean claros y simples.

1.2.2 Versiones de las WCAG

WCAG 1.0

La versión de las Pautas de Accesibilidad Web al Contenido en la Web 1.0 fue un avance importante para lograr que Internet sea más accesible para las personas con discapacidad. Finalizadas en 1999, las WCAG 1.0 proporcionaban 14 directrices y numerosos puntos de control que podían utilizarse para determinar la accesibilidad de una página web. Proporcionaban tres prioridades, niveles de cumplimiento o niveles de adecuación (la medida en que una página web cumple las directrices). La Prioridad 1 o Nivel de adecuación A era un requisito básico para que algunos grupos de usuarios pudieran usar el contenido web. La Prioridad 2 o Nivel de adecuación AA indicaba una mejor accesibilidad y la eliminación de importantes barreras de acceso al contenido. La Prioridad 3 o Nivel de adecuación AAA proporcionaba mejoras a la accesibilidad del contenido web. WCAG 1.0 estaba muy centrado en HTML (Mora, 2006).

Con el paso del tiempo, las WCAG 1.0 comenzaron a mostrar su antigüedad. Conforme las tecnologías web y las tecnologías para las personas con discapacidad avanzaron, la adecuación se hizo más difícil, ya

que algunos puntos de control se volvieron menos importantes y más difíciles de verificar. Por ello se inició el desarrollo de las WCAG 2.0 (Mora, 2006).

WCAG 2.0

Las Pautas de Accesibilidad al Contenido en la Web 2.0 se fundamentan en WCAG 1.0, pero también introducen algunos cambios significativos que es necesario discutir. A un nivel práctico, algunos de los cambios de las WCAG 2.0 son sutiles. Por ejemplo, los formularios todavía requieren etiquetas, las tablas de datos todavía requieren cabeceras y las imágenes todavía requieren un texto alternativo. Los desarrolladores web que ya diseñan sitios web accesibles no tendrán que cambiar mucho sus hábitos. Por otro lado, las WCAG 2.0 representan un cambio sustancial en su filosofía. Los cambios importantes implican que las pautas están centradas en principios más que en técnicas. Esto permite que las pautas sigan siendo relevantes incluso cuando la tecnología cambie. Además, están diseñadas para que la adecuación se pueda verificar de forma fiable. Aunque la medición de una verdadera adecuación puede ser difícil, las pautas están estructuradas para permitir una menor interpretación de lo que una verdadera adecuación significa (Mora, 2006).

El cambio de pautas centradas en las técnicas a pautas centradas en principios dio lugar a un número reducido de ideas de nivel superior o principios. WCAG 1.0 tenía catorce principios en el nivel superior. WCAG 2.0 sitúa únicamente cuatro principios en el nivel superior en virtud de los cuales se organizan pautas más específicas, llamadas criterios de éxito. Cada uno de estos cuatro principios se indica con una sola palabra:

Perceptible

Operable

Comprensible

Robusto

Las WCAG están pensadas principalmente para:

Desarrolladores de contenido web (desarrolladores de páginas web, diseñadores de sitios web, etc.).

Desarrolladores de herramientas de autor para la web.

Desarrolladores de herramientas de evaluación de accesibilidad web (Mora, 2006).

1.2.3 Evaluación de la accesibilidad en sitios web

Cuando se desarrolla o rediseña un sitio web, la evaluación de la accesibilidad de forma temprana y a lo largo del desarrollo permite encontrar al principio problemas de accesibilidad, cuando es más fácil

resolverlos. Técnicas sencillas, como es cambiar la configuración en un buscador, pueden determinar si una página web cumple algunas de las pautas de accesibilidad. Una evaluación exhaustiva, para determinar el cumplimiento de las pautas, es mucho más compleja.

Hay herramientas de evaluación que ayudan a realizar evaluaciones de accesibilidad. No obstante, ninguna herramienta en sí misma puede determinar si un sitio cumple o no las pautas de accesibilidad. Para determinar si un sitio web es accesible, es necesaria la evaluación humana.

El documento "Evaluación de accesibilidad de otros sitios web " proporciona asesoramiento sobre las revisiones preliminares, utilizando técnicas para evaluar de forma rápida algunos de los problemas de accesibilidad que puede presentar un sitio web. También proporciona procedimientos generales y consejos para evaluar el cumplimiento de las pautas de accesibilidad. En la evaluación automática, el primer paso consiste en realizar una comprobación de la gramática de las páginas, tanto del código HTML como de las hojas de estilo, para verificar que están bien formadas y son válidas. La validez gramatical es un requisito de accesibilidad. Es recomendable utilizar las herramientas de validación de código proporcionadas por el W3C.

Para todo diseñador y desarrollador web su principal objetivo es conseguir un sitio atractivo y fácil de navegar, que conquiste a sus visitantes de manera que lo usen a menudo y lo prefieran ante otros. Entonces el camino más rápido es cumplir con los estándares del W3C y las directrices de accesibilidad del WAI. Y para ello es necesario hacer revisiones durante el proceso de desarrollo y una vez creado el sitio. Existen en el mercado e incluso en la web, herramientas que hacen revisiones automáticas de la sintaxis del código fuente, de las hojas de estilo y de la accesibilidad.

Hacer un sitio web accesible puede ser algo sencillo o complejo, depende de muchos factores como por ejemplo, el tipo de contenido, el tamaño y la complejidad del sitio, así como de las herramientas de desarrollo y el entorno.

Muchas de las características accesibles de un sitio se implementan de forma sencilla si se planean desde el principio del desarrollo del sitio web o al comienzo de su rediseño. La modificación de sitios web inaccesibles puede requerir un gran esfuerzo, sobre todo aquellos que no se "etiquetaron" correctamente con etiquetas estándares de XHTML, y sitios con cierto tipo de contenido, como multimedia.

El documento "Plan de implementación de Accesibilidad Web" muestra los pasos básicos para introducir la accesibilidad en un proyecto web. Las Pautas de Accesibilidad al Contenido en la Web y los documentos de técnicas proporcionan información detallada para los desarrolladores (Ruiz, 2004).

1.3 Sistemas Integrados de Gestión Bibliotecaria

1.3.1 Antecedentes

A partir del nacimiento del formato MARC[8] para el almacenamiento de registros bibliográficos, los sistemas de automatización de bibliotecas se consolidaron a finales de la década de los años 1970. En los albores de los años 1980 se establecieron las bases del concepto de sistema integrado. Estos sistemas para la automatización de bibliotecas surgieron como una evolución de los sistemas monofuncionales que se emplearon hasta finales de los años 1970, los cuales tenían por objetivo resolver el problema de la gestión mecánica de funciones que suponían un mayor costo de recursos humanos a las grandes bibliotecas (*Library of Congress y The British Library*). A partir de la década de los años 1980, se comenzó a considerar el momento de los sistemas integrados, completos, centrados y únicos.

A principios de los años 1960 y con miras a automatizar sus actividades bibliotecarias, la Organización Internacional de Trabajo (OIT) creó un sistema denominado ISIS[9]. Dicho sistema operaba en computadoras IBM 360. Una vez que se implementó el sistema, la OIT inició la distribución de ISIS a nivel internacional. Se cubría así el vacío existente en materia de sistemas para el manejo y recuperación de información documental.

Como reflejo de la época en la que surgió ISIS, y como resultado de sus contratiempos tecnológicos, el IDRC[10], motivado por la necesidad de adaptar el ISIS a los nuevos equipos que había desarrollado la industria de la computación, comenzó a trabajar en un software denominado MINISIS. Se iniciaba con esto un rápido desarrollo que llevó en 1975 a la distribución de la versión "A", y para 1978 se liberó la versión "F", que es la que se distribuyó hasta los albores de la década de los años 1990. En 1986 salió al mercado la versión para microcomputadoras denominada MICRO CDS/ISIS y fue donado por la UNESCO a los países miembros, quienes de manera gratuita lo distribuyeron a las bibliotecas interesadas en su adquisición.

En segundo lugar, se encuentra la iniciativa del OCLC[11], nombrado inicialmente así por sus creadores y posteriormente denominado *Online Computer Library Center*. Inició sus actividades en 1967 con el objetivo principal de compartir recursos y reducir la razón del incremento del costo de 50 bibliotecas académicas existentes en el estado de Ohio, Estados Unidos. En el año 1971, comenzó a operar un sistema de catalogación que ofrecía acceso a una base de datos central con el recién creado formato

[8] Registro catalográfico legible por máquina (*Machine Readable Cataloguing Record)* para más información visitar http://www.loc.gov/marc/umbspa/um01a06.html
[9] *Integrated Set of Information System*
[10] *International Development Research Center*
[11] *Ohio College Library Center*

MARC 1 a sus miembros mediante terminales en línea. Por último, el sistema integrado de bibliotecas de la Universidad de Chicago, una institución pionera en la concepción de un sistema integral automatizado para uso bibliotecario y que, como resultado de la solicitud hecha en 1965 por su entonces director Dr. Herman H. Fusster a la National Science Foundation, desarrolló e integró un sistema automatizado para el manejo de sus datos bibliográficos (Navarrete, 2008).

1.3.2 Conceptualización de los Sistemas Integrados de Gestión Bibliotecaria

En la tesis para optar por el título de Licenciado en Biblioteconomía, de Guadalupe González Herrera "Software libre vs. Propietario: una evaluación de sistemas Janium vs. Koha" cita varios conceptos de Sistema Integrado de Gestión Bibliotecaria (SIGB) definidos por diferentes autores (Arriola Navarrete, Moya y García Melero) (Herrera, 2010):

- Es un conjunto de módulos de aplicación integrados en un solo programa y que comparten una base de datos bibliográfica en común y que ayuda a la gestión de procesos y servicios de las unidades de información.

- Sistemas para el proceso automatizado o informático, de información estructurada y no estructurada, sobre actividades y documentos, adaptable a la estructura organizativa de la biblioteca.

- Es un conjunto organizado de recursos humanos que utiliza dispositivos y programas informáticos, adecuados a la naturaleza de los datos que deben procesar, para realizar procesos y facilitar los servicios que permiten alcanzar los objetivos de la biblioteca: almacenar de forma organizada el conocimiento humano contenido en todo tipo de materiales bibliográficos para satisfacer las necesidades informativas, recreativas y de investigación de los usuarios.

Estos conceptos reúnen las características que componen un SIGB, por lo que es importante tenerlos en cuenta para realizar una valoración personal del mismo.

1.3.3 Catálogo de acceso público en línea

Con la aplicación de la tecnología informática a las bibliotecas a mediados del siglo XX, se ve la posibilidad de reproducir los catálogos en distintos soportes y actualizarlos y difundirlos rápidamente (Fernandez, 2008).

Los catálogos son listas ordenadas sistemáticamente de una colección de materiales bibliográficos, que además darán la ubicación de los mismos, siendo su misión doble:

- Identificar los documentos por los datos consignados en su descripción.

- Localizar su ubicación en la biblioteca. Normalmente por medio de una clave o signatura topográfica.

Entre las funciones de los catálogos están:

- Indicar los nombres de personas o entidades con responsabilidad intelectual en la elaboración de una obra (autores, colaboradores, traductores, prologuistas).
- Indicar los títulos de las obras que posee la Biblioteca.
- Indicar las obras que posee sobre una determinada materia, serie, colección.
- Indicar las características sobre el contenido y presentación de una obra (si se compone de más volumen).
- Reunir todas las obras sobre un mismo autor.
- Indicar la ubicación de los libros en los estantes.

Los catálogos se diferencian:

- **por su uso:** internos o públicos**.**
- **por su elaboración:**

✓ **Alfabético de obras y autores:** las fichas bibliográficas están ordenadas alfabéticamente por autores (autor personal, entidad, colaboradores) o por la primera palabra del título de las obras anónimas.

✓ **Alfabético de materias:** las fichas están ordenadas por la materia de la que trata la obra.

✓ **Topográfico:** de carácter interno ordenadas por situación en la estantería.

✓ **Sistemático:** las fichas están ordenadas por una notación determinada de conceptos.

1.4 Análisis de soluciones existentes o semejantes

En este apartado se realiza un estudio de una serie de herramientas que ayudan en la corrección y detección de los problemas de accesibilidad, de las que se analizan sus características con el objetivo de incorporarlas a la solución de este trabajo de diploma.

Existen herramientas que permiten identificar de forma automática problemas de accesibilidad. Las herramientas de validación automática no son suficientes para asegurar que un sitio web es accesible completamente, por lo que se requiere de intervención humana para una revisión detallada, ya que en ocasiones podrían detectarse errores de accesibilidad que en verdad no lo son.

1.4.1 Herramientas de evaluación automática

Estas herramientas son capaces de validar HTML, CSS, de detectar de forma automática estos errores de accesibilidad en sitios web, fragmentos de códigos, es decir, validan un sitio completo o un fragmento de código. Brindando un reporte de los errores encontrados.

1.4.1.1 Herramientas de validación de gramática, proporcionadas por el W3C

- **Validador (X)HTML de W3C**

  Este validador es un servicio online gratuito de validación de código que comprueba la conformidad de los documentos (X) HTML respecto a las gramáticas del W3C y otros estándares (X) HMTL (INTECO, 2008).

- **Validador de CSS de W3C**

  Es una herramienta gratuita para validar las hojas de estilo CSS solas o presentes en documentos (X) HTML, comprobando de esta manera si cumplen las especificaciones del W3C. Existe una versión online y una versión descargable multiplataforma (INTECO, 2008).

1.4.1.2 Herramientas de evaluación de Accesibilidad

Existen herramientas que permiten identificar de forma automática problemas de accesibilidad (INTECO, 2008).

- **TAW (Fundación CTIC).**

  Se trata de la herramienta de evaluación automática de accesibilidad de habla hispana más extendida.

- **Bobby (Watchfire).**

  Es el validador automático de accesibilidad más utilizado a nivel mundial. La comprobación de accesibilidad se basa en las pautas WCAG 1.0.

- **HERA (Fundación Sidar).**

  Herramienta online diseñada para facilitar a los desarrolladores la tarea de la revisión manual de accesibilidad de las páginas web según las WCAG 1.0.

1.4.2 Evaluación manual

Estas herramientas posibilitan la revisión de una forma más específica del código, permitiendo revisar el código generado cuando se navega por la web, o revisando diferentes partes de las páginas, te dan una

evaluación de lugares específicos, permitiendo que sea el usuario el que determine lo que se quiere evaluar, mostrando un análisis de los errores detectados de estos lugares (INTECO, 2008).

- **Web Developer Toolbar:**

  Web Developer es una extensión para Mozilla Firefox que añade una barra de herramientas con varias funciones de utilidad para los desarrolladores web. Esta barra está enfocada hacia el desarrollador web en general, aunque también incluye funciones útiles para la evaluación de la accesibilidad.

- **Firefox Accessibility Extension:**

  Firefox Accessibility Extension es una extensión para Mozilla Firefox que añade una barra de herramientas que incluye opciones que facilita la navegación por los contenidos a los usuarios con discapacidad y también permite realizar comprobaciones de accesibilidad.

- **Web Accessibility Toolbar (Internet Explorer):**

  La barra de herramientas *Web Accessibility Toolbar* es un plug-in para Internet Explorer que ha sido desarrollado para facilitar la evaluación manual de la accesibilidad de las páginas web.

- **HMTL Validator Tidy:**

  Es una extensión para Mozilla que agrega un validador HTML dentro de Firefox. Muestra el número de errores de cualquier página HTML en la barra de estado mientras se navega, además muestra los errores de código al seleccionar la opción “Ver código fuente”.

1.4.2.1 Herramientas de evaluación de color y contraste

- **Colour Contrast Analyser:**
  Comprueba las combinaciones de color de primer plano y color de fondo (contraste).

- **Fujitsu ColorSelector:**
  Es una herramienta que permite determinar la combinación de color de primer plano y color de fondo más accesible.

1.4.3 Conclusiones del análisis de soluciones semejantes

En el estudio realizado anteriormente se identificaron y analizaron una serie de herramientas que son utilizadas para ayudar con la aplicación de pautas de accesibilidad. Estas no cuentan con las funcionalidades necesarias para dar solución a los problemas de accesibilidad existentes en el catálogo en línea, pues solo hacen revisiones automáticas brindando informes acordes a estándares de la W3C.

Además dejan en manos de los usuarios la corrección de los problemas detectados, sin intervenir en el código fuente de las aplicaciones.

Se requiere de una herramienta no solo capaz de detectar problemas de accesibilidad, sino que facilite a desarrolladores del SIGB, la corrección de los mismos en el código fuente de la aplicación, dígase archivos .tmpl (HTML que incluye etiquetas de módulos de Perl), archivos CSS y JavaScript.

1.5 Lenguajes, tecnologías y herramientas utilizadas

Tomando como punto de partida el proyecto encargado en la UCI de la implementación del SIGB para su biblioteca, se propone continuar utilizando los lenguajes, tecnologías y herramientas que se establecieron en un principio por el grupo de analistas, ya que para desarrollar cualquier sistema informático es vital la correcta selección de estos elementos base, de esta manera, se garantiza el óptimo desempeño del sistema. Además, se mencionan las versiones utilizadas de cada elemento en el desarrollo de la solución.

1.5.1 Lenguaje Java

Java posee una curva de aprendizaje muy rápida. Debido a su semejanza con C y C++, y dado que la mayoría de los desarrolladores los conocen, aunque sea de forma elemental, resulta muy fácil aprender Java, fue diseñado como un lenguaje orientado a objetos desde el principio. Los objetos agrupan en estructuras encapsuladas tanto sus datos como los métodos (o funciones) que manipulan esos datos. La tendencia del futuro, a la que Java se suma, apunta hacia la programación orientada a objetos, especialmente en entornos cada vez más complejos Java fue diseñado para crear productos informáticos altamente fiables. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Sus características de memoria liberan a los programadores de una familia entera de errores (la aritmética de punteros), ya que se ha prescindido por completo los punteros, y la recolección de basura elimina la necesidad de liberación explícita de memoria. A nadie le gustaría ejecutar en su ordenador programas con acceso total a su sistema, procedentes de fuentes desconocidas. Así que se implementaron barreras de seguridad en el lenguaje y en el sistema de ejecución en tiempo real.

Java está diseñado para soportar aplicaciones que serán ejecutadas en los más variados sistemas operativos dígase (Unix, Windows, Mac) opera también sobre diferentes estaciones de trabajo y distintas arquitecturas. El compilador de Java genera un formato intermedio indiferente a la arquitectura diseñada para transportar el código eficientemente a múltiples plataformas hardware y software. Debido a estas características se escoge utilizar Java como lenguaje en el desarrollo de esta solución informática para que la misma sea multiplataforma (Marañón, 1999).

1.5.2 Lenguaje de Marcado de Hipertexto

HTML, lenguaje usado para escribir y publicar documentos en la (www)[12]. Es una aplicación de la ISO Standard 8879:1986 SGML[13].

HTML utiliza etiquetas, que consisten en breves instrucciones de comienzo y final, mediante las cuales se determina la forma en la que debe aparecer el texto en el navegador, así como también las imágenes y los demás elementos, en la pantalla del ordenador. Toda etiqueta se identifica porque está encerrada entre los signos menor que y mayor que (<,>), y algunas tienen atributos que pueden tomar algún valor.

HTML más que un lenguaje se ha convertido en un estándar aceptado en todo el mundo por lo que una página HTML se puede visualizar en una navegador de cualquier Sistema Operativo, sus normas las define el W3C (Lapuente, 2005).

1.5.3 Hojas de Estilo en Cascada

CSS es un lenguaje para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores. El principal objetivo del uso de CSS es separar la estructura del documento de su presentación y de esta forma aumentar la organización del código y el riesgo de pérdida de uno u otro.

Cuando se utiliza CSS, la etiqueta HTML no debería proporcionar información sobre cómo va a ser visualizada, solamente marca la estructura del documento. La información de estilo separada fichero CSS, específica cómo se ha de mostrar dicha etiqueta HTML. Algunas de las características que se le pueden definir a esta etiqueta son: color, fuente, alineación del texto, tamaño y posición.

La información de estilo puede ser adjuntada tanto como un fichero separado o en el mismo documento HTML. En este último caso podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo "*style*" (Lapuente, 2005).

1.5.4 Lenguaje de modelado

UML[14], es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el

[12] World Wide Web

[13] Standard Generalized Markup Language

[14] Unified Modeling Language

desarrollo de software. UML es un lenguaje expresivo, claro y uniforme, que no garantiza el éxito de los proyectos pero si mejora sustancialmente el desarrollo de los mismos, al permitir una nueva y fuerte integración entre las herramientas, los procesos y los dominios. Es independiente del proceso, aunque para utilizarlo óptimamente se debe usar en un proceso que sea dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. Es importante resaltar que UML es un lenguaje para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir (Lapuente, 2005).

1.5.5 Herramienta de modelado

Visual Paradigm for UML es una herramienta CASE que soporta el modelado mediante UML y proporciona asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software (Quesada, 2013).

Las ventajas que proporciona *Visual Paradigm for UML* son:

• Dibujo. Facilita el modelado de UML, ya que proporciona herramientas específicas para ello. Esto también permite la estandarización de la documentación, ya que la misma se ajusta al estándar soportado por la herramienta.

• Corrección sintáctica. Controla que el modelado con UML sea correcto.

• Coherencia entre diagramas. Al disponer de un repositorio común, es posible visualizar el mismo elemento en varios diagramas, evitando duplicidades.

• Integración con otras aplicaciones. Permite integrase con otras aplicaciones, como herramientas ofimáticas, lo cual aumenta la productividad.

• Trabajo multiusuario. Permite el trabajo en grupo, proporcionando herramientas de compartición de trabajo.

• Reutilización. Facilita la reutilización, ya que disponemos de una herramienta centralizada donde se encuentran los modelos utilizados para otros proyectos.

• Generación de código. Permite generar código de forma automática, reduciendo los tiempos de desarrollo y evitando errores en la codificación del software.

• Generación de informes. Permite generar diversos informes a partir de la información introducida en la herramienta.

1.5.6 Entorno de desarrollo Integrado

Un entorno de desarrollo integrado IDE[15], es un programa informático compuesto por un conjunto de herramientas de programación, que puede dedicarse a uno o varios lenguajes de programación y ha sido empaquetado como un programa de aplicación compuesto por un editor de código, un compilador, un depurador y un constructor de interfaz gráfica de forma amigable (Nourie, 2005). Como entorno de desarrollo integrado se utiliza NetBeans.

1.6 Metodología de desarrollo

Las metodologías para el desarrollo del software imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Una metodología tiene como principal objetivo aumentar la calidad del software que se produce en todas y cada una de sus fases de desarrollo. Hoy en día existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo y seleccionar la adecuada que posibilite obtener los resultados óptimos es uno de los principales problemas a los cuales se enfrentan los equipos de desarrollo.

En la realización de proyectos de software, es necesario basarse en una metodología de desarrollo de software que ayude a organizar y planificar todo el proceso de desarrollo de software para poder obtener un producto de calidad aceptable y los clientes se sientan satisfechos con el resultado. Un proceso de desarrollo de software es la definición del conjunto de actividades que guían los esfuerzos de las personas implicadas en el proyecto, a modo de plantilla que explica los pasos necesarios para terminar el proyecto, tiene la misión de transformar los requerimientos del usuario en un producto de software. Un proceso define “quién” está haciendo “qué”, “cuándo” y “cómo” para alcanzar un determinado objetivo.

Entre las metodologías se encuentran las llamadas “ágiles”, las cuales permiten desarrollar y obtener rápidamente aplicaciones de menor envergadura y tiempo de desarrollo donde el cliente participa en todo momento a lo largo de todo el proceso, algunos ejemplos son: XP (Programación extrema), Scrum, DSDM (Desarrollo de software dirigido por modelos); mientras que las “tradicionales” son utilizadas para dirigir procesos más grandes y con un tiempo mayor de duración, es el caso de RUP[16].

RUP es una metodología flexible, y fácil de adaptarse a las necesidades de cualquier proyecto, además esta le provee al equipo de desarrollo guías consistentes y personalizadas del proceso. El mismo en conjunto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. RUP utiliza UML para

[15] Integrated Development Environment

[16] Proceso Racional Unificado, por sus siglas en inglés *RUP* (*Rational Unified Process)*.

definir los modelos de software y puede definirse como un modelo que es dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. Se divide en 4 fases: inicio, elaboración, construcción y transición.

Inicio: El objetivo en esta etapa es determinar la visión del proyecto.

Elaboración: En esta etapa el objetivo es determinar la arquitectura óptima.

Construcción: En esta etapa el objetivo es llegar a obtener la capacidad operacional inicial.

Transición: El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores.

Para darle solución al problema planteado en el presente trabajo se optó por RUP como metodología de desarrollo de software, establecida en el proyecto Gestión Bibliotecaria, encargado de desarrollar el SIGB de la biblioteca de la UCI. En este se estableció usar la metodología RUP por el tamaño y complejidad del sistema, por lo que la solución a desarrollar continúa con el uso de la misma.

1.7 Fundamentación de la propuesta de solución

El catálogo en línea de la biblioteca de la UCI no tiene aplicadas las pautas de accesibilidad Web para débiles visuales, por lo que se necesita de una herramienta que facilite aplicar estas pautas. La solución propuesta en este trabajo ofrecería múltiples ventajas al equipo de desarrollo del SIGB ya que automatizaría en parte el proceso de corrección de los errores asociados a la no aplicación de las pautas en el catálogo en línea. Con el uso de esta herramienta se puede disminuir el tiempo dedicado por el equipo de desarrollo a la corrección de errores de accesibilidad.

1.8 Conclusiones parciales

En este capítulo se obtuvieron, los aspectos teóricos a tener en cuenta para la implementación de una herramienta que permita aplicarle pautas de accesibilidad web al catálogo en línea. Además, se comprobó la importancia del estudio de los conceptos relacionados con la accesibilidad web. Se realizó un análisis de las soluciones homólogas para finalmente contar con los elementos necesarios para dar solución a la situación problemática planteada anteriormente.

Capítulo 2. Características del sistema

2.1 Introducción

En el presente capítulo se describe la propuesta de solución para el desarrollo de la herramienta que constituye el objetivo general de la investigación. Además, se detallan los dos primeros flujos de trabajo de la metodología propuesta (RUP) en el capítulo anterior. Estos flujos de trabajos, incluyen de forma general la elaboración del modelo de dominio, la identificación y levantamiento de los requisitos funcionales y no funcionales que requiere el sistema, los casos de uso asociados a estos, las descripciones y los diagramas representativos de los elementos mencionados.

2.2 Objeto de automatización

Teniendo en cuenta que no se han aplicado correctamente las pautas de accesibilidad web para débiles visuales al catálogo en línea, el objeto a automatizar se define como la realización de una herramienta, la cual permita facilitar el proceso de aplicación de estas pautas en el catálogo en línea de la biblioteca de la UCI.

2.2.1 Flujo de eventos automatizados

El sistema va a permitir la corrección de las pautas de accesibilidad web para débiles visuales que no se tuvieron en cuenta en el desarrollo del OPAC de la biblioteca de la UCI. El sistema brinda entre las opciones a aplicar la posibilidad de corregir las imágenes que no tengan descripción, se debe proveer características de las alternativas equivalentes para algunos elementos no textuales usados con frecuencia en el contenido web, con el objetivo de que las personas con discapacidad visual a través del texto alternativo, puedan conocer la información que ofrecen diferentes imágenes, se mostrará un listado con las imágenes que no poseen descripción, al seleccionar una de estas imágenes se mostrará la misma, se dispondrá un cuadro de texto para introducir la descripción, así con cada una de ellas, se corregirá de forma automática.

Otra opción sería la de corregir el color y contraste, primeramente el usuario interactúa con la herramienta y se le brinda la opción de corregir color y contraste, se corregiría el problema de forma automática. Otra opción sería la de especificar el idioma, en este caso sería el español, se seleccionará los .tmpl y los .inc que se quieren corregir, donde se le agregaría de forma automática en las etiquetas que contienen enlaces y no tienen especificado el idioma en el que se visualizará el mismo, el hreflang con el idioma especificado en este caso el español. Otra opción sería la de corregir la aparición de ventanas emergentes sin previo aviso, donde de forma automática se le agregara un aviso al botón donde al dar clic sobre él abrirá una nueva ventana, llevando como mensaje un aviso que se abrirá el contenido en una nueva

ventana. Otra opción sería la de establecer unidades de medida donde el usuario al dar clic en esta opción de forma automática se corregiría este problema. Otra opción sería la de corregir acrónimos, donde el usuario al dar clic en esta opción, se localiza el acrónimo. Se seleccionará los .tmpl y los .inc que se quieren corregir, se mostrarán los posibles acrónimos, donde el desarrollador identificará cuales debe corregir, le agregará la descripción y la herramienta lo corregirá aplicando los cambios.

2.3 Modelo de dominio

"El modelo de dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan las "cosas" que existen o los eventos que suceden en el entorno en el que trabaja el sistema". Con la realización de este modelo se tiene una visión más completa del objeto de estudio. Se realiza un análisis previo al desarrollo de un software, donde al no ser visibles los procesos de negocios, se pueden identificar los conceptos para definir el sistema, además de los conceptos se muestra las relaciones existentes entre ellos (Jacobson, 2000).

A continuación, en la ilustración 1 se muestra el modelo de dominio:

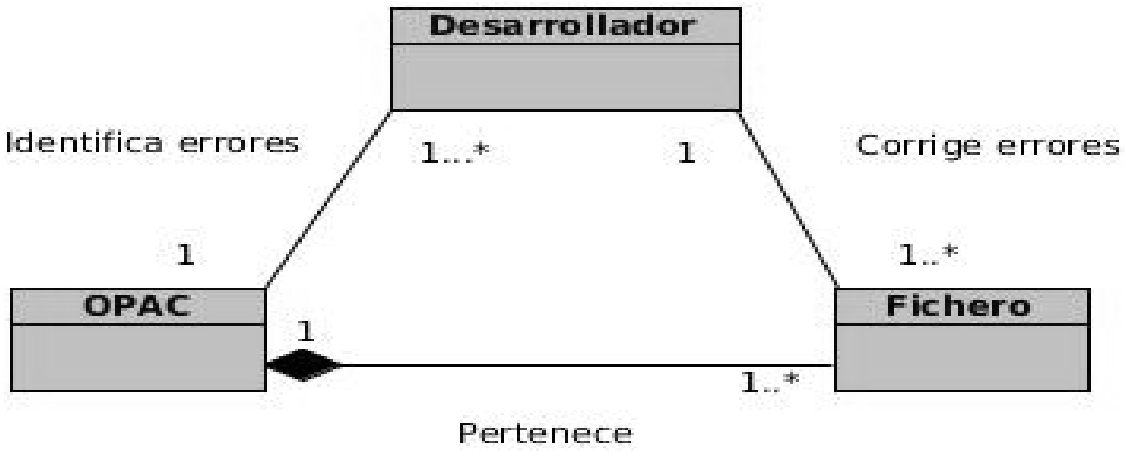

Ilustración 1: Diagrama de clases del modelo de dominio

2.3.1 Glosario de términos del dominio

Desarrollador: Es el encargado de modificar el sitio haciendo uso de la herramienta para aplicar las pautas de accesibilidad web para débiles visuales al OPAC.

OPAC: Catálogo en línea de la biblioteca de la UCI, que será modificado, al aplicarle pautas de accesibilidad web para débiles visuales a los ficheros que lo compone.

Fichero: Es la parte en la que se trabajará directamente para la corrección pautas de accesibilidad web para débiles visuales, corrigiendo los errores detectados en el OPAC.

2.4 Requisitos funcionales del sistema

Teniendo en cuenta los problemas de accesibilidad detectados en el catálogo en línea se proponen las siguientes funcionalidades a implementar en la herramienta.

- RF1 Corregir imágenes que no tienen descripción.
  - ✓ Se corregirán aquellas imágenes que no presenten descripción a través del texto alternativo.
- RF2 Modificar color y contraste.
  - ✓ Se corregirán los colores que no presenten un contraste adecuado entre, el color de la letra y el del fondo.
- RF3 Especificar idioma.
  - ✓ Se corregirán aquellos enlaces que no tengan especificado el idioma en el que se visualizará la misma.
- RF4 Describir acrónimos.
  - ✓ Se corregirán aquellos acrónimos que no presenten la etiqueta especificando que es un acrónimo y el título del mismo.
- RF5 Modificar unidades de medida.
  - ✓ Se corregirán las unidades de medida permitiéndole al usuario redefinir el tamaño de letra.
- RF6 Aparición de ventanas emergentes sin previo aviso.
  - ✓ Se corregirán las apariciones de las ventanas emergentes que no tienen un aviso, mostrando un aviso de que las mismas se abrirán.

De los requisitos funcionales anteriormente planteados algunos no se implementaron debido a su complejidad técnica, dada por la estructura del SIGB. Los mismos serán analizados en detalle en el flujo de trabajo de implementación en el epígrafe 4.3.

2.5 Requisitos no funcionales del sistema

Son requisitos que imponen restricciones en el diseño o la implementación. Son propiedades o cualidades que el producto debe tener.

Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

1. Se conocen como un conjunto de características de calidad, que es necesario tener en cuenta al diseñar e implementar el Software.

2. No son parte de la razón fundamental del producto pero si son necesarios para hacer funcionar el producto de la manera deseada.
3. No modifican la funcionalidad del producto y si añaden funcionalidad al producto.
4. Describen la experiencia del usuario cuando trabaja con el producto y fundamentalmente son las características que se representan por casos de usos.

Como requisitos no funcionales del sistema se tienen los siguientes:

- **Requerimientos de apariencia o interfaz externa.**

La aplicación brindará una interfaz simple y de fácil uso para que el usuario no tenga dificultad al interactuar con el sistema.

- **Requerimientos de Usabilidad.**

La herramienta será utilizada por desarrolladores del SIGB, estos podrán usar los diferentes servicios que brinda, en la corrección de errores de accesibilidad. Dado que los desarrolladores poseen conocimiento suficiente del SIGB el uso de la herramienta se realizará de forma rápida y eficiente.

- **Eficiencia.**

El sistema debe funcionar con un máximo rendimiento pero ajustado a bajas prestaciones de las computadoras, debido a que no todos los organismos poseen tecnología de punta.

- **Requisitos legales, de derecho de autor y otros.**

La Universidad de las Ciencias Informáticas tiene el derecho de autor sobre el Sistema para la aplicación de pautas de accesibilidad web al Sistema Integrado de Gestión Bibliotecaria de la Universidad de las Ciencias Informáticas.

El proyecto utiliza las políticas de software libre donde todas las herramientas que se utilizan son software libre. Para la herramienta *Visual Paradigm* se utiliza la licencia que la universidad compró.

- **Hardware.**

Las PCs clientes deben tener las siguientes características.

- ✓ Memoria RAM 256 MB o superior.
- ✓ Disco duro de 20 GB o más.
- ✓ Procesador de 1.2 GHz o más.

2.6 Modelo de casos de uso del sistema

Los casos de uso son artefactos narrativos que describen, bajo la forma de acciones y reacciones, el comportamiento del sistema desde el punto de vista del usuario. Por lo tanto, establece un acuerdo entre clientes y desarrolladores sobre las condiciones y posibilidades (requisitos) que debe cumplir el sistema. Constituyen una secuencia de interacciones entre el sistema y alguien o algo que usa alguno de sus

servicios. Un caso de uso es iniciado por un actor, quien desde ese momento junto con otros actores, intercambia datos, controla el sistema y participa en la interacción con otros casos de uso. El modelo de casos de uso describe la funcionalidad propuesta del nuevo sistema. Un caso de uso representa una unidad discreta de interacción entre un usuario (humano o máquina) y el sistema (Hernandez, 2007).

2.6.1 Actores del Sistema

A continuación se muestra en la tabla 1, los actores que tendrán interacción con la propuesta de solución:

Tabla 1: Actores del sistema

Actor	Justificación
Desarrollador	Es el actor que trabajará con el sistema a través de las diferentes opciones que brinda el mismo. Primeramente debe cargar el fichero sobre el cual desea trabajar, a través de la opción que se le brinda en “Menú”, luego en “Opciones” será capaz de realizar las opciones que se brindan, realizándose estas de forma semiautomáticas.

2.6.2 Diagrama de casos de uso del sistema

Los diagramas de casos de uso se utilizan para ilustrar los requerimientos del sistema al mostrar cómo reacciona una respuesta a eventos que se producen en el mismo. En este tipo de diagrama intervienen algunos conceptos nuevos: un actor es una entidad externa al sistema que se modela y que puede interactuar con él; un ejemplo de actor podría ser un usuario o cualquier otro sistema. Las relaciones entre casos de uso y actores pueden ser las siguientes (PRESSMAN, 2005):

- Un actor se comunica con un caso de uso.
- Un caso de uso extiende otro caso de uso.
- Un caso de uso usa otro caso de uso.

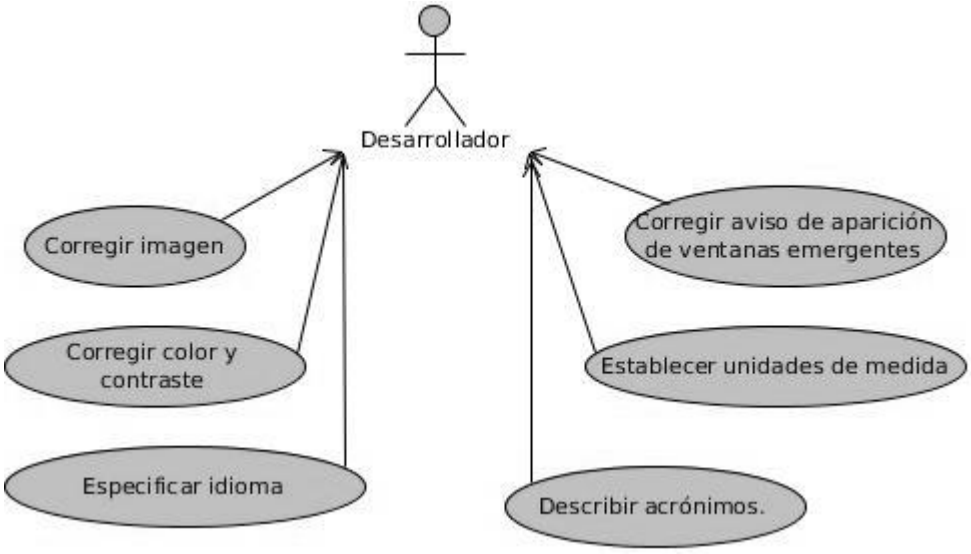

Ilustración 2 Diagrama de casos de uso del sistema

2.6.3 Descripciones de casos de uso

Las descripciones de casos de uso son reseñas textuales del caso de uso, explican los procesos o actividades que tienen lugar en el caso de uso. A continuación se muestran las descripciones textuales de los casos de usos.

Tabla 2: Descripción del CU_Corregir imagen

CU-1	**Corregir imagen**
Actor	**Desarrollador**
Propósito	Realizar descripción de imágenes.
Resumen	Se inicia el caso de uso cuando el desarrollador escoge en Acciones la opción corregir imagen, con la localización de forma automática de las imágenes que no tienen una descripción, es decir que tengan en este caso específico el atributo “alt” vacío o no lo tenga, el actor procede con la descripción de la imagen, posicionándose sobre cada una de las direcciones que se listan, y se mostrará la imagen correspondiente con un campo para introducir la descripción, y se acepta para aplicar los cambios, así se van realizando las descripciones con cada una de las imágenes que no la tenga.
Referencia	RF1
Precondiciones	Debe seleccionar el fichero que se desea revisar para ser corregido.
Poscondiciones	Corrección del fichero.
Prioridad	Crítico
Flujo de eventos	
Flujo básico Corregir imagen	

Actor	Sistema
1. Selecciona la opción describir imagen	2. Muestra el resultado de la búsqueda, mostrándolo en una lista de imágenes.
3. Selecciona una de las imágenes, donde aparece la foto de la misma y un cuadro de texto para introducir la descripción, esta operación se realiza con cada una de las imágenes.	4. Corrige la descripción de la imagen, aplicando estos cambios en los ficheros.

Tabla 3: Descripción del CU_Corregir color y contraste

CU-2	**Corregir color y contraste**
Actor	**Desarrollador**
Propósito	Corregir los colores y contrastes para facilitar una mejor visibilidad en el sitio.
Resumen	Se inicia el caso de uso cuando el desarrollador escoge en Acciones la opción corregir color y contraste, de forma automática se corrigen los errores, con colores y contrastes establecidos.
Referencia	RF2
Precondiciones	Debe seleccionar el fichero que se desea revisar para ser corregido.
Poscondiciones	Corrección del fichero.
Prioridad	Crítico
Flujo de eventos	
Flujo básico Corregir color y contraste	
Actor	**Sistema**
1. Selecciona la opción Corregir color y contraste.	2. Aplica un alto contraste entre los colores de texto y de fondo.

Tabla 4: Descripción del CU_Especificar idioma

CU-3	**Especificar idioma**
Actor	**Desarrollador**
Propósito	En el catálogo en línea, los enlaces no especifican el idioma a utilizar en el recurso, de ahí que se recomiende su uso.
Resumen	Se inicia el caso de uso cuando el desarrollador escoge en Acciones la opción especificar idioma. En los enlaces se identifica el idioma para especificar el lenguaje del recurso enlazado a través del atributo hreflang.
Referencia	RF3
Precondiciones	Debe seleccionar el fichero que se desea revisar para ser corregido.

Poscondiciones	Corrección del fichero.
Prioridad	Crítico
Flujo de eventos	
Flujo básico Especificar Idioma	
Actor	**Sistema**
1. Selecciona la opción Especificar idioma.	2. Localiza los enlaces en los cuales no aparezca el hreflang y se agrega con el idioma definido.

Tabla 5: Descripción del CU_Describir acrónimos

CU-4	**Describir acrónimos**
Actor	**Desarrollador**
Propósito	Describir aquellos acrónimos que tiene el sitio, y que no se tiene definido correctamente el significado.
Resumen	Se inicia el caso de uso cuando el actor escoge en Acciones la opción describir acrónimos, de forma automática se verifica los acrónimos que no tienen la etiqueta “acronym” y el atributo “title”, se muestra una lista de estos, al dar clic sobre cada uno de ellos se mostrará un campo para poner su significado, luego se aplicarían los cambios, y así con cada uno de ellos que estén en la lista.
Referencia	RF4
Precondiciones	Debe seleccionar el fichero que se desea revisar para ser corregido.
Poscondiciones	Corrección del fichero.
Prioridad	Crítico
Flujo de eventos	
Flujo básico Describir acrónimos.	
Actor	**Sistema**
1. Selecciona la opción Describir acrónimos. 3. En caso de que sea un acrónimo, se cambiaría este por su forma extendida o mencionar el significado haciendo uso de la etiqueta “acronym. Debe contener el atributo “title”.	2. Muestra el resultado de la búsqueda. 4. Aplica los cambios.

Tabla 6: Descripción del CU_Establecer unidades de medida

CU-5	**Establecer unidades de medida**
Actor	Desarrollador
Propósito	Posibilitar que el tamaño de la letra pueda ser redefinido, mostrando un tamaño de letra adecuado para mostrar un texto con un mejor

	nivel de acceso.
Resumen	Se inicia el caso de uso cuando el desarrollador escoge en Acciones la opción establecer unidades de medida, se define una unidad de medida específica, esta se aplicaría de forma automática en el sitio.
Referencia	RF5
Precondiciones	Debe seleccionar el fichero que se desea revisar para ser corregido.
Poscondiciones	Corrección del fichero.
Prioridad	Crítico
Flujo de eventos	
Flujo básico Establecer unidades de medida	
Actor	**Sistema**
1. Selecciona la opción Establecer unidades de medida.	2. Aplica la pauta, haciendo uso de la unidad relativa “em”, permitiéndole al usuario redefinir el tamaño de letra, con un escalado de fuente hasta de un %200.

Tabla 7: Descripción del CU_Corregir aviso de aparición de ventanas emergentes

CU-6	**Corregir aviso de aparición de ventanas emergentes**
Actor	Desarrollador
Propósito	Mostrarle al usuario cuando se abrirá una nueva ventana emergente.
Resumen	Se inicia el caso de uso cuando el desarrollador escoge en Acciones la opción corregir aviso de aparición de ventanas emergentes, de forma automática se localizan los target, agregándole un texto donde comunique que se abrirá en una nueva ventana.
Referencia	RF6
Precondiciones	Debe seleccionar el fichero que se desea revisar para ser corregido.
Poscondiciones	Corrección del fichero.
Prioridad	Crítico
Flujo de eventos	
Flujo básico Corregir aviso de aparición de ventanas emergentes	
Actor	**Sistema**
1. Selecciona la opción Corregir aviso de aparición de ventanas emergentes.	2. Aplica el aviso definido para la aparición de una ventana emergente.

2.7 Conclusiones parciales

Con la realización de este capítulo se concluye que:

- Se obtuvo el objeto de automatización y los principales flujos a los que va dirigido para comprender el negocio existente y definir las características a incorporar.
- Se obtuvo por medio del planteamiento de los requerimientos funcionales y no funcionales con los que debe contar el sistema, una mejor comprensión de las funcionalidades que requiere la herramienta y de qué manera serán presentadas al usuario.
- Se obtuvo mediante las descripciones de los casos de uso la interacción del usuario con el sistema y la respuesta del mismo a cada requisito funcional, utilizando un lenguaje comprensible.

Capítulo 3. Análisis y diseño de la propuesta de solución

3.1 Introducción

En el presente capítulo se abordará el flujo de trabajo de Análisis y Diseño. Se expondrá a través de un grupo de artefactos cómo será llevada la solución del sistema que se propone hasta el diseño. Para su modelado se utilizan los diagramas de clases del análisis, de interacción, el diseño de clases.

3.2 Análisis

El análisis del sistema constituye uno de los flujos de trabajos realizados durante el proceso de desarrollo de software. Este se desarrolla fundamentalmente dentro de la fase de elaboración, aunque se tiene en cuenta a lo largo del ciclo de vida del proyecto. El análisis se basa en la obtención de una visión del sistema que se preocupa de ver qué hace el mismo, de modo que sólo se interesa por los requisitos funcionales levantados. El objetivo de este flujo de trabajo es traducir los requisitos a una especificación que describe cómo implementar el sistema (Jacobson, 2000).

3.2.1 Modelo de Análisis

El modelo de análisis ofrece una especificación más precisa de los requisitos que la obtenida en la captura de requisitos. La misma es estructurada de un modo que sea más fácil de comprender, preparar y modificar. En él, se identifican las clases que describen la realización de los casos de uso, los atributos y las relaciones entre ellas. Constituye además, una aproximación al modelo de diseño, sin embargo no contempla el lenguaje de programación que se va a utilizar para el desarrollo de la aplicación, debido a que su objetivo principal es comprender con exactitud los requisitos del software y no explicar su solución (Jacobson, 2000).

Para una mayor comprensión se relacionan las clases que son utilizadas para la elaboración del modelo de análisis:

Clase Interfaz (Cl _Nombre de la clase): Modelan la interacción entre los usuarios y el sistema, es decir, ventanas, formularios, dispositivos, sistemas externos, entre otros.

Clase Controladora (CC _Nombre de la clase): Encapsulan el comportamiento de cada caso de uso y coordinan el trabajo de las clases interfaz y entidad.

Clase Entidad (CE _Nombre de la clase): Modelan toda la información del sistema que posee una vida larga y que puede ser persistente.

3.2.2 Diagrama de clases del análisis

El diagrama de clases del análisis es un artefacto en el que se representan los conceptos en un dominio del problema (Jacobson, 2000). A continuación se muestran los diagramas de clases del análisis correspondiente a cada caso de uso.

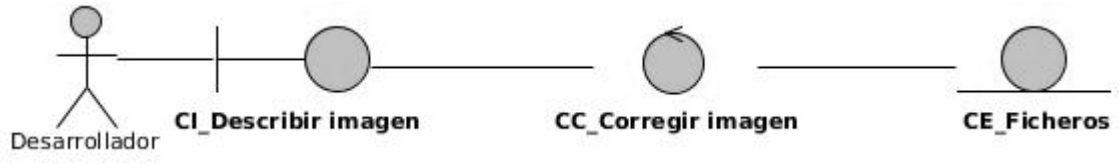

Ilustración 3: Diagrama de clases del análisis correspondiente al CU Corregir imagen

Ilustración 4: Diagrama de clases del análisis correspondiente al CU Corregir color y contraste

Ilustración 5: Diagrama de clases del análisis correspondiente al CU Especificar idioma

Ilustración 6: Diagrama de clases del análisis correspondiente al CU Corregir aviso de aparición de ventanas emergentes

Ilustración 7: Diagrama de clases del análisis correspondiente al CU Establecer unidades de medida

Ilustración 8: Diagrama de clases del análisis correspondiente al CU Describir acrónimos

3.3 Diseño

El diseño es un refinamiento del análisis, este tiene en cuenta los requisitos no funcionales que en definitiva definen el cómo cumple el sistema sus objetivos. El diseño debe ser suficiente para que el sistema pueda ser implementado sin ambigüedades. De hecho, cuando la precisión del diseño es muy grande, la implementación puede ser hecha por un generador automático de código. El flujo de trabajo está compuesto por los diagramas de interacción, que pueden ser de colaboración o secuencia y los diagramas de clases del diseño (Jacobson, 2000).

3.3.1 Diagrama de colaboración

Un diagrama de colaboración es una forma de representar iteraciones entre objetos. Es un tipo de diagrama que muestra las interacciones entre objetos organizados y enlazados entre ellos. Consiste en mostrar como las instancias específicas de las clases trabajan juntas para conseguir un objetivo común, en especificar un contrato entre objetos, implementar las asociaciones del diagrama de clases mediante el paso de mensajes de un objeto a otro, dicha implementación es llamada “enlace” (Rojas, 2008).

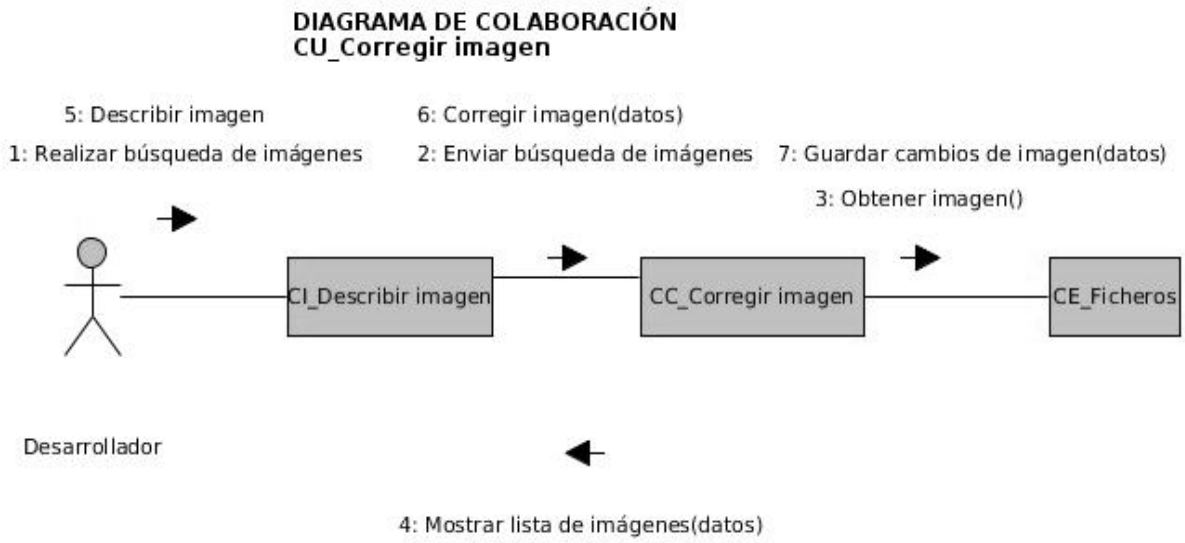

Ilustración 9: Diagrama de colaboración correspondiente al CU Corregir imagen

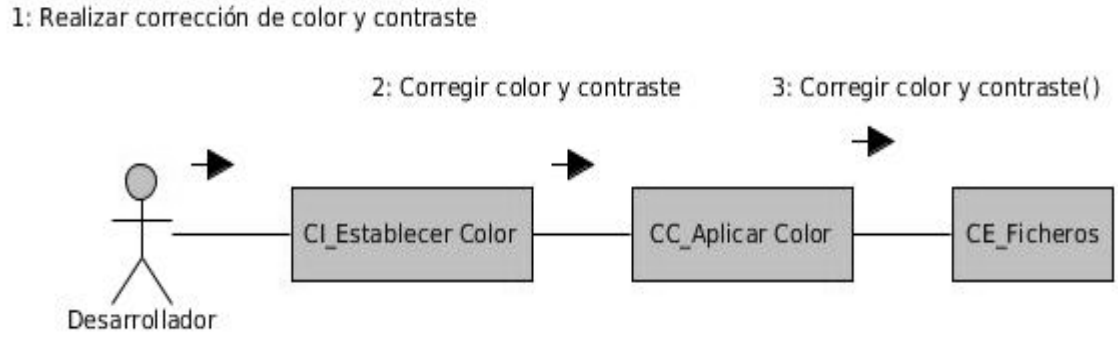

Ilustración 10: Diagrama de colaboración correspondiente al CU Corregir color y contraste

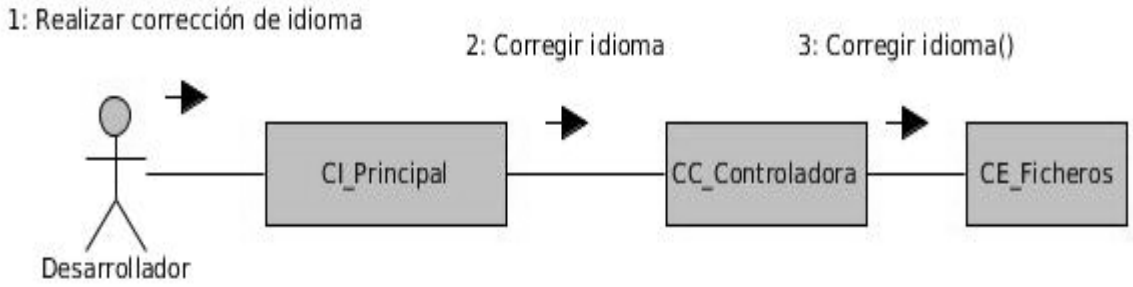

Ilustración 11: Diagrama de colaboración correspondiente al CU Especificar idioma

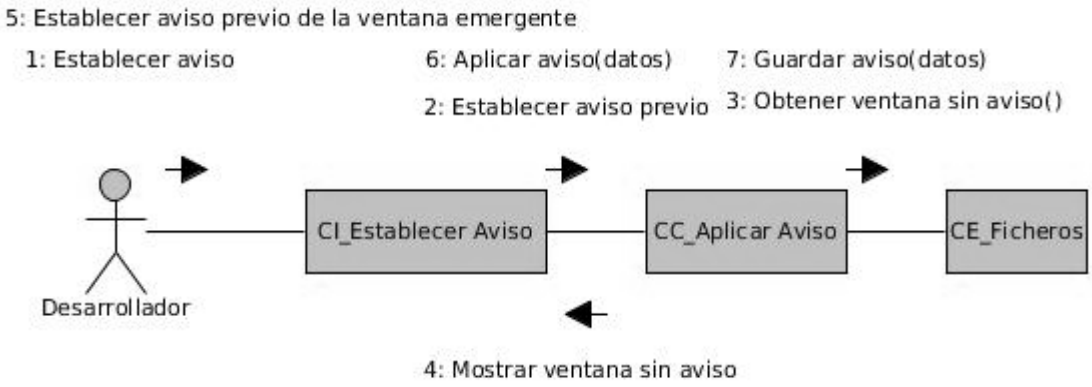

Ilustración 12: Diagrama de colaboración correspondiente al CU Corregir aviso de aparición de ventanas emergentes

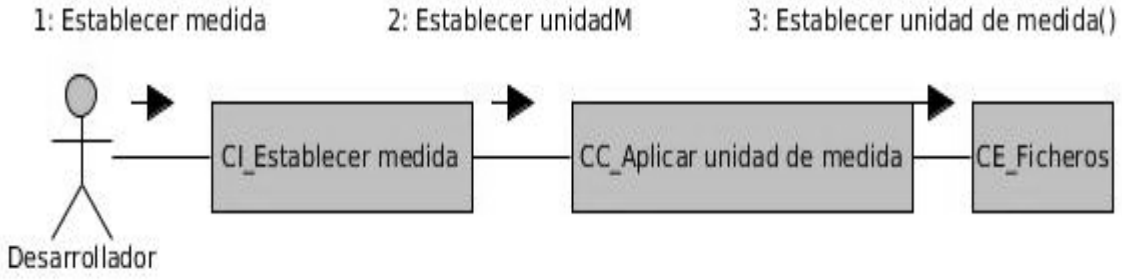

Ilustración 13: Diagrama de colaboración correspondiente al CU Establecer unidades de medida

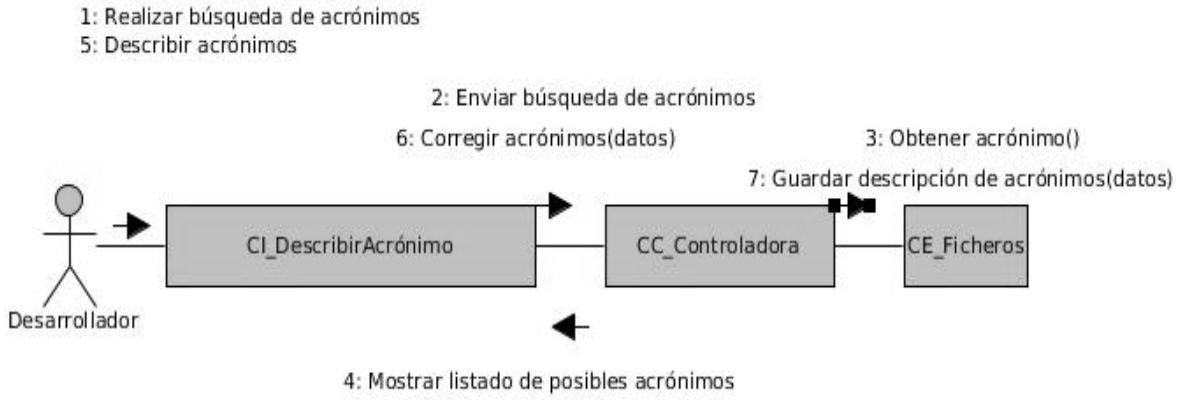

Ilustración 14: Diagrama de colaboración correspondiente al CU Describir acrónimos

3.3.2 Diagrama de clases del diseño

El Diagrama de Clases de Diseño describe gráficamente las especificaciones de las Clases de Software y las Interfaces en una aplicación. Normalmente contiene la siguiente información: Clases, asociaciones y

atributos. Interfaces, con sus operaciones y constantes. Información sobre los tipos de atributos. Navegabilidad. Dependencia (Quesada, 2013).

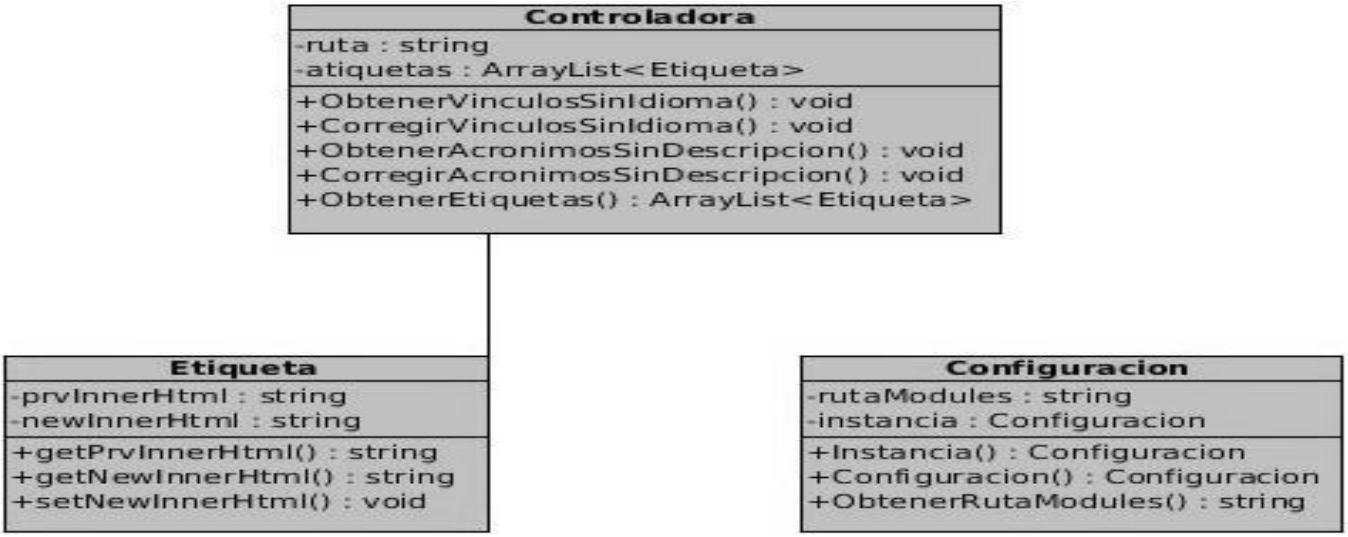

Ilustración 15: Diagrama de clases del diseño

3.3.3 Arquitectura del sistema

La necesidad de contar con porciones de la aplicación que se puedan “intercambiar” sin tener que modificar el resto de la aplicación es lo que impulsa el desarrollo en capas. Por lo que se cuenta con la utilización del patrón arquitectónico: N Capas haciendo uso de dos capas, como se muestra en la siguiente ilustración.

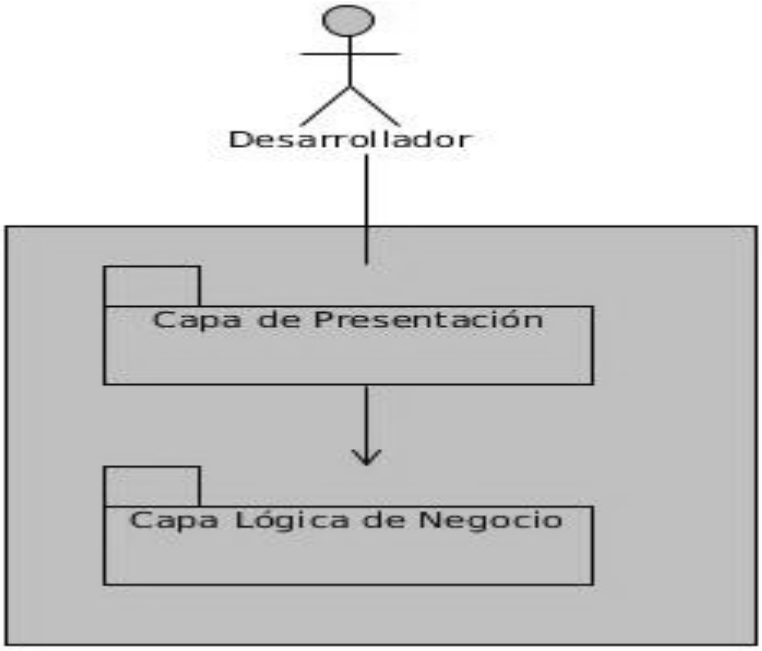

Ilustración 16: Niveles de capas

La Capa de Presentación consiste en una interfaz gráfica que reúne los aspectos de software enfocados a la interacción con el desarrollador.

La Capa Lógica de Negocio reúne los aspectos de software que serán los que automaticen los procesos de negocio. En esta capa se realiza el acceso a datos contenidos en los ficheros, para ejecutar las operaciones y enviar el resultado procesado a la capa de presentación.

Algunos tipos de arquitectura son más recomendables que otras para ciertas tecnologías, en este caso se utilizan los patrones de diseño GRASP[17] y GoF[18]. Los patrones GRASP representan los principios básicos de la asignación de responsabilidades a objetos, expresados en forma de patrones (Larman, 1999).

Los patrones GoF se clasifican en 3 categorías basadas en su propósito: creacionales, estructurales y de comportamiento (Larman, 1999).

3.3.3.1 Patrones de diseño GRASP y GoF

Los patrones de diseño GRASP que se utilizan en el desarrollo de esta solución se mencionan a continuación.

[17] *General Responsibility Assignment Software Patterns, en español* Patrones Generales de Software para Asignar Responsabilidades

[18] Gang of Four, en español Pandilla de los Cuatro, formada por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides

- **Experto** soluciona el problema de asignar una responsabilidad de forma general, al recomendar para esta función a la clase que maneje la información necesaria.
- **Alta Cohesión** aconseja mantener la clase lo más cohesionada posible para controlar la complejidad de la misma.
- **Controlador** ayuda a decidir quién se encarga de administrar un evento del sistema.

La siguiente tabla muestra cómo se evidencia el uso de los diferentes tipos de patrones GRASP, mencionados anteriormente.

Tabla 8: Patrones GRASP

Patrón	Ejemplo de uso.
Experto	La clase Controladora, es la encargada de manejar todos los datos relacionados a la obtención de elementos del código fuente analizado, que se almacenarán en una lista.
Alta Cohesión	En la clase Controladora, se tienen los métodos necesarios para hacer funcionar el sistema.
Controlador	Se centralizan las actividades fundamentales en la clase Controladora.

Los patrones de diseño GoF que se utilizan en el desarrollo de esta solución se mencionan a continuación.

- **Estructurales**

Módulo: Agrupa varios elementos relacionados, como clases y métodos, utilizados globalmente, en una entidad única.

3.4 Conclusiones parciales

Con la realización de este capítulo se concluye que:

- Se obtuvo una visión general del sistema a través de las actividades desarrolladas en el flujo de análisis y diseño.
- Se obtuvo la representación lógica del negocio mediante los diagramas de análisis del sistema, diagramas de colaboración y de clases del diseño.
- Se crean las condiciones necesarias para el comienzo del flujo de trabajo de implementación del sistema.

Capítulo 4. Implementación y Pruebas

4.1. Introducción

Este capítulo expone lo referente a los flujos de trabajo Implementación y Pruebas, los cuales son determinantes en el proceso de desarrollo de software. Para ello, se modela el diagrama de componentes, haciendo una representación de la implementación de las clases de diseño en términos de componentes y cómo estos se organizan de acuerdo con los nodos específicos en el modelo de despliegue. Además, se realiza un análisis de los casos de prueba, teniendo en cuenta los datos de entrada, resultados esperados y condiciones que deben cumplirse mientras se ejecuta el caso de prueba, con el objetivo de comprobar los errores que pueda tener el sistema, corregirlos y obtener un óptimo funcionamiento.

4.2. Modelo de Implementación

En el flujo de trabajo de implementación, a partir del resultado del diseño, se implementa el sistema en términos de componentes, es decir, clases para lo cual se realiza el modelo de implementación.

4.2.1. Diagrama de Paquetes

Un paquete es un mecanismo utilizado para agrupar elementos UML.

Permite organizar los elementos modelados con UML, facilitando de ésta forma el manejo de los modelos de un sistema complejo.

Permiten dividir un modelo para agrupar y encapsular sus elementos en unidades lógicas individuales. Se pueden utilizar para plantear la arquitectura del sistema a nivel macro (Gutiérrez, 2007).

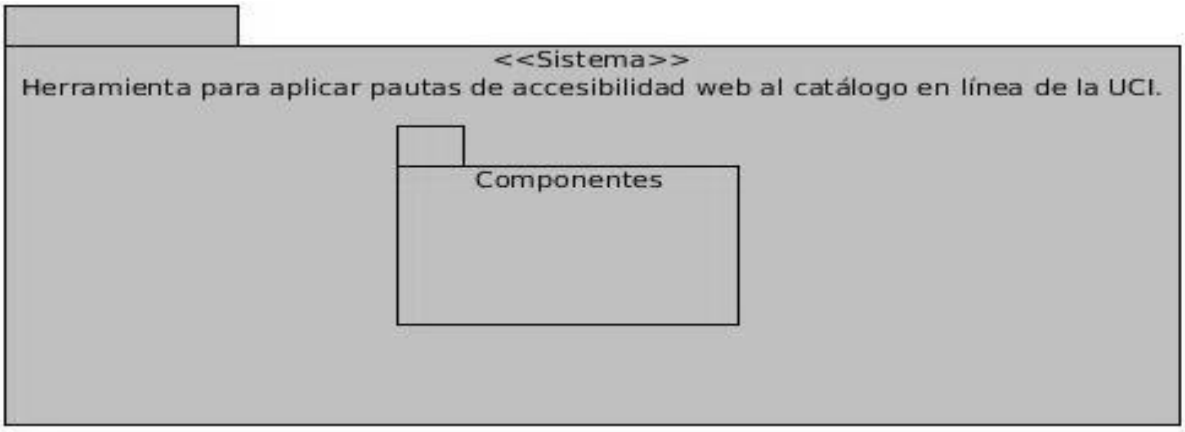

Ilustración 17: Diagrama de paquetes del sistema

4.2.2. Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y su relación, mostrando las dependencias lógicas entre componentes software. El diagrama de componentes hace parte de la vista física de un sistema, la cual modela la estructura de implementación de la aplicación por sí misma, su organización en componentes y su despliegue en nodos de ejecución. La vista de implementación se representa con los diagramas de componentes (Pressman, 1998).

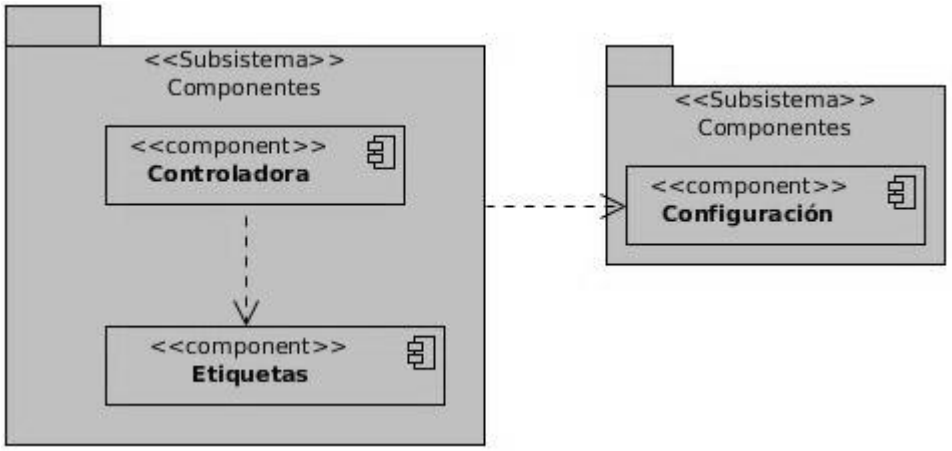

Ilustración 18 Diagrama de paquete "Componentes"

4.3. Diagrama de despliegue

El diagrama de despliegue se utiliza para modelar la configuración de los elementos de procesado en tiempo de ejecución y de los componentes, procesos y objetos de software que viven en ellos. Se modelan los nodos físicos y las asociaciones de comunicación que existen entre ellos (Pressman, 1998). La presente solución es una aplicación para entornos de escritorio, que no interactúa en ningún momento con una base de datos. Por lo que será ejecutada en una estación de trabajo. La Ilustración 19 muestra el modelo de despliegue de la solución.

Nodo PC:

Representa las computadoras que utilizarán los desarrolladores para interactuar directamente con la herramienta. La herramienta será utilizada en cada puesto de trabajo para facilitar la corrección de las pautas de accesibilidad, en los ficheros sobre los cuales se trabaje.

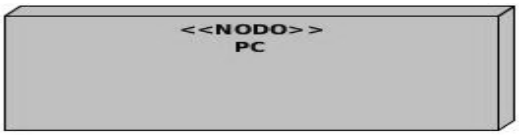

Ilustración 19 Diagrama de despliegue

4.4. Análisis de los requisitos funcionales implementados en la herramienta

De los requisitos funcionales propuestos, sólo dos se pudieron implementar. Primeramente se debe seleccionar el fichero que se desea revisar para aplicarle las pautas. Para ver las vistas de la aplicación ir al **Anexo_1**.

- **RF3 Especificar idioma.**

Se especifica el idioma en los enlaces que no lo presentan a través del uso del hreflang, permitiendo que al utilizarse tecnologías asistivas, estas puedan funcionar correctamente al saber el idioma en el que se visualizara el contenido de este enlace. Actualmente los enlaces en el catálogo se encuentran como se muestran en la siguiente ilustración.

```
<!-- TMPL_IF NAME="noadminemail" -->
<p>Ha ocurrido un error al enviar su mensaje al administrador. Por favor, visite la biblioteca para actualizar
sus datos personales. </p>
<p><a href="/cgi-bin/koha/opac-user.pl">Regresar a su resumen.</a></p>
<!-- TMPL_ELSE -->
<p><!-- TMPL_VAR NAME="errormessage" --></p>
```

Ilustración 20 Enlace que no tiene especificado el idioma

Después de aplicarse la corrección de la pauta se emplea el atributo hreflang como se muestra en la siguiente ilustración.

```
15 <!-- TMPL_IF NAME="noadminemail" -->
16 <p>Ha ocurrido un error al enviar su mensaje al administrador. Por favor, visite la biblioteca para
   actualizar sus datos personales. </p>
17 <p><a href="/cgi-bin/koha/opac-user.pl" hreflang="es-ES" >Regresar a su resumen.</a></p>
18 <!-- TMPL_ELSE -->
19 <p><!-- TMPL_VAR NAME="errormessage" --></p>
```

Ilustración 21 Enlace con el idioma especificado

- **RF4 Describir acrónimos.**

Se corrigen los acrónimos a través del uso de la etiqueta acronym y el title. Se muestra un listado con los posibles acrónimos, donde se seleccionara uno a uno, y se le irá poniendo la descripción correspondiente, en la siguiente ilustración se muestra como se encuentra un acrónimo en el código fuente.

```
    <div id="bd">
<!-- TMPL_INCLUDE name="masthead.inc" -->ISBD

<div id="yui-g">
```

Ilustración 22 Acrónimo sin ser corregido en el código fuente

La siguiente ilustración muestra la vista, de cómo es mostrado en la herramienta, los acrónimos detectados en el código fuente, agregándole la descripción.

Ilustración 23 Vista de la ventana en la que se visualizan los acrónimos detectados

Queda corregido en el código fuente como se muestra a continuación.

```
  <div id="bd">
<!-- TMPL_INCLUDE name="masthead.inc" --><acronym title="Estandar Internacional de Descripción Biográfica"
>ISBD</acronym>

<div id="yui-g">
```

Ilustración 24 Acrónimo corregido en el código fuente

4.5. Análisis de requisitos funcionales no implementados en la herramienta

De los requisitos propuestos, una vez comenzado el flujo de implementación, los que se muestran a continuación no fueron implementados debido a su complejidad técnica y a las características y estructura particulares del SIGB.

Para un mejor entendimiento, es útil tener en cuenta que la herramienta propuesta trata de detectar los problemas de accesibilidad directamente en el código fuente de la aplicación, compuesto de archivos perl con extensión .pl, plantillas .tmpl que contienen HTML y etiquetas del módulo HTML::Template, las cuales incluyen archivos externos de JavaScript y CSS. Las páginas del catálogo en línea se generan de forma dinámica leyendo configuraciones y valores de la base de datos. Para un mejor entendimiento de la estructura del sistema se presenta la siguiente figura:

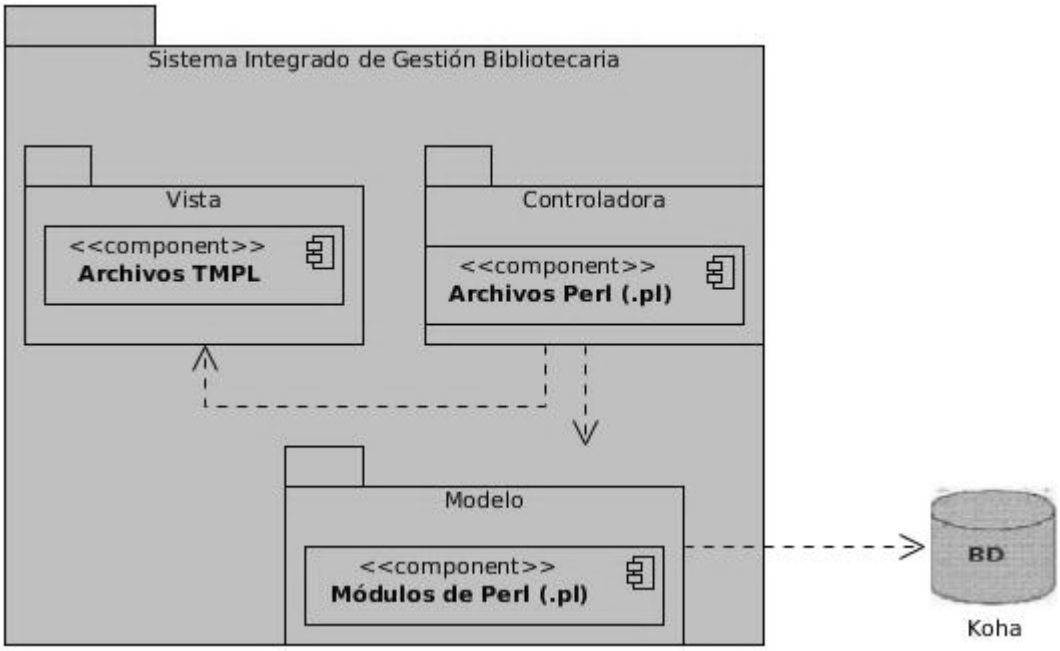

Ilustración 25 Diagrama de componentes Visión de la Arquitectura

- **RF1 Corregir imágenes que no tienen descripción.**

En el SIGB la forma de presentar las imágenes no está estandarizada, pudiendo encontrarlas como fondo en estilos CSS, o siendo leídas de bases de datos y declaradas en las plantillas .tmpl, también tratadas dinámicamente con JavaScript haciendo uso de librerías jquery. A continuación se ven algunos ejemplos:

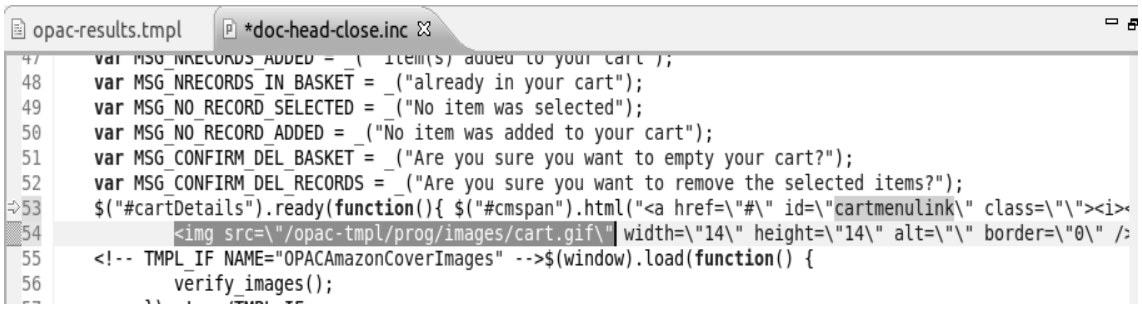

Ilustración 26 Imagen tratada con jquery tomado de la plantilla doc-head-close.inc del catálogo

El elemento determinante de la accesibilidad que se analiza en el caso de las imágenes es la presencia del atributo alt el cual es necesario establecer en donde hallan imágenes relevantes que necesiten una descripción y no la tengan. En las figuras anteriores la imagen tratada con jquery es solo un ejemplo de las diversas formas en que las imágenes pueden ser generadas dinámicamente con JavaScript lo que complejiza la tarea de incrustar código fuente sobre una línea con características particulares de la sintaxis de jquery. Dada la complejidad y que no hay una estructura estándar de generar las imágenes desde JavaScript, se propone que la corrección sea hecha por desarrolladores.

Ilustración 27 Imagen establecida como fondo en CSS

En el caso de las imágenes generadas desde archivos .css como en la figura anterior no es posible establecer el atributo alt, ya que no constituye parte de la especificación de CSS sino del lenguaje HTML y la imagen no se encuentra definida en la plantilla .tmpl

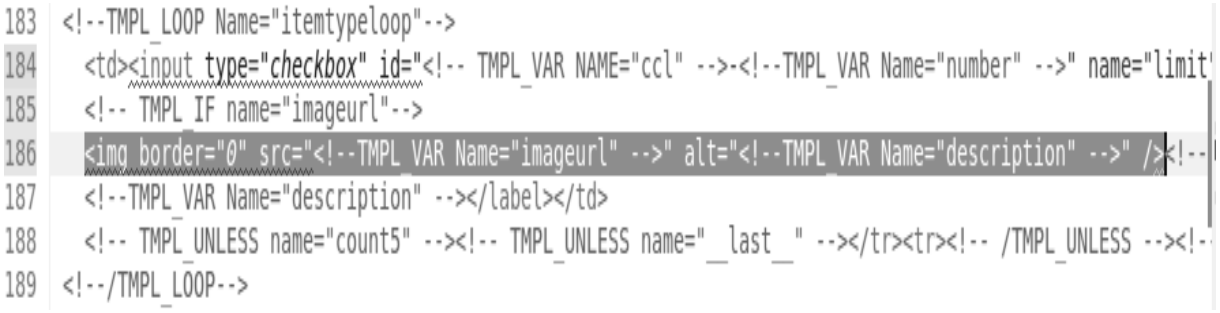

Ilustración 28 Imagen con atributos establecidos con valores en base de datos, tomada de la plantilla opac-advsearch.tmpl del catálogo

Otro caso lo constituyen las imágenes donde los valores de los atributos se leen de la base de datos y no están presentes en el código fuente de la aplicación. En la figura anterior se puede observar que los atributos src y alt de la etiqueta img se establecen a través de src="<!--TMPL_VAR Name="imageurl" -->" alt="<!--TMPL_VAR Name="description" -->", donde la etiqueta TMPL_VAR pertenece al módulo de Perl HTML::Template, que permite al archivo .pl enviar los valores leídos de la base de datos a la plantilla .tmpl para mostrar su contenido en el navegador, por lo que no es factible cambiar estos valores en el código fuente de la aplicación. La herramienta en estos casos pudiera detectar que no hay problemas de accesibilidad ya que el atributo alt tiene un valor asignado, pero en realidad su valor depende de lo que está almacenado en base de datos y pudiera estar vacío. La mayor parte de las imágenes en el catálogo no tienen especificada la ruta (src) y el atributo alt en el código de la aplicación sino en la base de datos.

El SIGB, en su módulo de administración contempla la edición de las descripciones de estas imágenes asociadas a tipos de ejemplares en el catálogo, por lo que a parte de la complejidad a la hora de implementar esta funcionalidad, en el módulo de administración se concibieron las configuraciones para darle una descripción. En la siguiente figura se observan estas opciones.

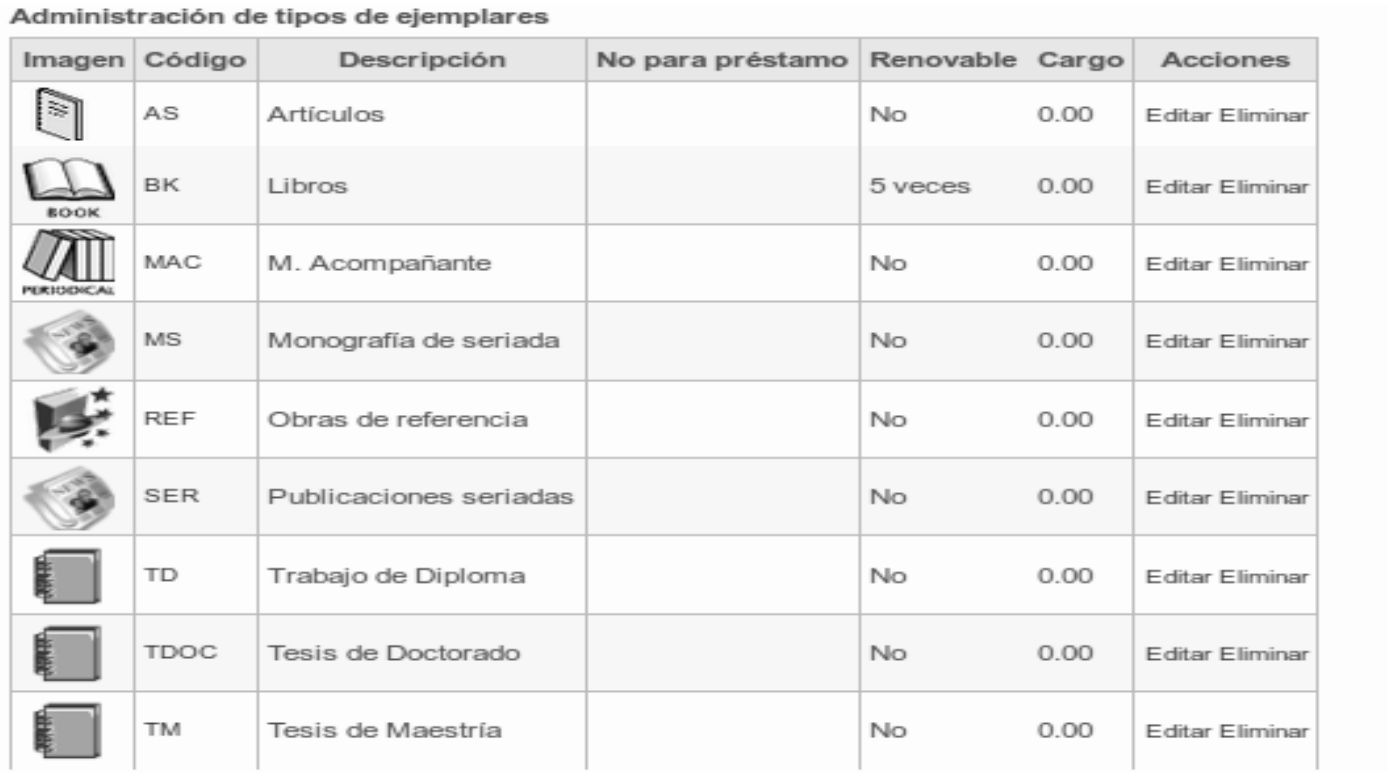
Administración de tipos de ejemplares

Imagen	Código	Descripción	No para préstamo	Renovable	Cargo	Acciones
	AS	Artículos		No	0.00	Editar Eliminar
BOOK	BK	Libros		5 veces	0.00	Editar Eliminar
PERIODICAL	MAC	M. Acompañante		No	0.00	Editar Eliminar
	MS	Monografía de seriada		No	0.00	Editar Eliminar
	REF	Obras de referencia		No	0.00	Editar Eliminar
	SER	Publicaciones seriadas		No	0.00	Editar Eliminar
	TD	Trabajo de Diploma		No	0.00	Editar Eliminar
	TDOC	Tesis de Doctorado		No	0.00	Editar Eliminar
	TM	Tesis de Maestría		No	0.00	Editar Eliminar

Ilustración 29 Administración de tipos de ejemplares en el módulo de administración del SIGB

Se concluye que no es factible implementar la funcionalidad asociada a las imágenes, ya que en algunos de los casos la complejidad es alta y la estructura del sistema no lo permite.

- **RF2 Modificar color y contraste.**

En el SIGB la forma de asignar color a los textos es diversa y no está estandarizada, pudiendo encontrarse que el color es establecido en archivos CSS, en las plantillas .tmpl y también dinámicamente con JavaScript. Todo ello dificulta implementar de forma automática en la herramienta la corrección de estos problemas, dado que habría que implementar cada caso específico que se detecte.

Las herramientas analizadas en el Capítulo 1 permiten detectar los problemas de accesibilidad asociados al color y contraste, sin embargo lo realizan sobre código HTML generado y presentado en el navegador, por lo que no fueron útiles para implementar esta funcionalidad en la herramienta que debe detectar y corregir los problemas en el código fuente de una aplicación muy específica.

Otra cuestión la constituye el hecho de que los colores que se aplican en archivos CSS afectan la presentación de varios elementos HTML a la vez, por lo que de tratar corregir un problema de accesibilidad por problemas de contraste en un elemento, pudiera afectar la presentación de otros elementos que utilizan el mismo estilo y no tienen ningún problema.

Ilustración 30 Color y contraste tomado de Búsqueda simple en el catálogo en línea (http://catalogoenlinea.uci.cu/cgi-bin/koha/opac-main.pl)

De igual manera, como las páginas que genera el SIGB se construyen a partir de plantillas, las mismas se conforman generando archivos .tmpl, que hacen inclusiones en tiempo de ejecución de otros archivos con extensión .inc, por lo que aunque el contexto en que se presentan los elementos HTML es el mismo al ser visualizados en el navegador, no sucede así cuando se analiza el código fuente donde los elementos que tienen problemas de color y contraste están ubicados en archivos diferentes y las propiedades que determinan su presentación están dispersas en varios archivos CSS.

La resolución de este tipo de problema, dada su complejidad, no es factible su implementación en la herramienta sino que constituye una tarea de los desarrolladores de software del SIGB, que conocen los elementos particulares de su estructura, las pautas de desarrollo utilizadas y la ubicación de los elementos HTML y los estilos asociados.

- **RF5 Modificar unidades de medida.**

Las WCAG 2.0 son más flexibles en cuanto a la manera en que los autores pueden ofrecer a los usuarios la característica de escalado de fuentes, admitiéndose otros métodos como proporcionar botones para ampliar la fuente u ofrecer varias versiones de las hojas de estilo (Hernández, 2012). No obstante, se recomienda usar unidades relativas como “em” o porcentajes, ya que en esta versión son consideradas una técnica de suficiencia para cumplir el criterio de éxito 1.4.4. Por otro lado y dado que en el contexto de las WCAG 2.0 la tecnología usada para presentar texto podría no ser HTML + CSS, se admite como válido el uso de tecnologías que soporten de forma nativa el escalado de fuentes; en caso contrario, el autor es responsable de proporcionar métodos para permitir al usuario aumentar los textos hasta un 200 % sin que se pierda alguna funcionalidad (Hernández, 2012).

De igual manera, como las páginas que genera el SIGB se construyen a partir de plantillas, las mismas se conforman generando archivos .tmpl, que hacen inclusiones en tiempo de ejecución de otros archivos con extensión .inc, por lo que aunque el contexto en que se presentan los elementos HTML es el mismo al ser

visualizados en el navegador, no sucede así cuando se analiza el código fuente donde no se tiene acceso a las propiedades de los elementos que conforman el HTML de la página generada por lo que se recomienda que sea el desarrollador el encargado de permitirle al usuario redefinir el tamaño de letra a través de nuevas clases que se le añadan al SIGB.

Se propone como alternativa añadir atributos de accesibilidad para controlar el tamaño de las letras al sitio de forma manual y que permita a los usuarios escoger el tamaño preferido, como se muestra la siguiente figura, en que se puede hacer uso de los elementos Aa^{+}

Ilustración 31 Modificar unidades de medida

- **RF6 Aparición de ventanas emergentes sin previo aviso.**

El elemento determinante de la accesibilidad que se analiza en el caso de aparición de ventanas emergentes sin previo aviso, es la presencia de un aviso que se muestre cuando se utilice el atributo target, permitiéndole saber al usuario que se abrirá una nueva ventana emergente. Debido a que los enlaces son tratados dinámicamente con JavaScript haciendo uso de librerías jquery, en la plantilla no se puede identificar correctamente el uso de este atributo para agregarle el texto con el aviso adecuado para la nueva ventana que se abrirá. Dado también por la complejidad de la estructura, que no está estandarizada debido a las especificidades de Perl y la utilización de Koha que no presenta una arquitectura estandarizada.

4.6. Pruebas

Las pruebas son un grupo de actividades planeadas con anticipación y que se realizan de forma sistemática. Además son el último bastión para la evaluación de la calidad del software y para la detección de errores en el mismo (PRESSMAN, 2005).

4.6.1. Pruebas de funcionalidad

Las pruebas de funcionalidad examinan si una aplicación cumple con los requisitos funcionales propuestos por el cliente, también tienen como objetivo revelar los problemas y errores que poseen las funcionalidades. Además se debe verificar la validación de los datos y es importante realizarlas en cualquier fase de desarrollo del software (D, 2005).

Para llevar a cabo las pruebas de funcionalidades se utiliza el método de pruebas de caja negra.

4.6.1.1 Pruebas de caja negra

Las pruebas de caja negra, también denominadas pruebas de comportamiento, se centran en los requisitos funcionales del software, o sea, la prueba de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra no es una alternativa a las técnicas de prueba de caja blanca. Más bien se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores que los métodos de caja blanca (Pressman, 2001).

4.6.1.2 Diseño de casos de prueba

Las pruebas funcionales tienen entre sus objetivos probar la funcionalidad del código, intentar encontrar casos en los que la herramienta no atiende a sus especificaciones. A continuación se describen los casos de pruebas realizados.

Condiciones de ejecución:

Debe tener instalado el SIGB para corregir los ficheros del mismo con dicha aplicación.

Tabla 9: Caso de prueba #1 Prueba de funcionalidad para especificar idioma en los recursos enlazados

Escenario	Descripción	Idioma	Respuesta del sistema	Flujo central
1 Especificar idioma.	Mediante este escenario se muestra al desarrollador el resultado en un mensaje donde se le informa que fue corregido correctamente el idioma.	hreflang=" es-Es"	El sistema muestra un mensaje donde se informa que se ha corregido el idioma.	El desarrollador accede a las opciones que se brindan en menú se selecciona el fichero que se desea revisar, luego en opciones se escoge la opción "Corregir idioma en los enlaces", de forma automática se corrige este fichero, se muestra un mensaje informando que se ha corregido el idioma.

Tabla 10: Caso de prueba #2 Prueba de funcionalidad para buscar acrónimos sin descripción

Escenario	Descripción	Respuesta del sistema	Flujo central
2 Corregir acrónimos.	Mediante este escenario se le muestra al desarrollador un grupo de palabras que podrían ser identificadas como acrónimos.	El sistema muestra una ventana con los posibles acrónimos, luego de seleccionarlo, se activa un área en la ventana que permite poner la descripción del mismo.	El desarrollador accede a las opciones que se brindan en menú, se selecciona el fichero que se desea revisar, luego en opciones se selecciona "Buscar Acrónimos sin descripción", se muestra una ventana donde aparecen los posibles acrónimos y por apreciación del desarrollador selecciona los que son acrónimos, donde al seleccionar uno de ellos se activa un área de la ventana, donde permite agregarle la descripción al acrónimo, luego de agregarle las descripciones a todos los acrónimos, se guardan los cambios y sales de esa ventana a través del botón "Guardar y salir", la herramienta ejecuta los cambios sobre el fichero seleccionado.

4.6.2. Resultados de las pruebas

Los dos casos de prueba diseñados fueron aplicados en una iteración por parte del propio equipo de desarrollo, obteniendo un total de 2 no conformidades, siendo las dos significativas. Todas las no conformidades encontradas fueron corregidas y posteriormente se realizó una segunda iteración que no arrojó no conformidades.

4.7. Conclusiones parciales

Con la realización de este capítulo se concluye que:

- Se obtuvo a través del modelo de implementación, los diagramas de paquetes, componentes y despliegue, una mejor vista del desarrollo de la herramienta y el entorno en el que será utilizada.
- Se demostró el funcionamiento de los requisitos funcionales y se documentó los que no pudieron ser implementados.
- La aplicación de las pruebas realizadas al sistema, permitió detectar y corregir no conformidades, que pudieran atentar en contra del correcto funcionamiento del sistema final.

En este capítulo se abordaron los aspectos fundamentales de la implementación y prueba, donde los resultados obtenidos fueron satisfactorios.

Conclusiones generales

La presente investigación ha mostrado las diferentes herramientas existentes, que se utilizan para el análisis de la accesibilidad web, ayudando a la detección de los errores y a su corrección. Mostrando la necesidad de una herramienta que facilite la aplicación de pautas de accesibilidad web para débiles visuales, para facilitar y agilizar el proceso de corrección. En este marco y al finalizar el desarrollo de este trabajo de diploma, se concluye que:

El estudio de soluciones homólogas permitió definir las características del sistema desarrollado, y demostró la necesidad de implementar una herramienta para aplicar pautas de accesibilidad web para débiles visuales al catálogo en línea.

La valoración crítica del análisis y diseño realizado por el analista, permitió refinar las funcionalidades, para mejorar las características del sistema acorde a las expectativas del usuario.

El sistema implementado ayudará a agilizar el proceso de aplicación de pautas de accesibilidad web para débiles visuales en el catálogo en línea ya que su corrección de forma manual sería muy tediosa y extensa por la cantidad de elementos con la que cuenta el catálogo, entre ellos páginas HTML, archivos CSS, JavaScript, imágenes, los cuales hay que considerar para la aplicación de las pautas.

A pesar de no implementarse todos los Requisitos Funcionales, se implementó una herramienta que cumple con el objetivo general de desarrollar una herramienta informática que facilite la aplicación de las pautas de accesibilidad web en el catálogo en línea de la biblioteca de la UCI.

Recomendaciones

Debido al rápido crecimiento y perfeccionamiento en que se encuentra el mundo de la informática, para próxima versiones de la herramienta se recomienda:

Realizar un estudio en profundidad de algoritmos que permitan la aplicación de pautas de accesibilidad web para débiles visuales y enriquecer la aplicación con nuevas funcionalidades que permitan aplicar las mismas al catálogo en línea.

Referencias Bibliográficas

D, I T. 2005. Informática. *iti.es.* [En línea] 2005. [Citado el: 10 de abril de 2013.] ://www.iti.es/servicios/servicio/resource/7234/index.html..

ENRIQUE, JAVIER. 2010. www.buenastareas.com. *www.buenastareas.com.* [En línea] junio de 2010. [Citado el: 22 de octubre de 2012.] http://www.buenastareas.com/ensayos/Sistemas-Integrado-De-Gestion-Bibliotecaria/450218.html.

FERNANDEZ, IRUNE URIA. 2008. paginaspersonales.deusto.es. *Sistema de Busqueda de Información en Bibliotecas.* [En línea] 2008. [Citado el: 15 de octubre de 2012.] http://paginaspersonales.deusto.es/abaitua/konzeptu/htxt/grupoe.htm.

GUTIERRES, DAMIAN. 2007. codecompiling.net. *codecompiling.net.* [En línea] 2007. [Citado el: 25 de abril de 2013.] www.codecompiling.net/files/slides/UML_clase_05_UML_paquetes.pdf.

HERNÁNDEZ, KATIA. 2007. sparxsystems.com.ar. *El modelo de caso de uso.* [En línea] 2007. [Citado el: 2 de diciembre de 2012.] http://www.sparxsystems.com.ar/resources/tutorial/use_case_model.html.

HERNÁNDEZ, SUSANA CASTILLO. 2012. *Propuesta de accesibilidad en el catálogo en línea para discapacitados visuales.* Ciudad de La habana : s.n., 2012.

HERRERA, GUADALUPE GONZÁLEZ. 2005. hdl.handle.net. *hdl.handle.net.* [En línea] 2005. [Citado el: 15 de 1 de 2013.] http://hdl.handle.net/10760/14590.

HERRERA, GUADALUPE GONZÁLEZ. 2010. hdl.handle.net. *hdl.handle.net.* [En línea] 2010. [Citado el: 6 de 1 de 2013.] http://hdl.handle.net/10760/14590.

JACOBSON, I. 2000. *El proceso unificado de desarrollo de software.* Madrid : vol. 2012, 2000. ISBN 84-7829-036-2.

LAPUENTE, MARÍA JESÚS LAMARCA. 2005. hipertexto.info. *El nuevo concepto de documento en la cultura de la imagen.* [En línea] 2005. [Citado el: 28 de septiembre de 2012.] http://www.hipertexto.info/documentos/html.htm .

LARMAN, CRAIG. 1999. ecured.cu. *Patrones de diseño.* [En línea] 1999. [Citado el: 10 de enero de 2013.] http://www.ecured.cu/index.php/Patrones_de_dise%C3%B1o_y_arquitectura.

LEMUS, MARISOL SÁNCHEZ. 2010. *Propuesta de una guía de Accesibilidad Web para discapacitados en las herramientas Repositorio de Objetos de Aprendizaje y Herramienta de Autor Web de la Universidad de las Ciencias Informáticas.* Ciudad de La Habana : s.n., 2010.

MORA, SERGIO LUJÁN. 2006. accesibilidadweb.dlsi.ua.es. *accesibilidadweb.dlsi.ua.es.* [En línea] 2006. [Citado el: 20 de octubre de 2012.] accesibilidadweb.dlsi.ua.es/?menu=pautas-accesibilidad-contenido-web.

NAVARRETE, ÓSCAR ARRIOLA. 2008. *Sistemas integrales para la automatización de bibliotecas basados en software libre.* 2008.

NICOT, LESLIER LÓPEZ. 2008. *Propuesta de Guía para lograr la Accesibilidad Web en los Proyectos Productivos.* Ciudad de La Habana : s.n., 2008.

NOURIE, DANA. 2005. java.sun.com. *Getting Starte with an Integrated Development Environment(IDE).* [En línea] 24 de marzo de 2005. [Citado el: 2 de diciembre de 2012.] http://java.sun.com/developer/technicalArticles/tools/intro.html..

PRESSMAN, ROGER S. 1998. *Ingeniería del software: un enfoque práctico. .* 1998.

PRESSMAN. 2005. *Ingeniería de Software. Un enfoque práctico. .* 2005.

PRESSMAN. 2005. mastermagazine.info. *Definición de casos de uso.* [En línea] 2005. [Citado el: 3 de diciembre de 2012.]

PRESSMAN, ROGER S. 2001. *Ingeniería de software: un enfoque prático.* Madrid : s.n., 2001. ISO 9000.

QUESADA, CARLOS ALBERTO. 2013. slideshare.net. *Diagrama de clases.* [En línea] 2013. [Citado el: 12 de febrero de 2013.] http://www.slideshare.net/jpbthames/diagramas-de-clases.

REGLA. 2013. ie.inf.uc3m.es. *INGENIERÍA DEL SOFTWARE I y II.* [En línea] 2013. [Citado el: 19 de mayo de 2013.] http://www.ie.inf.uc3m.es/grupo/docencia/reglada/Is1y2/PracticaVP.pdf.

ROJAS, CARLOS ALBERTO. 2008. virtual.usalesiana.edu.bo. *Diagrama de Colaboración.* [En línea] 2008. [Citado el: 8 de enero de 2013.] http://virtual.usalesiana.edu.bo.

RUIZ, ENCARNACIÓN QUESADA. 2004. w3c.es. *w3c.es.* [En línea] 2004. [Citado el: 15 de octubre de 2012.] www.w3c.es/Traducciones/es/WAI/intro/accessibility#i-what.

VALDEZ, REINIER ANTONIO DÍAZ. 2012. repositorio_institucional.uci.cu. *Solución para la diseminación selectiva de información del Sistema Integrado de Gestión Bibliotecaria de la Universidad de las Ciencias Informáticas.* [En línea] 2012. [Citado el: 17 de mayo de 2013.] http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_05237_12.

INTECO. 2008. Comprobación de accesibilidad: Herramientas de evaluación de la accesibilidad web. 2008.

MARAÑÓN, GONZALO ÁLVAREZ. 1999. Características del lenguaje java. iec.csic.es. [En línea] 1999. [Citado el: 12 de octubre de 2012.] http://www.iec.csic.es/criptonomicon/java/quesjava.html.

Bibliografía consultada

ANDALUCIA, CONFEDERACIÓN DE EMPRESARIO DE. 2010. webaccesible.cea.es. *webaccesible.cea.es.* [En línea] 2010. [Citado el: 22 de sptiembre de 2012.] http://webaccesible.cea.es.

D, I T. 2005. Informática. *iti.es.* [En línea] 2005. [Citado el: 10 de abril de 2013.] ://www.iti.es/servicios/servicio/resource/7234/index.html..

ENRIQUE, JAVIER. 2010. www.buenastareas.com. *www.buenastareas.com.* [En línea] junio de 2010. [Citado el: 22 de octubre de 2012.] http://www.buenastareas.com/ensayos/Sistemas-Integrado-De-Gestion-Bibliotecaria/450218.html.

FERNANDEZ, IRUNE URIA. 2008. paginaspersonales.deusto.es. *Sistema de Busqueda de Información en Bibliotecas.* [En línea] 2008. [Citado el: 15 de octubre de 2012.] http://paginaspersonales.deusto.es/abaitua/konzeptu/htxt/grupoe.htm.

GUTIERRES, DAMIAN. 2007. codecompiling.net. *codecompiling.net.* [En línea] 2007. [Citado el: 25 de abril de 2013.] www.codecompiling.net/files/slides/UML_clase_05_UML_paquetes.pdf.

HERNÁNDEZ, KATIA. 2007. sparxsystems.com.ar. *El modelo de caso de uso.* [En línea] 2007. [Citado el: 2 de diciembre de 2012.] http://www.sparxsystems.com.ar/resources/tutorial/use_case_model.html.

HERNÁNDEZ, SUSANA CASTILLO. 2012. *Propuesta de accesibilidad en el catálogo en línea para discapacitados visuales.* Ciudad de La habana : s.n., 2012.

HERRERA, GUADALUPE GONZÁLEZ. 2005. hdl.handle.net. *hdl.handle.net.* [En línea] 2005. [Citado el: 15 de 1 de 2013.] http://hdl.handle.net/10760/14590.

HERRERA, GUADALUPE GONZÁLEZ. 2010. hdl.handle.net. *hdl.handle.net.* [En línea] 2010. [Citado el: 6 de 1 de 2013.] http://hdl.handle.net/10760/14590.

JACOBSON, I. 2000. *El proceso unificado de desarrollo de software.* Madrid : vol. 2012, 2000. ISBN 84-7829-036-2.

LAPUENTE, MARÍA JESÚS LAMARCA. 2005. hipertexto.info. *El nuevo concepto de documento en la cultura de la imagen.* [En línea] 2005. [Citado el: 28 de septiembre de 2012.] http://www.hipertexto.info/documentos/html.htm .

LARMAN, CRAIG. 1999. ecured.cu. *Patrones de diseño.* [En línea] 1999. [Citado el: 10 de enero de 2013.] http://www.ecured.cu/index.php/Patrones_de_dise%C3%B1o_y_arquitectura.

LEMUS, MARISOL SÁNCHEZ. 2010. *Propuesta de una guía de Accesibilidad Web para discapacitados en las herramientas Repositorio de Objetos de Aprendizaje y Herramienta de Autor Web de la Universidad de las Ciencias Informáticas.* Ciudad de La Habana : s.n., 2010.

MORA, SERGIO LUJÁN. 2006. accesibilidadweb.dlsi.ua.es. *accesibilidadweb.dlsi.ua.es.* [En línea] 2006. [Citado el: 20 de octubre de 2012.] accesibilidadweb.dlsi.ua.es/?menu=pautas-accesibilidad-contenido-web.

NAVARRETE, ÓSCAR ARRIOLA. 2008. *Sistemas integrales para la automatización de bibliotecas basados en software libre.* 2008.

NICOT LESLIER LÓPEZ. 2008. *Propuesta de Guía para lograr la Accesibilidad Web en los Proyectos Productivos.* Ciudad de La Habana : s.n., 2008.

NOURIE, DANA. 2005. java.sun.com. *Getting Starte with an Integrated Development Environment(IDE).* [En línea] 24 de marzo de 2005. [Citado el: 2 de diciembre de 2012.] http://java.sun.com/developer/technicalArticles/tools/intro.html..

PRESSMAN, ROGER S. 1998. *Ingeniería del software: un enfoque práctico. .* 1998.

PRESSMAN. 2005. *Ingeniería de Software. Un enfoque práctico. .* 2005.

PRESSMAN. 2005. mastermagazine.info. *Definición de casos de uso.* [En línea] 2005. [Citado el: 3 de diciembre de 2012.]

PRESSMAN, ROGER S. 2001. *Ingeniería de software: un enfoque prático.* Madrid : s.n., 2001. ISO 9000.

QUESADA, CARLOS ALBERTO. 2013. slideshare.net. *Diagrama de clases.* [En línea] 2013. [Citado el: 12 de febrero de 2013.] http://www.slideshare.net/jpbthames/diagramas-de-clases.

REGLA. 2013. ie.inf.uc3m.es. *INGENIERÍA DEL SOFTWARE I y II.* [En línea] 2013. [Citado el: 19 de mayo de 2013.] http://www.ie.inf.uc3m.es/grupo/docencia/reglada/Is1y2/PracticaVP.pdf.

ROJAS, CARLOS ALBERTO. 2008. virtual.usalesiana.edu.bo. *Diagrama de Colaboración.* [En línea] 2008. [Citado el: 8 de enero de 2013.] http://virtual.usalesiana.edu.bo.

RUIZ, ENCARNACIÓN QUESADA. 2004. w3c.es. *w3c.es.* [En línea] 2004. [Citado el: 15 de octubre de 2012.] www.w3c.es/Traducciones/es/WAI/intro/accessibility#i-what.

VALDEZ, REINIER ANTONIO DÍAZ. 2012. repositorio_institucional.uci.cu. *Solución para la diseminación selectiva de información del Sistema Integrado de Gestión Bibliotecaria de la Universidad de las Ciencias Informáticas.* [En línea] 2012. [Citado el: 17 de mayo de 2013.] http://repositorio_institucional.uci.cu/jspui/handle/ident/TD_05237_12.

INTECO. 2008. Comprobación de accesibilidad: Herramientas de evaluación de la accesibilidad web. 2008.

MARAÑÓN, GONZALO ÁLVAREZ. 1999. Características del lenguaje java. iec.csic.es. [En línea] 1999. [Citado el: 12 de octubre de 2012.] http://www.iec.csic.es/criptonomicon/java/quesjava.html.

W3C. 1994. Evaluación de accesibilidad de otros sitios web . w3.org. [En línea] 1994. [Citado el: 25 de octubre de 2012.] http://www.w3.org/WAI/eval/Overview.html.

W3C. 1994. Plan de implementación de Accesibilidad Web. w3.org. [En línea] 1994. [Citado el: 25 de octubre de 2012.] http://www.w3.org/WAI/impl/Overview.html.

Anexos

Anexo_1: Vistas de la herramienta

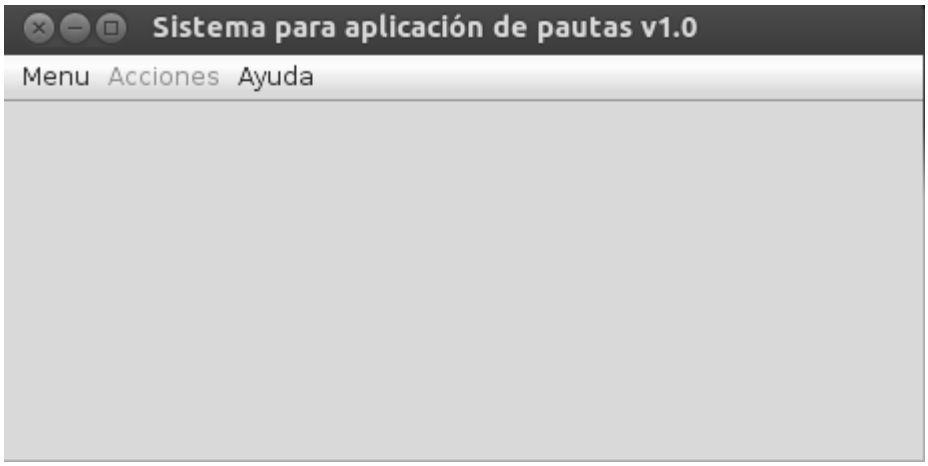

Ilustración 32 Vista de la interfaz principal

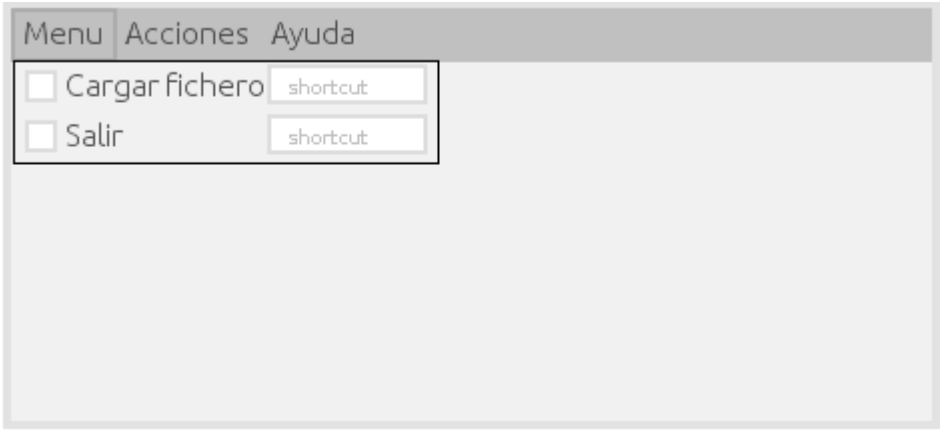

Ilustración 33 Vista de la interfaz principal con el menú desplegado

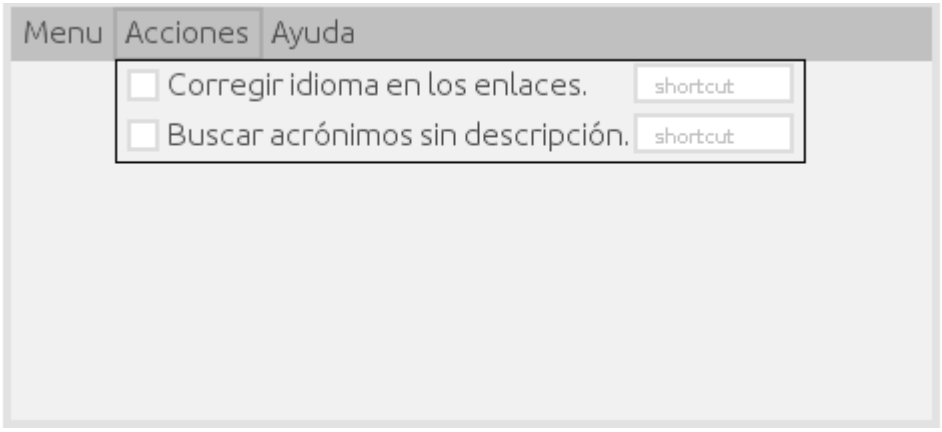

Ilustración 34 Vista de la interfaz principal con las opciones que brinda Acciones, desplegado

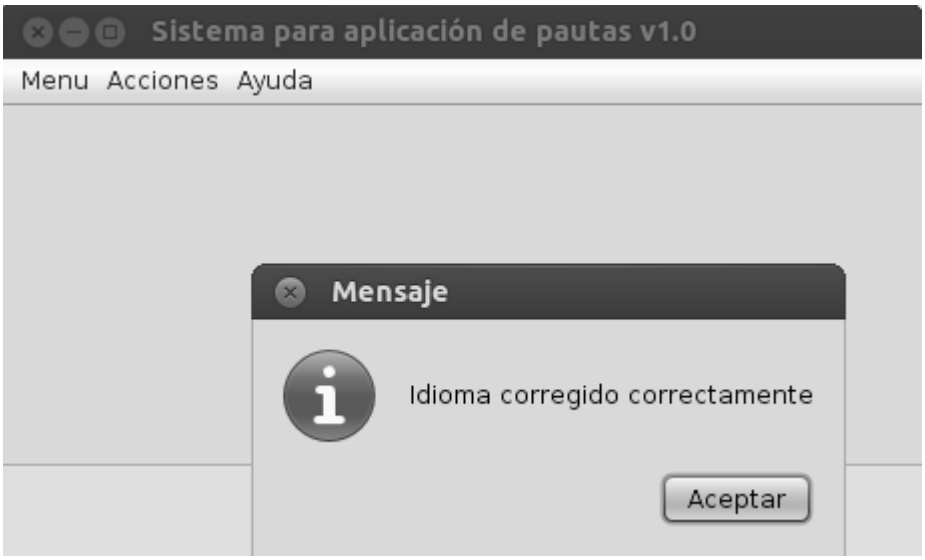

Ilustración 35 Vista del mensaje lanzado después de ser corregido el idioma en los enlaces

Ilustración 36 Vista de la interfaz, donde se mostrarán los acrónimos encontrados