

Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Centro de Enseñanza Asistida por Computador - CENEAC

Desarrollo de una Herramienta para Verificación de Criterios de Accesibilidad en sitios Web – HEVAC

Trabajo Especial de Grado
presentado ante la Ilustre
Universidad Central de Venezuela
por el Bachiller
Ronald de Jesús Aguilera González
C.I. 16.182.844
para optar al título de
Licenciado en Computación

Tutora:
Profa. Yusneyi Yasmira Carballo Barrera

Caracas, Junio de 2012

DEDICATORIA

Dedico este Trabajo de Especial de Grado a mi Madre, quien con todo su esfuerzo, apoyo y amor incondicional que siempre dio, ha dado y seguirá dando todo por mí y me ha enseñado a luchar para alcanzar mis metas. A Dios por estar presente en todo mi camino, aun en los momentos más difíciles. A mi tía Maritza González, quien me ha prestado todo su apoyo y sobre todo la oportunidad de estudiar en Caracas.

AGRADECIMIENTOS

El presente Trabajo Especial de Grado ha sido realizado gracias al apoyo y confianza recibida por parte de numerosas personas e instituciones, a quienes brindo todo mi agradecimiento por hacer posible la elaboración de este proyecto.

Mis agradecimientos van en primer lugar a Dios, por darme la fuerza, paciencia y constancia necesaria para llevar a cabo el proyecto.

A mi Madre, por brindarme todo su apoyo incondicional en cada uno de los momentos en que más me hizo falta.

A mi familia, por su preocupación y disposición a apoyarme en cualquier circunstancia, en especial a mis tíos Maritza González y José Gomes, por darme la oportunidad de estudiar en Caracas.

A mi tutora Profa. Yusneyi Carballo Barrera, que con su apoyo, paciencia y conocimientos supo orientarme en la realización de este proyecto, logrando de forma conjunta la elaboración de un trabajo de investigación que beneficie la accesibilidad en la Web a los ciudadanos con discapacidad visual y desarrolladores de sitios Web.

A la empresa FYC Soluciones Integrales, por darme la oportunidad de crecer profesionalmente en los 5 años que estuve en la empresa.

A mis amigos Luis Arturo Aulestia y David Martínez, por darme ese último empuje final para concluir este trabajo.

A mis antiguos supervisores, Manuel Cánchica, Leslie García y Carolina Elortegui, por brindarme su apoyo y ayuda en el aprendizaje y fortalecimiento del área de desarrollo e implementación de Middlewares y Sitios Web, por su flexibilidad y paciencia en los momentos en que necesité de tiempo para la culminación de este proyecto.

Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Centro de Enseñanza Asistida por Computador - CENEAC

TRABAJO ESPECIAL DE GRADO

Título: Desarrollo de una Herramienta para Verificación de Criterios de Accesibilidad en sitios Web - HEVAC

Autor: Ronald de Jesús Aguilera González

Unidad de Investigación: Tecnología Educativa.
Palabras Claves: Accesibilidad, W3C, HTML, Discapacidad Visual.

Tutora: Yusneyi Carballo Barrera.
Fecha de Presentación: 25 de Junio de 2012.

Resumen

En este Trabajo Especial de Grado se presenta el desarrollo de una Herramienta para comprobar los problemas de accesibilidad que pueden presentar los sitios Web y que afectan la navegación de los usuarios. La misma está basada en la verificación de recomendaciones y pautas para lograr la accesibilidad en la Web, las cuales están fijadas por el Consorcio internacional del *World Wide Web* (W3C). El desarrollo se basa en un ambiente cliente-servidor, usando la tecnología *Java* para comunicación vía Web basada en *Servlets*. La respuesta de la Herramienta es generada por las evaluaciones de incumplimientos de las pautas del W3C que presentan las páginas a verificar. El desarrollo de la Herramienta se apoya en una metodología de desarrollo *ad-hoc* utilizando en principio la Programación Extrema (XP) y los artefactos de notación UML en las etapas de análisis y diseño. Con el desarrollo de la Herramienta, se obtiene un listado de problemas relacionados con la accesibilidad de la página Web que es evaluada y que pueden afectar a las personas con discapacidad, especialmente la discapacidad visual, y que deben ser solventadas por el desarrollador o diseñador del sitio Web.

Palabras Claves: Discapacidad Visual, Accesibilidad en la Web, W3C, WCAG, Programación Extrema, HTML, PDF, *Java*, *PostgreSQL*, *Servlets*, JSP.

ÍNDICE

IMÁGENES

TABLAS

INTRODUCCIÓN

La accesibilidad en las Tecnologías de Comunicación e Información (TIC) es uno de los aspectos ligados a garantizar la igualdad de oportunidades para las personas con discapacidad y es un tema que debe llamarnos a reflexión. Los avances tecnológicos plantean cada día nuevos retos que hacen necesario el crear conciencia sobre las realidades y necesidades de personas con características distintas y con limitaciones o discapacidades que pueden obstaculizar el uso de esos adelantos tecnológicos.

En la llamada Era de la Información, la población se hace cada día más dependiente del acceso a las comunicaciones y a los dispositivos que lo permiten. Muchas actividades económicas tienen un lugar en la red mundial de telecomunicaciones, por lo que es necesario considerar y velar el aspecto de la accesibilidad para aquellas personas en situación de exclusión a fin de que sus posibilidades en las sociedades informatizadas no se vean mermadas, y a fin de minimizar las manifestaciones de la denominada brecha digital.

En todas las sociedades del mundo existen aún obstáculos que impiden que las personas con discapacidad ejerzan sus derechos y libertades, además que dificultan su plena participación en las actividades de sus respectivas sociedades, siendo primeramente responsabilidad del Estado adoptar las medidas necesarias para minimizar, sino eliminar, estos obstáculos. En forma más concreta, también es una responsabilidad de quienes tenemos participación en el desarrollo de tecnologías y aplicaciones orientadas al uso de las denominadas Tecnologías de la Información y la Comunicación, el crear productos y servicios que garanticen la accesibilidad y la inclusión.

El interés del presente trabajo, tomando como base la investigación sobre las pautas de accesibilidad en la Web citados por el W3C, es desarrollar una herramienta, fundamentada en estas pautas, que permita realizar la verificación del cumplimiento de accesibilidad las sobre las páginas Web que el usuario desee verificar.

A manera de orientar al lector en el seguimiento y comprensión de este documento, el presente trabajo se encuentra estructurado como se describe a continuación:

- **Capítulo I:** Describe la deficiencia de acceso a las TIC que padecen los ciudadanos con discapacidad, principalmente con la discapacidad visual, se establecen los objetivos de este trabajo y se determina su alcance.
- **Capítulo II:** Se presentan los fundamentos conceptuales de la discapacidad visual y las pautas de accesibilidad en la Web citados por el W3C.

- **Capítulo III:** Se destaca metodología utilizada para el desarrollo del trabajo, se describe los procesos de implementación y de pruebas llevadas a cabo con el uso de la herramienta.

Este documento finaliza presentando los resultados y conclusiones del Trabajo Especial de Grado, recomendaciones que permitirán la evolución del prototipo HEVAC (Herramienta de Verificación de Accesibilidad), las referencias bibliográficas de las fuentes visitadas y los anexos que contienen una información complementaria del presente trabajo.

CAPÍTULO I: EL PROBLEMA

En el presente capítulo se presentan los argumentos que motivan este Trabajo Especial de Grado, la problemática actual, la justificación e importancia de llevar a cabo la implementación de este trabajo, los objetivos y alcances establecidos y los resultados que se esperan lograr al finalizar las pruebas integrales de la herramienta.

1. Planteamiento del Problema

En la actualidad el entorno social, económico y cultural que envuelve al ser humano evoluciona de una manera estrepitosa. La revolución de la tecnología informática, ha provocado una transformación radical de las formas de producción, difusión y consumo del conocimiento y la cultura. La aparición de nuevas tecnologías tales como: La televisión digital y por suscripción, el uso de los computadores en los hogares, el acceso a Internet y la telefonía celular están provocando nuevas necesidades formativas y de conocimiento en los ciudadanos. Por estas y otras razones, es claro, que las personas de hoy requieren de nuevas habilidades y conocimientos para poder desarrollarse en el mundo contemporáneo.

Sin embargo, la adopción de la tecnología también exige un proceso de selección y de decisión en cuanto a la tecnología apropiada, quién la va a implementar, para quién va dirigida, qué tipo de contenido se suministrará y para qué tipo de comunicación. Este proceso es útil, dado que ayuda a concretar ideas, a crear visiones y a motivar a un mayor número de habitantes a alcanzar aptitudes de lectura y escritura a través de la tecnología.

Por otra parte, existe un gran número de habitantes que poseen discapacidades tanto físicas como psicomotoras, los cuales requieren adquirir estas aptitudes de lectura-escritura. Actualmente, una de cada diez personas posee una discapacidad, entre cuarenta y cincuenta millones son personas con discapacidad visual, de los cuales más de cinco millones residen en Latinoamérica evidenciando precarios niveles formación y capacitación para el trabajo (OMS, 2009).

Es esencial que las TIC sean accesibles con el fin de ofrecer un acceso equitativo e igualdad de oportunidades para las personas con diversas capacidades. En efecto, la Convención de las Naciones Unidas sobre los Derechos de las Personas con Discapacidad reconoce el acceso a las TIC, incluyendo la Web, como un derecho humano básico (Naciones Unidas, 2006).

La Web está diseñada fundamentalmente para el trabajo de todas las personas, independientemente del uso de hardware, uso de software, idioma, localización, cultura o capacidad física o mental. Cuando la Web cumple con este objetivo, es accesible a personas con un

amplio rango de sensibilidad auditiva, capacidad de movimiento, capacidad visual y la capacidad cognitiva.

La Web elimina las barreras de comunicación e interacción que enfrentan muchas personas en el mundo físico. Sin embargo, cuando los sitios, tecnologías o herramientas Web están mal diseñados, pueden crear barreras que excluyen a las personas al uso de la Web.

Considerando, que las personas que poseen alguna discapacidad pueden tener problemas al usar las tecnologías de la Web, se observa como problema principal la necesidad de una herramienta que realice la verificación del cumplimiento de accesibilidad Web enfocado en las necesidades de personas con discapacidad visual basado en las Pautas de Accesibilidad en Contenido Web versión 2.0 establecidos por el W3C.

Esta herramienta considera un conjunto de funcionalidades de otros aplicativos en la Web, tales como Test de Accesibilidad Web (TAW) (CTIC, 2009) y el *Web Accesibility Evaluation Tool* (WAVE) (WebAIM, 2001) para ser implementadas en la herramienta. Las funcionalidades son las siguientes:

- Lectura del URL de un sitio Web que referencie a una página.
- Evaluación del código interno de la página con el fin de comprobar las pautas de accesibilidad Web citadas por el W3C (W3C-WCAG, 2008).
- Reporte de problemas hallados con el código de la fuente verificada y resaltada donde se encontró el error.

Por otro lado, se incorporan nuevas funcionalidades, las cuales representan parte del aporte de trabajo:

- Despliegue del reporte de problemas simplificado, adaptando los datos mostrados para personas con discapacidad.
- Registro histórico de seguimiento en las verificaciones realizadas a páginas Web previamente comprobadas.
- Consejos técnicos de desarrollo mostrados en el detalle de los problemas encontrados en la verificación.
- Despliegue del reporte de problemas en formato de impresión, tales como documentos PDF.

A partir de estas funcionalidades se toman las bases necesarias para desarrollar la herramienta propuesta para este trabajo.

2. Justificación e Importancia

La sociedad actual ha sido calificada como *sociedad de la información y el conocimiento*, apoyada, de forma preponderante, en las denominadas Tecnologías de la Información y la Comunicación (TIC). En este contexto, el desarrollo personal y social de las personas va a estar determinado, en gran medida, por su cualificación en el manejo de estas tecnologías.

Sin embargo, no todos los dispositivos tecnológicos comunes en el mercado son susceptibles de ser utilizados fácilmente por cualquier individuo. En muchos casos, se requiere su adaptación para evitar que sean un factor más de desventaja o segregación para algunas personas. De forma especial, y dado el alto contenido visual de las Tecnologías de Información y Comunicación, las personas con ceguera o discapacidad visual se ven limitadas en su uso.

Estas nuevas metodologías deben ser adaptadas, mejoradas o rediseñadas para reformar su accesibilidad en tareas que deben ser consideradas desde el momento de su concepción, pasando por su diseño y producción. Así, teniendo en cuenta las necesidades y la diversidad del conjunto de sus usuarios, se favorecerá su acceso en igualdad de condiciones, con independencia de su condición cultural, social, de salud o de capacidades.

Por otro lado, la aplicación de la tecnología, en general, ha supuesto y supone una fuente constante de soluciones para las personas con ceguera y deficiencia visual en los diferentes ámbitos de su autonomía y bienestar: vida diaria, movilidad, educación, trabajo, ocio, cultura, etc.

3. Objetivos de la Investigación

A continuación se indica el objetivo general y los objetivos específicos del presente Trabajo.

3.1. Objetivo General

Desarrollar una Herramienta para la Verificación de las Pautas de Accesibilidad en Contenidos Web establecidas por el W3C en la guía WCAG versión 2.0 (W3C-WCAG, 2008).

3.2. Objetivos Específicos

En consecuencia se plantean los siguientes objetivos específicos:

1. Revisión de las Pautas de Accesibilidad de Contenido Web (*Web Content Accesibility Guidelines*, WCAG) planteados por el W3C en el documento WCAG (W3C-WCAG, 2008), especialmente, las pautas que apoyan la accesibilidad de sitios Web para personas con discapacidad visual.

2. Seleccionar las pautas a ser incorporadas en el componente verificador de la herramienta.
3. Aplicar la metodología *ad-hoc* de desarrollo de software seleccionada para la creación del componente de verificación de pautas de accesibilidad, el cual incorporará Java, *Servlets* y JSP entre otras tecnologías.
4. Desarrollar una herramienta Web (HEVAC) que permita el uso del componente de verificación, el despliegue de resultados y otras funcionalidades de la aplicación propuesta como solución, utilizando Java, HTML, CSS y PostgreSQL, entre otras tecnologías.
5. Realizar pruebas de funcionalidad, usabilidad y accesibilidad.
6. Probar las funcionalidades de verificación de la herramienta mediante la evaluación de accesibilidad de un grupo de páginas Web.
7. Documentar la investigación.

4. Límites y Alcances

El alcance de la herramienta desarrollada es:

- La herramienta está orientada principalmente a los siguientes tipos de usuarios:
  - Desarrolladores, Diseñadores y Administradores de sitios Web.
  - Facilitadores de personas con discapacidad.
  - Personas con Discapacidad Visual Moderada.
  - Personas con limitaciones temporales.
- La herramienta Web está desarrollada bajo el lenguaje Java, con la especificación de la máquina virtual JRE versión 6 usando las especificaciones *Servlets* y JSP. A su vez, se utiliza el servidor Web y contenedor de *Servlets*, Apache Tomcat, en su versión 7.0.
- La herramienta registra un historial de análisis ejecutados bajo un mismo sitio Web usando el servidor de base de datos PostgreSQL con la intención de realizar un seguimiento de los cambios de accesibilidad del sitio Web en cuestión.
- La herramienta verifica los problemas y advertencias de accesibilidad Web basados en algunas pautas citadas en la especificación WCAG versión 2.0.
- Las verificaciones se realizarán en el código HTML de la página consultada, no incluirá los códigos de las hojas de estilos en cascada (CSS) debido a que existen pocas técnicas citadas en las pautas de accesibilidad relacionadas con la tecnología CSS y de igual manera no se incluirán elementos de *JavaScripts* ya que se necesita interpretar el código del *script* para verificar si está modificando la página HTML, la hoja de estilo en cascada o el mismo *Script*. No soportará la verificación de accesibilidad en páginas HTML que posean marcos o

frames ya que pueden existir marcos que incluyan a páginas externas ajenas al sitio Web que atenten contra la accesibilidad.

5. Resultados que se esperan obtener

La herramienta Web deberá mostrar los resultados de las verificaciones en un formato adaptado a los usuarios con discapacidad visual. De estos resultados se esperan las siguientes características:

- Los resultados que arrojará la herramienta de verificación de accesibilidad, pueden ejecutarse usando los criterios de Nivel de Conformidad o Principios de Accesibilidad.
- La primera vista de los resultados que se ofrecerá principalmente, muestra un recuadro básico indicando la cantidad de problemas y advertencias que se encontraron al realizar la verificación de accesibilidad. La vista detallada estará disponible como un enlace alternativo. Esto se realiza con la intención de mostrar inicialmente la vista básica que posee adaptaciones para usuarios con baja visión, mientras que los detalles de las verificaciones son de mayor interés para diseñadores y desarrolladores.
- En los resultados se podrá mostrar el código donde se encontró un problema o una advertencia, además de mostrar un consejo de desarrollo o recomendación.
- Los resultados que arrojará la herramienta de verificación de accesibilidad puede ser exportado en formato PDF y esto incluirá tanto la vista básica como la vista detallada.

CAPÍTULO II: MARCO TEÓRICO

El presente capítulo está constituido por las referencias teóricas de los temas principales en los que se fundamenta este Trabajo, los cuales son la Discapacidad Visual y la Accesibilidad en la Web, realizando sobre este último un énfasis adicional sobre las normativas existentes por las que se acoplan las distintas tecnologías para realizar el cumplimiento de la accesibilidad.

1. La Discapacidad Visual

El sistema de Clasificación Internacional de Deficiencias, Discapacidades y Minusvalías (CIDDM) de la Organización Mundial de la Salud (OMS) se utiliza para clasificar los trastornos (enfermedades), deterioros, discapacidades y minusvalías. Las definiciones son las siguientes (CIF, 2001):

- Una **enfermedad** es una condición médica, independientemente de su origen o fuente, que representa o puede representar un daño significativo a los seres humanos.
- Un **deterioro** es toda pérdida o anormalidad de una estructura anatómica o una función fisiológica o psicológica.
- Una **discapacidad** es toda restricción o ausencia (debido a un deterioro) de la capacidad para realizar una actividad dentro del rango considerado normal para un ser humano.
- Una **minusvalía** indica la posición de desventaja de una persona en la sociedad, como resultado de un deterioro y/o una discapacidad.

El término "deterioro visual" se refiere a una limitación funcional de uno o ambos ojos o del sistema visual debido a un trastorno o enfermedad que puede resultar en una discapacidad visual o una minusvalía del sistema visual. Por ejemplo, la degeneración macular (una enfermedad) puede dar lugar a la agudeza visual reducida (una alteración en la visión). La discapacidad visual es una limitación de la (s) capacidad (es) de la persona (por ejemplo, la incapacidad para leer letra pequeña), y una minusvalía del sistema visual se refiere a una limitación de la autonomía personal y socioeconómica. En pocas palabras, la discapacidad visual puede ser considerada como la visión insuficiente para las necesidades de un individuo.

La discapacidad visual se clasifica en (Colenbrander, 1999):

- Moderada: Pueden realizar tareas visuales usando ayudas especiales e iluminación adecuada casi como cualquier persona sin problemas de visión.
- Severa: Requieren más tiempo y más esfuerzo para realizar tareas visuales, se debe poner más voluntad y ser preciso aun empleando ayudas ópticas.

- Profunda: Puede ser muy difícil realizar tareas visuales y no pueden hacer nada que exija visión fina o de detalle.
- Ceguera: Ausencia total de la visión.

Según la clasificación de Pierre Henry, la discapacidad visual puede catalogarse en (González García, 1990):

- Deficientes visuales absolutos o de nacimiento.
- Deficientes visuales de nacimiento operados.
- Deficientes visuales con percepciones luminosas débiles.
- Deficientes visuales que han pasado de videntes a ciegos.
- Deficientes visuales que han pasado de videntes a semiciegos.
- Deficientes visuales que han pasado de semividentes a ciegos.

La clasificación de la discapacidad visual varía en todo el mundo. En (OMS, 2009) se clasifican los niveles de discapacidad visual basado en la agudeza visual y/o limitación del campo visual, y define la ceguera como un deterioro profundo (esto puede hacer referencia a la ceguera de un ojo o ceguera de la persona). La definición de ceguera de la OMS especifica la agudeza visual inferior a 20/400 y/o permanecer en el campo visual de menos de 10 grados en el ojo de mejor visión.

La agudeza visual es la capacidad del sistema de visión para percibir, detectar o identificar objetos espaciales con unas condiciones buenas de iluminación. Para una distancia al objeto constante, si el paciente ve nítidamente una letra pequeña, tiene más agudeza visual que uno que no la ve.

Para esta investigación se tomará la clasificación establecida por la OMS, la cual toma a la agudeza visual como criterio de clasificación y se define en la Tabla 1.1:

Categoría de Impedimento Visual	Agudeza Visual	Condición Visual
0	20/20 a 20/60	Normal o aceptable
1	20/60 a 20/200	Deterioro visual (baja visión)
2	20/200 a 20/400	Deterioro visual severo
3	20/400 a 5/300, campo visual de 10° a 5°	Ceguera
4	5/300 a percepción de luz, campo visual < 5°	
5	Sin percepción de luz, ceguera total	

Tabla 1.1: Categorías de Discapacidades Visuales.
Fuente: Organización Mundial de la Salud, 2009.

En Optometría, para calcular la agudeza visual de un paciente, lo que se hace es someterlo a distintas pruebas visuales, tales como:

- **Prueba de Snellen:** son las más populares. Están formados por filas de letras que van de tamaño más grande a más pequeño conforme bajamos la mirada. Cuanto más abajo logre ver nítido el paciente, mayor agudeza visual tendrá (ver Imagen 1.1).
- **Prueba de Landolt:** formados por filas de caracteres que son circulares pero con un trazado no continuo, sino con una discontinuidad que el paciente tendrá que identificar (anillos de Landolt). El sistema de cálculo de la agudeza visual es el mismo que el anterior: arriba los caracteres más grandes y abajo los más pequeños. Por consecuencia, es normal en estas pruebas la presencia de la letra C en forma circular y en distintos tamaños y posiciones. El paciente en este caso tendrá que resolver dónde está la discontinuidad: arriba, abajo, derecha, etc**.**
- **Prueba de contraste y frecuencias:** su objetivo es el cálculo de la agudeza visual al contraste (diferenciación de blanco y negro) que somos capaces de distinguir.

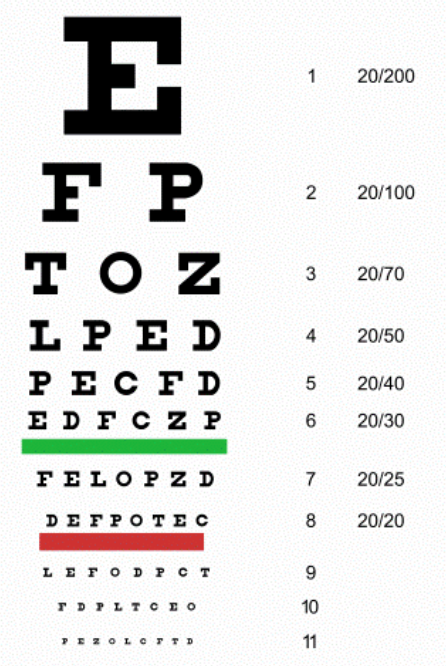


Imagen 1.1: Gráfico de Snellen.
Fuente: Wikipedia.org

El campo visual es lo que permite al ojo humano enfocar varias cosas a la vez, es todo lo que abarca la vista al mirar de frente hacia un punto fijo y se mide en grados de ángulo.

1.1. Prevención y tratamiento

Existen diversas enfermedades y problemas que pueden afectar a los ojos y por lo tanto hacer perder la visión. Entre éstas se encuentran las anormalidades, los defectos genéticos, envenenamientos, la falta de iris al nacer, las enfermedades y la malnutrición, siendo éstas dos últimas las principales causas (OMS, 2009).

La catarata, responsable del 50% de los casos de ceguera a nivel mundial, sigue siendo la causa principal de discapacidad visual en todas las regiones del mundo. Las enfermedades crónicas no transmisibles como el **glaucoma** y la **retinopatía diabética** causan el 12% y el 5% de ceguera mundial respectivamente. El **tracoma** y la **oncocercosis** son las principales causas infecciosas de ceguera evitable. La **ceguera infantil** requiere mayor atención ya que hasta la mitad de todos los casos son prevenibles o tratables mediante intervenciones conocidas (OMS, 2009).

De manera general se describe a continuación las enfermedades antes mencionadas, así como su tratamiento y prevención (ONCE, 2007).

- La **catarata**: es una enfermedad progresiva relacionada con el envejecimiento, que impide que la luz llegue a la retina debido a una opacidad del cristalino causado por el exceso de acumulación de proteína o al cambio de coloración del cristalino (amarillento o marrón) a causa de la edad (ver Imagen 1.2).

  A pesar de que no existen medidas preventivas para evitar las cataratas existen factores de riesgo que propician su desarrollo: enfermedades como la diabetes, el consumo de alcohol y tabaco y la exposición prolongada a los rayos del sol.

  No existe ningún medicamento para el tratamiento para las cataratas, se puede recurrir al uso de anteojos graduados y de protección solar durante su etapa inicial. Posteriormente el único tratamiento eficaz es la cirugía que consiste en remplazar el cristalino por un lente.

Visión Normal Visión con Catarata

Imagen 1.2: Comparación de la visión con Catarata.
Fuente: ONCE, España.

- El **glaucoma**: consiste en la pérdida del campo visual a causa de un aumento de presión interna en el ojo dañando el nervio óptico. Ésta enfermedad es silenciosa ya que aparentemente no tiene síntomas, es detectado por medio de exámenes oftalmológicos. Los enfermos de glaucoma comienzan por perder la visión lateral y después, si no es

tratada la visión frontal, la pierden por completo. La incidencia más alta se encuentra en personas afroamericanas mayores de 40 años, descendientes latinos mayores de 60 años y personas con familiares que han tenido glaucoma (ver Imagen 1.3).

La visión que se pierde por el glaucoma no se puede recuperar pero si se detecta a tiempo los efectos que produce pueden ser controlados con tratamiento que incluye medicamentos, cirugía láser, cirugía convencional o una combinación de ellos.

Visión Normal Visión con Glaucoma

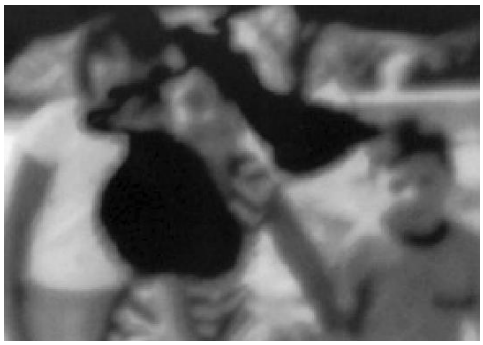
Imagen 1.3: Comparación de la visión con Glaucoma.
Fuente: ONCE, España

- La **retinopatía diabética**: es una complicación ocular de la diabetes que está causada por el deterioro de los vasos sanguíneos que irrigan la retina. El daño de los vasos sanguíneos de la retina puede tener como resultado que estos sufran una fuga de fluido o sangre. Si la enfermedad avanza se forman nuevos vasos sanguíneos y prolifera el tejido fibroso en la retina, lo que tiene como consecuencia que la visión se deteriore, pues la imagen enviada al cerebro se hace borrosa (ver Imagen 1.4).

  Es un daño progresivo e irreversible a los vasos sanguíneos de la retina del ojo y se puede presentar de dos maneras: la no proliferativa, donde los vasos retinianos se vuelven porosos y dejan filtrar el líquido hacia la retina ocasionando una visión borrosa y la proliferativa, donde hay un crecimiento de vasos sanguíneos dentro del ojo los cuales son frágiles y pueden sangrar, ocasionando hemorragias retinianas y edemas retinianos llevando a la pérdida progresiva de la visión.

  Para su oportuna detección se recomienda a los diabéticos hacerse un examen oftalmológico una vez al año. Una vez diagnosticada la retinopatía diabética se debe tratar, según sea el caso, con: cirugía láser para la retinopatía no proliferativa que consiste en sellar o cauterizar los vasos retinianos porosos para detener la filtración del líquido a la retina; la vitrectomía es una intervención quirúrgica aplicable a la retinopatía proliferativa que consiste en remover la sangre y las cicatrices para después aplicar láser sobre la retina.

Visión Normal



Visión con Retinopatía Diabética

Imagen 1.4: Comparación de la visión con Retinopatía Diabética.
Fuente: ONCE, España

- El **tracoma**: es una infección ocular causada por la bacteria *chlamydia trachomatis* y se relaciona con los sectores socioeconómicos bajos debido a la falta de higiene (ver Imagen 1.4).

  Los síntomas se presentan como una conjuntivitis y si no se recibe tratamiento puede conducir a cicatrización. Los párpados se irritan de manera severa, las pestañas pueden invertirse y rozar la córnea produciendo ulceraciones oculares, cicatrización posterior, pérdida visual y ceguera. El tratamiento para curar esta infección es mediante antibióticos y en casos más severos con una cirugía de párpado para regresarlo a su posición normal.

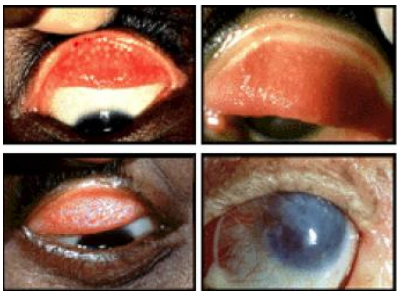


Imagen 1.5: Tracoma.
Fuente: ONCE, España

- La **oncocercosis** o ceguera del río: es una enfermedad provocada por un parásito llamado *onchocerca volvulus* que produce alteraciones en los ojos hasta llegar a la ceguera. Esta enfermedad se transmite cuando una mosca negra del género *simulium* pica a una persona infectada e ingiere las microfilarias (gusanitos pequeños) que están debajo de la piel; luego de seis a doce días dentro de la mosca las microfilarias se convierten en larvas. La mosca pica a otra persona y deposita las larvas infectantes. Las larvas se desarrollan dentro de la persona hasta volverse adultos. Machos y hembras se unen dentro de un nódulo. Las

hembras dentro de los nódulos producen millares de microfilarias. Las microfilarias migran debajo de la piel y pueden llegar a los ojos y producir ceguera.

El tratamiento para la oncocercosis es mediante medicamentos controlados debido a sus efectos secundarios. Para no contraerla se recomienda, además de destruir los criaderos de larvas, aplicar larvicidas, insecticidas y repelentes así como utilizar ropa adecuada para evitar picaduras.

- La **ausencia de vitamina A**: es la principal causa de ceguera infantil en el mundo. Por ser ésta un componente de los pigmentos visuales encargados de una adecuada visión, una deficiencia importante de esta vitamina puede ocasionar desde ceguera nocturna hasta la pérdida de la visión. Su prevención es a base de cápsulas y alimentos ricos en este nutriente.

Las tendencias mundiales muestran desde principios de los años noventa una disminución de las tasas internacionales de discapacidad visual y variaciones en la distribución de sus causas. La discapacidad visual y la ceguera debidas a enfermedades infecciosas han disminuido mucho, lo cual refleja el éxito de las medidas internacionales de salud pública, pero hay un aumento notorio del número de personas con discapacidad visual o ceguera debido a trastornos relacionados con el aumento de la esperanza de vida (NEI, 2008).

La presbicia (incapacidad de leer o realizar trabajos cercanos que se produce con el envejecimiento) causa disfunción visual si no se corrige. No se conoce exactamente la magnitud de este problema, pero estudios preliminares indican que podría ser muy frecuente, sobre todo en los países en desarrollo (NEI, 2008).

1.2. Datos y cifras referentes a la Discapacidad Visual

A continuación se presentan los datos y cifras a nivel mundial concernientes a la discapacidad visual (OMS, 2009):

- En el mundo hay aproximadamente 314 millones de personas con discapacidad visual, 45 millones de las cuales son ciegas.
- La mayoría de las personas con discapacidad visual tienen edad avanzada, y el riesgo es mayor para las mujeres a todas las edades y en todo el mundo.
- Aproximadamente un 87% de las personas con discapacidad visual en el mundo viven en países en desarrollo.
- El número de personas ciegas debido a enfermedades infecciosas ha disminuido mucho, pero el deterioro visual relacionado con la edad va en aumento.

- Las cataratas siguen siendo la principal causa de ceguera en todo el mundo, excepto en los países más desarrollados.
- La corrección de los errores de refracción podría devolver una visión normal a más de 12 millones de niños de 5 a 15 años.
- Aproximadamente un 85% de los casos mundiales de discapacidad visual son evitables.

1.3. Grupos de riesgo

La clasificación mundial de los grupos de riesgo relacionados con la discapacidad visual se divide de la siguiente manera (OMS, 2009):

- **Por edad:** Aproximadamente un 82% de las personas con discapacidad visual son mayores de 50 años, pese a que sólo representan un 19% de la población mundial.

  A medida que aumenta la población mundial y la proporción de ancianos aumenta también el número de personas en riesgo de sufrir discapacidad visual relacionada con la edad, incluso en los países en desarrollo.

  La ceguera infantil sigue siendo un problema importante en todo el mundo. Se calcula que 1,4 millones de menores de 15 años son ciegos durante muchos años. Por otra parte, más de 12 millones de niños de 5 a 15 años sufren discapacidad visual por errores de refracción (miopía, hipermetropía o astigmatismo) no corregidos, trastornos que se pueden diagnosticar y corregir fácilmente con el uso de lentes.

- **Por sexo:** Los estudios realizados revelan sistemáticamente que las mujeres corren mayor riesgo que los hombres de sufrir discapacidad visual, cualquiera que sea la región del mundo y la edad**.**

- **Por localización geográfica:** La discapacidad visual no está distribuida uniformemente por el mundo. Aproximadamente un 85% de las personas con discapacidad visual viven en países en desarrollo**.**

1.4. Integración social

Que una persona padezca de ceguera no implica obligatoriamente incapacidad. El ajuste individual va a depender de la edad de comienzo, el carácter de la persona, su educación y recursos socioeconómicos (ONCE, 2007).

En las personas jóvenes el impacto es en muchos casos una tragedia mientras que las personas mayores lo aceptan con mayor facilidad. Lo mismo sucede con las personas ciegas de

nacimiento o con las que pierden la visión gradualmente, donde la aceptación es más fácil que la de una persona que ha perdido la vista en un accidente.

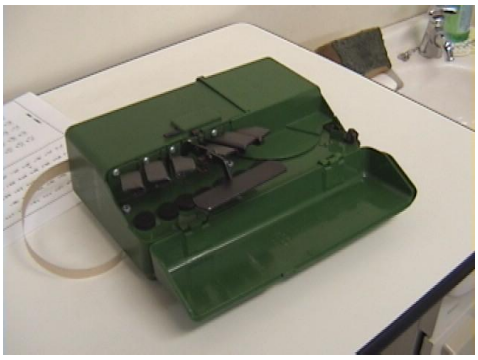
La educación es un factor muy importante, que se debe tomar en cuenta. Por ejemplo, la relación que hay entre la escuela y la familia. El nacimiento de un niño ciego puede provocar un desajuste familiar muy intenso; normalmente muchas familias optan en un primer momento por intentar solucionar el problema médicamente; una vez descartadas todas las posibilidades terapéuticas recurren a escuelas especializadas en estos temas, las cuales suelen dar respuestas a las necesidades educativas que plantean los alumnos y familiares de éste. La escuela interviene el desarrollo socioeducativo del niño y, dentro de sus posibilidades, sobre la familia y el entorno social en el que vive.

1.5. Recursos para personas con Discapacidad Visual

Los recursos más utilizados para las personas con discapacidad visual son los siguientes (ONCE, 2007):

- **Perro guía:** el perro guía supone para la persona ciega, con quien convive y para quien trabaja, más que sus ojos. Él es portador de oportunidades, de independencia, de libertad, de colaboración sin condiciones, de abnegación, de amor y comprensión. Por todo ello, el perro guía debe recibir amor, cuidados y respetos suficientes (ver Imagen 1.6).
- **Máquina Perkins:** La máquina de escribir Perkins es una modificación de la máquina de escribir normal, que crea caracteres Braille en el papel en lugar de imprimir caracteres visuales. Consta de 6 teclas que corresponden a los 6 puntos que se usan en el sistema Braille (ver Imagen 1.6).
- **El bastón:** El bastón es un instrumento con el que la mayoría de los ciegos consiguen hoy en día movilidad e independencia. Con él se facilita el desplazamiento por el entorno evitando los obstáculos e intentando orientarse con la ayuda del oído. Este objeto es ligero, puesto que a menudo hacen uso de él en sus tareas cotidianas.
- **Ascensor:** En la mayoría de ascensores se puede observar que al lado de los botones de subida o bajada existen unos puntos que hacen referencia al número de piso en Braille**.**
- **Semáforos:** En lugares muy transitados podemos encontrar semáforos que llevan incorporado un sistema que avisa a la persona ciega cuando puede pasar. Este aviso se hace de manera acústica con un timbre que suena cuando el semáforo está en verde**.**

Perro Guía



Máquina de Perkins

Imagen 1.6: Recursos para personas con Discapacidad Visual.
Fuente: Wikipedia.org

2. Accesibilidad en la Web

El concepto de accesibilidad es la base para el trabajo de integración de los ciudadanos con discapacidad al mundo de Internet, incrementando la utilidad y el atractivo de los sitios Web existentes. En un contexto general se puede definir que la accesibilidad consiste en el "*acceso a la información sin limitación alguna por razón de deficiencia, discapacidad o minusvalía*", de modo que la accesibilidad no es de interés únicamente para las personas con discapacidad sino que mejora el acceso Web en general (CTIC, 2009).

Hoy, la accesibilidad se ha introducido fuertemente en el uso de Internet. En relación con ello, el *World Wide Web Consortium* (W3C) ha publicado el *Web Content Accesibility Guidelines* (WCAG), como una recomendación general sobre la materia, siendo validado y utilizado como un marco de referencia para la adopción de medidas en el Gobierno de los Estados Unidos de Norte América, la Comunidad Europea e instituciones para ciegos como por ejemplo la ONCE (ver Glosario), entre otras innumerables organizaciones que han velado por la universalidad del uso de la informática (SIDAR, 2002).

Otro aspecto importante a considerar en materia de accesibilidad son las ayudas técnicas. Éstas son elementos, ya sean software o hardware, que permiten facilitar el uso de alguna herramienta informática, en contexto son los elementos que facilitan la interacción entre las personas con discapacidad y el computador; por ejemplo, un lector de pantalla es una ayuda técnica que permite a las personas con discapacidad visual interactuar con un computador.

Los principales beneficios en Internet que posee la accesibilidad son (ONCE, 2007):

- Facilita el acceso a los sitios Web del Gobierno. De esta forma, las personas con discapacidad visual podrán ingresar a una gran fuente de información y beneficios, que

permite mejorar notablemente su calidad de vida, además apoya la integración de distintos grupos de usuarios con discapacidad.

- Impulsa el trabajo a través del teléfono e Internet. Así, se constituye en ambas herramientas nuevas oportunidades a las personas con discapacidad visual a introducirse desde sus hogares al mundo laboral.
- Materializa la responsabilidad social del Gobierno con los grupos de usuarios con discapacidad del país.
- Permite un rápido acceso a usuarios con una conexión limitada en velocidad, pues la información presentada por medios distintos al texto, debe ser alternativamente representada por descripciones permitiendo la inhibición de imágenes, sonidos y otros medios por parte de estos usuarios.
- Mejora notoriamente la claridad y velocidad en la navegación y por lo tanto el acceso a la información y la usabilidad del sitio mejora de modo directamente proporcional.
- Facilita el acceso a la información con independencia en el dispositivo de acceso utilizado.

2.1. Pautas WCAG 2.0

Los documentos denominados Pautas de Accesibilidad al Contenido en la Web (WCAG) 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.; las WCAG son parte de una serie de pautas de accesibilidad, que incluyen las Pautas de Accesibilidad para Herramientas de Autor (ATAG*,* en inglés *Authoring Tool Accesibility Guidelines*) y las Pautas de Accesibilidad para Agentes de Usuario (UAAG*,* en inglés *User Agents Accesibility Guidelines*) (W3C-WAI, 1998).

Las Pautas WCAG 2.0 son la culminación de un largo proceso que ha generado por el camino un intenso debate, no exento de polémica, para lograr conciliar las distintas visiones de los diferentes sectores implicados sobre la Accesibilidad Web. El consenso alcanzado es el fruto de los esfuerzos de muchas empresas, instituciones, desarrolladores, técnicos, evaluadores, personas con discapacidad y, en general, de toda la comunidad (INTECO, 2009).

2.1.1. Documentos de la Recomendación

Las Pautas WCAG 2.0, al igual que en la primera versión, están organizadas en varios documentos, algunos de los cuales se consideran informativos o de apoyo para una apropiada comprensión de la Recomendación. En este apartado se detallan los distintos tipos de documentos y la relación que existe entre ellos.

En el caso de las WCAG 2.0 estos documentos son cuatro. El documento principal es la Recomendación en sí, y se considera normativo, es decir, es un documento estable y que no admite modificación. Los otros tres son documentos informativos y su contenido está sujeto a posibles cambios y mejoras durante la vigencia de las Pautas.

2.1.1.1. Recomendación WCAG 2.0: El documento de las Pautas

El documento principal que conforma la Recomendación WCAG 2.0 es el documento de las Pautas, "*Web Content Accessibility Guidelines (WCAG) 2.0",* en inglés. Este es el documento normativo; es estable y no cambiará con el tiempo.

Para que dicho documento no sea excesivamente largo, no describe en profundidad cómo cumplir con cada criterio de éxito, evitando entrar en detalle. Sin embargo, en el día a día lo más normal es usar los documentos de soporte, más detallados, y que se describen a continuación.

2.1.1.2. Técnicas para el WCAG 2.0

El documento ("*Techniques for WCAG 2.0",* en inglés) recoge un conjunto de soluciones técnicas que se pueden utilizar para cumplir con los criterios de accesibilidad recogidos en las Pautas. En este sentido es similar al documento correspondiente de WCAG 1.0. Las técnicas pueden ser generales o específicas a una tecnología concreta, como pueden ser HTML/XHTML, CSS, scripts, multimedia y WAI-ARIA. Las Técnicas pueden ser "de suficiencia" (si se aplica esta técnica se considerará cumplido el criterio) o "complementarias" (el uso de esta técnica puede ayudar a mejorar la accesibilidad, pero no implica necesariamente que se esté cumpliendo el criterio).

2.1.1.3. Comprender el WCAG 2.0

El documento ("*Understanding WCAG 2.0",* en inglés) proporciona explicaciones más detalladas sobre el contenido de la Recomendación, a modo de aclaraciones sobre el significado de las Pautas y de los criterios de éxito, así como la intención del grupo de trabajo, cómo ayudan estos criterios a personas con diferentes discapacidades, notas sobre el soporte en navegadores y productos de apoyo, ejemplos y enlaces a otros recursos.

2.1.1.4. Cómo cumplir con WCAG 2.0

El documento ("*How to meet WCAG 2.0",* en inglés) es una guía rápida para los criterios de éxito. Para cada uno de ellos proporciona una lista de enlaces a las Técnicas de suficiencia en el documento de Técnicas. Es una página dinámica que se puede configurar a través de una serie de casillas de verificación, según el nivel de cumplimiento deseado y las tecnologías usadas. Por ejemplo, si sólo se está interesado en CSS y Nivel AA, es posible configurar este documento para leer sólo los criterios y técnicas relevantes para dichas características.

2.1.2. Antecedentes: Pautas WCAG 1.0

En 1999 se publicaron las Pautas de Accesibilidad al Contenido de la Web 1.0, fruto de varios años de trabajo dentro de la WAI y anteriormente en diferentes organismos como la Universidad de Wisconsin-Madison, a mediados de los años 90.

Las WCAG 1.0 están organizadas en 14 Pautas generales, que conforman los principios generales de accesibilidad que se deben tener en cuenta al crear contenidos en la Web. Dentro de cada Pauta se encuentran una serie de puntos de verificación, con una orientación práctica, que ofrecen explicaciones técnicas acerca de cómo hacer accesibles los contenidos, teniendo en cuenta los diferentes elementos usados en dichos contenidos. Estos puntos de verificación estaban asociados a tres niveles de prioridad, relacionados a su vez con los tres niveles de cumplimiento A, AA y AAA.

2.1.3. Carencias de las pautas WCAG 1.0

Las Pautas WCAG 1.0 eran (INTECO, 2009):

- **Interpretables:** diferentes personas las pueden interpretar a su manera de formas muy distintas.
- **Limitadas a tecnologías W3C:** Las Pautas WCAG 1.0 se basa en el supuesto de que HTML es la única tecnología con soporte para la accesibilidad.
- **No incluyen nuevos usos de tecnologías W3C existentes:** por ejemplo, los nuevos usos de HTML y JavaScript en AJAX no se tratan.
- **Rígidas:** Se redactaron en un periodo de rápido avance en las tecnologías de acceso, pero no se actualizó: muchos de sus puntos se cualifican con la frase "Hasta que los agentes de usuario...".

Como resultado de estas limitaciones, se crearon numerosas adaptaciones de las pautas en diferentes países, incluso incorporando éstas en la legislación nacional. Esto ha generado ciertos problemas para la creación de herramientas de evaluación y corrección, ha dificultado su implementación en navegadores y productos de apoyo y ha impedido la creación de un sello de certificación reconocido internacionalmente.

Empresas que se han esforzado para hacer accesibles sus tecnologías han tenido dificultades para lograr su aceptación por no haber sido definidas mediante el proceso del W3C. En todo caso, las Pautas WCAG 1.0 son el punto de partida de las sucesivas recomendaciones y siguen siendo un referente válido para solucionar las principales barreras de accesibilidad.

2.1.4. Estructura y Organización de las Pautas WCAG 2.0

La jerarquía de los componentes de las Pautas WCAG 2.0 está estructurada en cuatro niveles:

- Principios fundamentales (normativos).
- Pautas (normativas).
- Criterios de éxito (normativos).
- Técnicas de éxito y fallos comunes (sólo informativos).

El WCAG 1.0 tiene Pautas, y cada una de ellas tiene sus Puntos de Verificación. Estos puntos son la base para determinar el cumplimiento; WCAG 2.0 tiene cuatro Principios fundamentales, que a su vez contienen Pautas. Cada pauta tiene sus Criterios de Éxito. Estos últimos son la base para determinar el cumplimiento. Para cada criterio de éxito se proporcionan técnicas y fallos comunes que son orientativos.

2.1.5. Principios Básicos, Pautas y Criterios de éxito

Las nuevas Pautas WCAG 2.0 presentan una estructura con ciertas similitudes, aunque también con notables diferencias.

En primer lugar, las Pautas se organizan en **cuatro Principios básicos** (Perceptible, Operable, Comprensible y Robusto), que constituyen la base filosófica de las Pautas.

Dentro de cada Principio básico se encuentran las **Pautas** en sí, también de carácter general, aunque referidas a aspectos específicos de cada Principio básico.

Por último, cada Pauta se desarrolla en una serie de **Criterios de Éxito,** que de forma similar a los puntos de verificación en WCAG 1.0, establecen una serie de criterios de accesibilidad que deben cumplir los contenidos web, y que pueden ser verificados para comprobar el cumplimiento de las Pautas. Los criterios de éxito están clasificados por niveles de conformidad A (el menos exigente), AA y AAA (el más exigente); un mismo criterio puede ocurrir con ligeras diferencias en distintos niveles. Son independientes de la tecnología usada para crear el contenido. Los criterios se han redactado para ser verificados sin ambigüedad, por una herramienta automática o por una persona. Cada criterio de éxito incluye un enlace al apartado correspondiente en los demás documentos de soporte.

Además, cada criterio de éxito puede enlazar con diversas **Técnicas,** que pueden ser de dos tipos:

- **Técnicas de suficiencia**: si se sigue esta técnica se cumple con el criterio para el elemento en el que se realice la verificación.
- **Técnicas complementarias**: son técnicas que ayudan a mejorar la accesibilidad, pero que no garantizan el completo cumplimiento de los criterios.

Las Técnicas no se consideran normativas ni obligatorias, sino que sólo recogen recomendaciones de soluciones conocidas adaptadas a diversas tecnologías, pero que no tienen por qué ser las únicas posibles soluciones. Además, se incluyen también referencias a condiciones de fallo, que consisten en técnicas erróneas y malas prácticas que se sabe que provocan incumplimientos de los criterios.

2.1.6. Resumen de Principios Básicos y Pautas de Accesibilidad

A continuación se recoge un breve resumen con la descripción que se hace en la Recomendación de los Principios Básicos y de las Pautas incluidas en cada Principio, sin entrar a profundizar en los criterios de éxito específicos de cada Pauta (W3C-WCAG, 2008).

Principio 1: Perceptible

"La información y los elementos de la interfaz de usuario deben presentarse a los usuarios de formas en las que los usuarios puedan percibirlos."

Dentro de este principio se recogen 4 Pautas:

- **Pauta 1.1: Alternativas textuales.** Proporcione alternativas textuales para cualquier contenido no textual, de modo que pueda ser alterado formas acordes a las necesidades de las personas, como texto de gran formato, braille, síntesis de voz o un lenguaje más simple.
- **Pauta 1.2: Alternativa para multimedia tempo-dependientes.** Proporcione alternativas para el contenido basado multimedia en el tiempo.
- **Pauta 1.3: Adaptable.** Cree contenido que pueda ser presentado de diferentes formas (por ejemplo, un esquema de presentación más simple) sin perder información o estructura.
- **Pauta 1.4: Distinguible (vista y oído).** Facilite a los usuarios ver y escuchar el contenido, incluyendo la separación entre fondo y primer plano

Principio 2: Operable

"Los componentes de la interfaz y la navegación deben ser operables".

Este Principio contiene 4 Pautas:

- **Pauta 2.1: Acceso mediante teclado.** Haga toda la funcionalidad disponible desde teclado.
- **Pauta 2.2: Suficiente tiempo.** Proporcione a los usuarios suficiente tiempo para leer y usar el contenido.
- **Pauta 2.3: Destellos.** No diseñe el contenido en formas que se conoce que pueden provocar ataques epilépticos.
- **Pauta 2.4: Navegable.** Proporcione formas de ayudar a los usuarios a navegar el contenido y determinar dónde están.

Principio 3: Comprensible

"La información y el manejo de la interfaz de usuario debe ser comprensible".

Existen 3 Pautas bajo este Principio:

- **Pauta 3.1: Legible y entendible.** Haga el contenido textual legible y comprensible.
- **Pauta 3.2: Predecible.** Haga que las páginas web aparezcan y se manejen de manera predecible.
- **Pauta 3.3: Ayuda a la entrada de datos.** Ayude a los usuarios a evitar y corregir los errores.

Principio 4: Robusto

"El contenido debe ser suficientemente robusto para que pueda ser interpretado por una amplia variedad de agentes de usuario, incluyendo los productos de apoyo."

Este Principio solo contiene una Pauta:

- **Pauta 4.1: Compatible.** Maximice la compatibilidad con los agentes de usuario actuales y futuros, incluyendo los productos de apoyo.

En la Imagen 1.7 se presenta la jerarquía de organización de las Pautas WCAG 2.0. En el grupo "Técnicas de Éxito y Fallos Comunes" se indican técnicas específicas para crear contenido accesible; en algunos elementos se destaca una tecnología involucrada (HTML, Flash, entre otras), y en otros casos, se hace referencia a una técnica de tipo general que implica a elementos creados con diversas tecnologías.

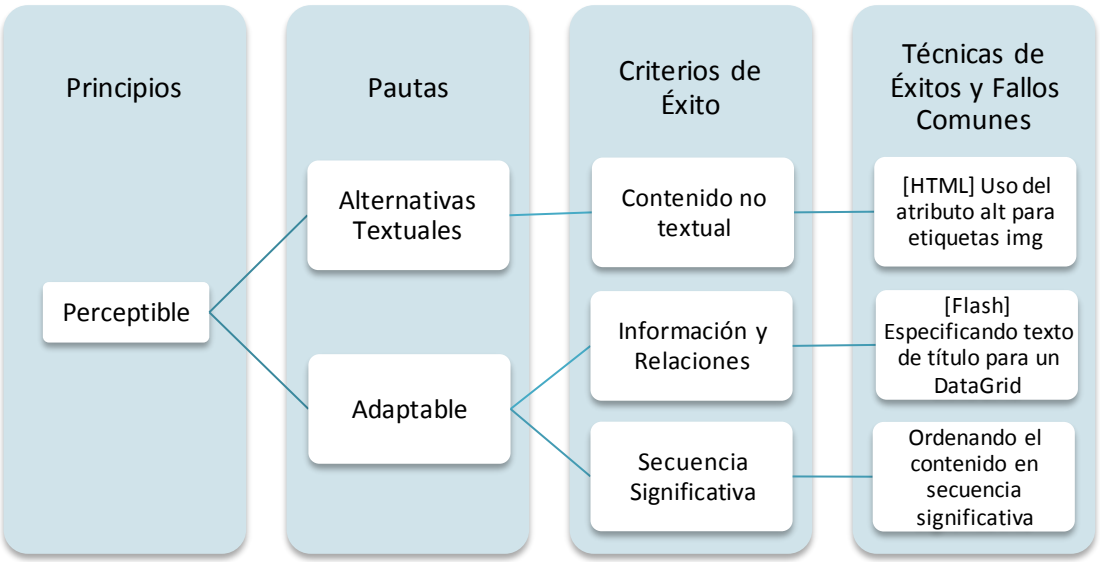


Imagen 1.7: Gráfico jerárquico de las Pautas WCAG 2.0.
Fuente: Elaboración Propia.

En la herramienta HEVAC se utilizan "Técnicas de Éxitos y Fallos Comunes" asociadas a elementos generados con la tecnología HTML y algunas de tipo general. Las mismas pueden ser consultadas en el apartado (W3C-WAI-2, 2008).

Un total de 25 técnicas fueron incorporadas al verificador de accesibilidad de manera programada. Estas técnicas están directamente relacionadas a apoyar criterios de accesibilidad para personas con discapacidad visual, aunque también ayudan a la accesibilidad de contenidos para personas con discapacidad auditiva y motora.

Debido a que existen pocas técnicas citadas en las pautas de accesibilidad relacionadas con la tecnología CSS (Hojas de Estilo en Cascada), además de requerir la incorporación de un analizador de esta tecnología, las técnicas CSS no serán incorporadas a la herramienta, y de igual manera no se incluirán verificaciones a elementos de *JavaScripts* que utilice la página a verificar ya que se requiere interpretar el código del *script* para comprobar si está realizando modificaciones la página Web, la hoja de estilo en cascada o el mismo *script*.

Es de destacar que en versiones futuras se pueden incorporar a la herramienta más técnicas para así ampliar los resultados de las verificaciones que se realizan a las páginas Web.

2.2. WCAG 2.0 en relación a elementos y tecnologías específicas

En esta sección se explicará el tratamiento que dan las nuevas Pautas a algunos de los elementos y tecnologías que se encuentran con más frecuencia en el contenido Web, indicando las posibles diferencias de enfoque con respecto de las WCAG 1.0, así como los Criterios de Éxito que deben ser considerados, teniendo en cuenta un Nivel de Conformidad AA.

Debido a la nueva filosofía de las Pautas WCAG 2.0, la correspondencia entre puntos de verificación de WCAG 1.0 y criterios de éxito de WCAG 2.0 no es ni mucho menos directa, sino tan sólo una aproximación para establecer algunas analogías que faciliten la transición a quienes ya conocen las Pautas WCAG 1.0. Algunos puntos de verificación ya no tienen sentido en las nuevas Pautas (por ejemplo, los que se refieren al uso de tecnologías específicas del W3C), al tiempo que existen nuevos criterios de éxito que las WCAG 1.0 no contemplaban. Además, dada la nueva estructura de las WCAG 2.0, en algunos casos la correspondencia no será con un criterio de éxito concreto, sino más bien con un principio más general o un requisito de conformidad (INTECO, 2009).

2.2.1. Imágenes y Mapas de Imagen

Las imágenes son uno de los elementos fundamentales en todo contenido Web, por lo que su accesibilidad es clave en la percepción global del sitio. En el contexto de las WCAG 2.0, tecnológicamente neutrales, el concepto de "imagen" puede referirse tanto a archivos en mapa de bits insertados como tal en una página Web (PNG, JPG o GIF, por ejemplo), como a imágenes insertadas dentro de otro tipo de documentos o tecnologías, incluyendo también cualquier representación gráfica no textual, como pueden ser gráficos vectoriales, iconos y símbolos o ASCII Art. Por lo tanto, a diferencia de las Pautas WCAG 1.0, las alternativas a las imágenes pueden darse de maneras muy distintas dependiendo de la tecnología que se esté usando, dado que no siempre se insertarán las imágenes mediante una etiqueta HTML.

En el Anexo 2.1 se encuentran los renglones correspondientes a las imágenes para ser analizados como elementos de accesibilidad para las Pautas WCAG 2.0.

2.2.2. Multimedia (Audio, Vídeo y Presentaciones)

En este apartado, el término "multimedia" hace referencia a los contenidos que se presentan en formatos no textuales, tales como audio, vídeo, animaciones o presentaciones interactivas, entendidas estas últimas como aquellas presentaciones más o menos simples, donde el usuario tiene un cierto control sobre la visualización del contenido, pero sin mayores posibilidades de interacción. La clave en este aspecto son los distintos formatos multimedia (no textuales) en los que se puede presentar la información, sin llegar a alcanzar la categoría de

objetos programados, donde las posibilidades de interactividad y control por parte del usuario son mucho mayores (ver el apartado correspondiente a Objetos Programados).

Las Pautas WCAG 2.0 recogen muchos criterios que deben ser tenidos en cuenta para los elementos multimedia, tales como alternativas textuales a dichos elementos, audio-descripción y subtitulado, o la interfaz y el control por parte del usuario de la reproducción de contenido multimedia: La información de estos elementos multimedia es ampliada en el Anexo 2.2.

2.2.3. Objetos Programados, Scripts y AJAX

Debido a la neutralidad tecnológica de las Pautas WCAG 2.0, ya no existe distinción entre este tipo de contenidos y los realizados puramente con HTML y CSS. En esta nueva concepción, un objeto incrustado en una página Web, un *applet* Java o un script pueden ser completamente accesibles sin requerir una alternativa, siempre y cuando la tecnología usada tenga soporte para la accesibilidad y el objeto o *script* se haya desarrollado de manera accesible. Asimismo, siendo AJAX tan sólo un uso concreto de JavaScript, se trata del mismo modo que el resto de contenidos, pudiendo ser totalmente accesible si la programación se realiza de la forma adecuada.

Por lo tanto, a este tipo de contenidos son aplicables la totalidad de las Pautas WCAG 2.0, si bien en este apartado se mencionarán tan sólo aquellos que se pueden considerar más específicos de este tipo de tecnologías.

Existe un borrador con mayor información sobre roles y aplicaciones accesibles pertenecientes al W3C llamado "*Accesible Rich Internet Applications*" (ARIA, en inglés).

2.2.4. Independencia del Dispositivo

La independencia del dispositivo es un aspecto clave en las Pautas WCAG 2.0, recogida en la Pauta 2.1, que hace referencia a la accesibilidad mediante teclado. Esta Pauta engloba varios criterios de éxito que de algún modo constituyen la evolución de los puntos de verificación en las Pautas WCAG 1.0, ya que las nuevas Pautas son tecnológicamente neutrales, y por lo tanto no se habla ya de "eventos de script" o de "objetos con interfaz propia", sino que todo el contenido debe cumplir con la independencia del dispositivo, independientemente de cuál sea la tecnología usada.

También debe tenerse en cuenta el Requisito de Conformidad 5, relativo a la no-interferencia sobre el acceso al resto de los contenidos. En este sentido, el criterio de éxito 2.1.2 (Nivel A) hace referencia a las "trampas de teclado", una práctica observada con bastante frecuencia en las páginas Web, donde una mala detección de los eventos de teclado resulta en la imposibilidad de acceder a todos los contenidos, o en el mejor de los casos a efectos indeseados

que dificultan el acceso. Este "efecto trampa" suele producirse cuando en un enlace o botón destinado a una función específica (por ejemplo, "imprimir" o "buscar") se introduce un evento de teclado que se ejecuta al pulsar una tecla, pero sin detectar cuál es la tecla pulsada. En esos casos, un usuario que navegue mediante teclado, usando la tecla de tabulación, puede quedar "atrapado" en dicho enlace, o sin poder acceder al resto de contenidos, ya que en la siguiente pulsación de la tecla *Tab* se ejecutará la función del evento, en lugar de continuar navegando por la página.

Por otro lado, los criterios de éxito relativos al orden de tabulación, indican que, si una página se puede navegar de forma secuencial y el orden de navegación afecta a la comprensión o al manejo, los elementos que reciben el foco lo hagan de una forma que preserve el significado de los contenidos.

Técnica de Ejemplo

Actualmente la mayor parte de los exploradores de Internet proveen la navegación mediante la tecla tabulador entre enlaces y formularios, esta navegación comienza generalmente en el primer enlace o control de formulario dentro de la página, continuando en orden de aparición dentro del código HTML. En caso de necesitar un orden distinto al anteriormente descrito, se debe usar el atributo `tabindex` en los elementos afectados. Es aconsejable el uso del atributo anteriormente descrito para asegurar el orden deseado de navegación.

```
<form name="Jan">
  <input type="text" tabindex="1" value="N&uacute;mero uno" name="burns">
  <input type="text" tabindex="3" value="N&uacute;mero tres">
  <input type="text" tabindex="5" value="N&uacute;mero cinco">
  <input type="text" tabindex="6" value="N&uacute;mero seis">
  <input type="text" tabindex="4" value="N&uacute;mero cuatro">
  <input type="text" tabindex="2" value="N&uacute;mero dos">
</form>
```

El anterior ejemplo se navega según el orden que aparece dentro de las cajas y se ve de la siguiente forma:

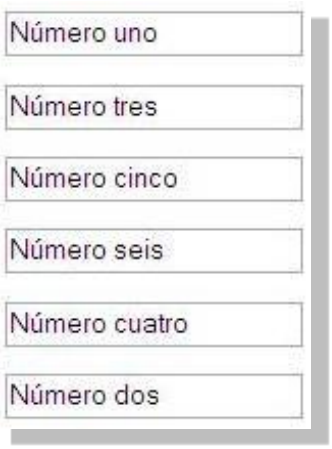


Imagen 1.8: Campos HTML Indexados por tabulaciones.
Fuente: Elaboración Propia.

2.2.5. Destellos, Parpadeos, Movimiento y Actualización Automática

Si en las WCAG 1.0 se especificaba que se debía evitar provocar destellos en la pantalla (cambios bruscos de luminosidad) para evitar posibles ataques epilépticos, los nuevos criterios de las Pautas WCAG 2.0 es aún más específico, indicando que no debe existir ningún contenido que produzca más de tres destellos en cualquier período de un segundo, a menos que estos destellos ocupen un área inferior al 25% de un campo visual de 10º a una distancia normal de trabajo. Debe tenerse en cuenta, además, que cualquier contenido que incumpla este criterio de éxito puede interferir con la capacidad del usuario para acceder a la totalidad de la página Web, por lo que estaría incumpliéndose también el Requisito de Conformidad relativo a la no interferencia.

Igualmente, de forma similar al punto de verificación 7.4 de WCAG 1.0, el criterio de éxito anterior establece que si existen contenidos que se actualizan automáticamente y se muestran en paralelo a otra información, debe existir un mecanismo que permita pausar, detener u ocultar esta actualización, o que permita al usuario controlar la frecuencia de actualización, a menos que esta actualización sea esencial para transmitir la información.

Por último, se indica que cualquier límite de tiempo que exista en el contenido, el usuario debe ser capaz de desactivar, ajustar o extender dicho límite, salvo en los casos donde este límite sea parte fundamental del proceso o tarea en cuestión.

Técnicas de Ejemplo.

La etiqueta `<blink>`, que proporciona el efecto de parpadeo, no está dentro de los estándares HTML, por lo que su uso no está permitido. En otros medios que contengan parpadeo como animaciones, videos, etcétera, es necesario proporcionar un control para el usuario, de modo que puedan deshabilitar este elemento.

2.2.6. Maquetación y Presentación

Las Pautas WCAG 1.0 estaban orientadas a las tecnologías desarrolladas por el W3C, por lo que hacían referencia explícita al uso de CSS para maquetar y presentar la información así como a la accesibilidad del contenido en caso de no estar disponibles las hojas de estilo.

En las WCAG 2.0, tecnológicamente neutrales, esta limitación ha desaparecido, admitiéndose como válida cualquier tecnología que tenga soporte para la accesibilidad, siempre y cuando el desarrollo se haga de manera accesible.

Se analizará el tratamiento que se hace en las WCAG 2.0 sobre algunas técnicas habituales de maquetación y otros aspectos relacionados con la presentación del contenido en el Anexo 2.3.

2.2.7. Encabezados, Listas, Citas y otros Bloques de Texto

En este apartado se incluyen una serie de criterios relativos al marcado estructural de contenidos, lo que permite identificar la funcionalidad y las relaciones entre los distintos elementos de la página Web.

Así, algunos de los puntos de verificación de WCAG 1.0, se pueden encontrar ahora bajo un criterio de éxito más genérico en la versión 2.0, que especifica que la información, estructura y relaciones transmitidas mediante presentación deben poder ser determinadas programáticamente, esto es, el agente de usuario debe poder inferirlas a partir del código o del marcado estructural, o bien estar disponibles en forma de texto. La importancia de este criterio de éxito se encuentra más allá del enunciado en sí, encontrándose en los documentos de apoyo una gran cantidad de técnicas de implementación, así como de condiciones de fallo conocidas, relacionadas con el marcado semántico y estructural de contenidos.

Asimismo, existen diversas técnicas relativas al marcado en HTML de listas, bloques, elementos de formulario, tablas de datos, elementos de énfasis, etc.; en el caso concreto del marcado de citas, por ejemplo, también se contempla específicamente como condición de fallo el uso de los elementos de cita sólo para provocar efectos visuales, y no con carácter estructural.

Técnica de Ejemplo

Con la frase de ejemplo "Las páginas que aprueban el WCAG", debería codificarse:

```
<p>Las p&aacute;ginas que aprueban el <acronym title="Web Content Accessibility Guidelines" lang="en">WCAG</acronym></p>
```

2.2.8. Idiomas y Comprensión del Lenguaje

En las Pautas WCAG 2.0, los criterios relativos al entendimiento del contenido, tanto en lo relativo al idioma como en lo referente al tipo de lenguaje usado, indican que el contenido debe ser legible y entendible.

Así, los criterios de éxito en la nueva versión, relativos al idioma de la página, que establece que el idioma de la página debe poder determinarse de forma programática (el agente de usuario debe poder inferirlo a partir del código, del marcado o de alguna otra propiedad a la que el agente de usuario tenga acceso). También se establece que el idioma de las diferentes partes o frases del contenido pueda determinarse programáticamente, e indicando algunas excepciones como nombres propios, términos técnicos u otro tipo de palabras de idioma indeterminado o que formen parte de texto circundante.

Técnica de Ejemplo.

Se debe incluir el lenguaje natural o de origen principal de cada página Web. En HTML, utilizando la marca `<html>` y el atributo `"lang"`. Ejemplo de una página en español: `<html lang ="es">`. Cada idioma se identifica con un código de letras distintas, Alemán=de, Inglés=en, Francés=fr, Italiano=it, Portugués=pt.

2.2.9. Tablas de Datos

Los criterios de éxito de este apartado son muy amplios, recogiendo de forma genérica el uso adecuado del marcado estructural de los contenidos. En lo relativo a las tablas, se pueden encontrar referencias específicas en las Técnicas: Uso de elementos de marcado de tablas para presentar información tabular, Uso del atributo `scope` y Uso de los atributos `id` y `headers` para asociar celdas de datos con celdas de cabecera en tablas de datos.

En los puntos de verificación de la versión 1.0, relativo a los títulos y resúmenes de tablas de datos, en las WCAG 2.0 ya no hay referencias a esta exigencia, aunque sí se contempla como fallo de cumplimiento el uso del atributo `summary` en tablas de maquetación, donde su uso es inadecuado referenciado en los documentos WCAG 2.0 como una condición de fallo.

Técnicas de Ejemplo:

Al crear una tabla que represente una agenda telefónica se debe indicar el título de la tabla de modo que el usuario entienda el contexto de la tabla y especificar con los atributos correctos los encabezados de columnas de la tabla. En HTML se codificaría de esta manera:

```
<table summary="Esta tabla contiene información de la agenda">
```

```
  <caption>Mi guía telefónica</caption>
  <tr>
    <th scope="col">Nombre</th>
    <th scope="col">Teléfono</th>
  </tr>
  <tr>
    <td>Juan Pérez</td>
    <td>6453723</td>
  </tr>
</table>
```

Con la etiqueta `<caption>` se referencia el título de la tabla, con esto los usuarios entenderán el contexto en el que se sitúa la tabla. Con la etiqueta `<th>` se identifica los encabezados de las columnas de las tablas, de este modo será fácil distinguir el tipo de información que entrega la tabla. Usando el atributo `scope` se podrá identificar el tipo de celda de encabezado. Ejemplo: `scope="col"` indica que esta celda es la cabecera de todas las demás celdas que están en la misma columna.

2.2.10. Marcos

En la versión WCAG 1.0, que describe la necesidad de titular cada marco para facilitar su identificación y navegación, tiene ahora una cierta equivalencia que indica que se debe especificar el nombre y rol de los elementos, así como permitir el ajuste de aquellos valores o propiedades que sean relevantes.

También, como parte de la versión anterior, relativo a la descripción larga de las relaciones entre marcos, en las WCAG 2.0 no hay tal necesidad, además de que en las nuevas versiones de XHTML ya no existe el atributo `longdesc` para el elemento `frame`.

2.2.11. Formularios

La interacción del usuario con el contenido Web es uno de los aspectos que más ha evolucionado en las Pautas WCAG 2.0, que incorporan diversos criterios de éxito relativos al control y prevención de errores al introducir datos, el etiquetado de los controles o la presencia de instrucciones para que el usuario sepa cómo debe rellenar los campos, la mayor de los cuales se agrupan bajo la **Pauta 3.3**, "Ayudar a los usuarios a evitar y corregir errores".

En los criterios de éxitos referentes a este aspecto, se establece que si se detecta automáticamente un error en la entrada del usuario se proporcionen, en la medida de lo posible, sugerencias que ayuden a corregir el error, siempre que éstas no comprometan la seguridad o el propósito del contenido.

Asimismo, si la interacción con la página tiene algún tipo de consecuencia legal (un acuerdo o contrato), económico (compras o transacciones financieras), o de gestión de datos del usuario, el

criterio de éxito sobre prevención de errores exige que se cumpla al menos una de las siguientes posibilidades:

- **Reversibilidad**: si se puede volver al estado anterior a la entrada de datos.
- **Verificación**: si los datos introducidos son analizados ante posibles errores y se le da la oportunidad al usuario de corregirlos.
- **Confirmación**: si existe un mecanismo para revisar, confirmar y corregir la información antes de terminar de enviarla.

En las Pautas WCAG 2.0, se establece que se deben proporcionar etiquetas o instrucciones cuando los controles requieren entrada por parte del usuario; sin embargo, y atendiendo a la neutralidad tecnológica, entre otras cosas, no se requiere una manera específica de implementar estas etiquetas, admitiéndose como válidas distintas técnicas.

En este sentido, en HTML se pueden incluir las etiquetas de la manera "tradicional", es decir, usando el elemento `label` y asociándolo a su control mediante la combinación de atributos `for-id`, pero también se admite la posibilidad de que la etiqueta se asigne mediante un atributo `title` en el control, ya que las nuevas Pautas 2.0 consideran válido cualquier método que permita determinar programáticamente el contenido de la etiqueta. Es de señalar también que ya no se considera como un requisito el posicionar la etiqueta de una determinada manera con respecto del control, ya que los agentes de usuario y productos de apoyo actuales son capaces de reconocer la asociación explícita entre etiquetas y controles, sin tener que "adivinar" cuál es la etiqueta en base a su posición.

Técnicas de Ejemplo:

En HTML es necesario utilizar la marca `<label>` y su propiedad `"for"` para identificar correctamente las etiquetas. En el siguiente ejemplo se indica la forma de hacerlo.

```
<form action="http://www.tusdatos.com/tusnombres" method="post">
  <p>Identificaci&oacute;n</p>
  <p>
    <label for="nombres">Nombres</label>
    <input type="text" id="nombres">
    <br>
    <label for="apellidos">Apellidos</label>
    <input type="text" id="apellidos">
    <br>
  </p>
  <p>
    <input type="submit" value="ENVIAR">
    <input type="reset" value="LIMPIAR">
  </p>
</form>
```

El resultado de este código es el siguiente:

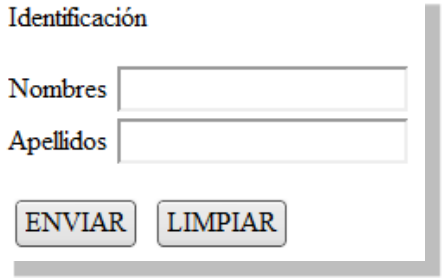


Imagen 1.9: Formulario con campos identificados.
Fuente: Elaboración Propia.

2.2.12. Metadatos

En las Pautas WCAG 1.0, relativo a la inclusión de metadatos, ha quedado obsoleto en las nuevas WCAG 2.0, no siendo ya un requisito. No obstante, existen diversas técnicas relacionadas con el uso de metadatos y cómo este uso puede favorecer el cumplimiento de las WCAG 2.0, por ejemplo, proporcionando información sobre páginas alternativas, diferentes formas de acceder a la información o recursos que faciliten la navegación.

Por otro lado, existe un criterio de éxito que establece que las páginas deben tener un título que describa su función o contenido. Si bien este criterio no se refiere específicamente al elemento `title` de las páginas (que puede considerarse un metadato), se considera que proveer un título descriptivo mediante este elemento es una técnica de suficiencia válida para este criterio.

El Anexo 2.4 contiene una mayor información acerca de los componentes considerados como Metadatos para las Pautas WCAG 2.0.

2.2.13. Documentos PDF

Teniendo en cuenta que las Pautas WCAG 2.0 son tecnológicamente neutrales, y que actualmente los PDF tienen soporte para la accesibilidad, no existe en realidad ninguna diferencia entre los documentos PDF y cualquier otro contenido, tratándose solo de una tecnología distinta de HTML con CSS. En este sentido, son de aplicación la totalidad de las Pautas WCAG 2.0, e incluso es posible incrustar multimedia dentro de documentos PDF, siempre que se haga de forma accesible.

En cualquier caso, dado que el uso más común de PDF es la creación de documentos eminentemente textuales, son de especial relevancia los criterios relativos al marcado estructural de los contenidos (tratados en el apartado **Encabezados, listas, citas y otros bloques de texto**), así como los concernientes al uso de Imágenes o Tablas de datos. Otro uso frecuente de

PDF es la creación de formularios que pueden llenarse por el usuario, aspecto que se referencia en el apartado sobre **Formularios** del presente documento.

2.2.14. Adobe Flash

Del mismo modo que con los documentos PDF, la tecnología Adobe Flash se trata en las Pautas WCAG 2.0 como cualquier otro contenido Web, ya que actualmente esta tecnología dispone de soporte para la accesibilidad. Por lo tanto, son igualmente aplicables todos los criterios de las Pautas WCAG 2.0, no siendo necesario proporcionar una alternativa al Flash siempre y cuando éste se haya desarrollado de manera accesible.

Dado que Flash puede usarse para desarrollar sitios Web completos, no es posible concretar unos criterios que tengan mayor importancia que otros. No obstante, y tan sólo a modo de recordatorio, se citan a continuación algunos criterios de especial aplicación en determinados casos de uso de la tecnología Flash:

- **Animaciones sencillas sin interacción por parte del usuario**: pueden aplicar los criterios relativos a Imágenes, Multimedia o Destellos, parpadeos y movimiento.
- **Inclusión de vídeo o audio**: aplicarán los criterios sobre Multimedia, Destellos, parpadeos y movimiento o Independencia del dispositivo.
- **Aplicaciones con interacción del usuario**: son de especial importancia los criterios sobre Objetos programados, Independencia del dispositivo o Formularios.

CAPÍTULO III: MARCO METODOLÓGICO Y DESARROLLO DE LA HERRAMIENTA

El presente capítulo está constituido por dos secciones en las cuales se basa la ejecución del presente Trabajo: el Proceso de Implementación donde se lleva a cabo el desarrollo de la herramienta a través de una metodología, uso de tecnologías de soporte y librerías externas; el Proceso de Pruebas, que lleva a cabo la obtención de resultados al ejecutar la herramienta desarrollada sobre sitios Web existentes.

1. Proceso de Implementación de la Herramienta HEVAC

El proceso de implementación que se presenta a continuación utiliza los principios y métodos establecidos en la Ingeniería del Software, destinado a entregar un enfoque sistemático disciplinado y cuantificable al desarrollo, operación y mantenimiento del software a entregar. Para lograr esto se debe precisar la metodología de desarrollo a aplicar, el cual es una estructura conceptual que es usado para estructurar, planear y controlar el proceso de desarrollo en los sistemas de información.

1.1. Selección de la Metodología de Desarrollo

El Proceso de Implementación inicia seleccionando la metodología de desarrollo XP y obteniendo las funcionalidades o requerimientos de la herramienta por medio de las historias de usuario de esta disciplina, apoyándose además en la etapa del diseño de la herramienta a través de diagramas UML, por lo tanto se puede considerar que la metodología de desarrollo a utilizar durante este proceso es una metodología **Ad-Hoc.**

1.1.1. Programación Extrema

La Programación Extrema o *eXtreme Programming* (XP) es un proceso de desarrollo de software ágil y ligero propuesto por Kent Beck. XP está basado en los principios de simplicidad, comunicación, retroalimentación, coraje y respeto y su principal objetivo es obtener código operativo lo antes posible durante el ciclo de desarrollo (Wells, 2009).

XP se centra en trabajar estrechamente con el cliente y entregar versiones pequeñas del sistema con mucha frecuencia, idealmente cada dos semanas. En cada versión se debe desarrollar lo mínimo y de la manera más simple posible para que funcione correctamente y no tener que agregar complejidad al código. El diseño evoluciona conjuntamente sobre la marcha, haciendo una entrega de diseño para cada una de las entregas del sistema y luego modificándolo en las siguientes versiones. En algunos casos se considera que la documentación es el código fuente (ExtremeProgramming.Org, 1999).

Para desarrollar un proyecto siguiendo el proceso de desarrollo de software XP, se aplica un conjunto mínimo de reglas y prácticas agrupadas en los segmentos de planificación, gerencia, diseño, codificación y pruebas. Las reglas que se aplican en cada uno de estos segmentos son (Jeffries, 2001):

- **Equipo completo:** Forman parte del equipo todas las personas que tienen algo que ver con el proyecto, incluido el cliente y el responsable del proyecto.
- **Planificación:** Se hacen las historias de usuario y se planifica en qué orden se van a hacer las versiones. La planificación se revisa continuamente.
- **Versiones pequeñas:** Las versiones deben ser lo suficientemente pequeñas como para poder hacer una cada pocas semanas. Deben ser versiones que ofrezcan algo útil al usuario final y no trozos de código que no se pueda ver funcionando.
- **Diseño simple:** Hacer siempre lo mínimo imprescindible de la forma más sencilla posible. Mantener siempre sencillo el código.
- **Pareja de programadores:** Los programadores trabajan por parejas y se intercambian las parejas con frecuencia.
- **Desarrollo guiado por las pruebas automáticas:** Se deben realizar programas de pruebas automáticas y deben ejecutarse con mucha frecuencia. Cuantas más pruebas se hagan, mejor.
- **Mejora del diseño:** Mientras se codifica, se debe mejorar el código existente si se considera conveniente con el fin de extraer funcionalidades comunes, eliminar líneas de código innecesarias y de alta complejidad.
- **Integración continua:** Debe tenerse siempre un ejecutable del proyecto que funcione y en cuanto se tenga una nueva pequeña funcionalidad, debe recompilarse y probarse.
- **El código es de todos:** Cualquiera puede y debe manipular y conocer cualquier parte del código. Para ello se hacen las pruebas automáticas.
- **Normas de codificación:** Debe haber un estilo común de codificación, de forma que parezca que ha sido realizado por una única persona.
- **Metáforas:** Se debe buscar frases o nombres que definan cómo funcionan las distintas partes del programa. Esto ayuda a que todos los programadores y el cliente conozcan el módulo del programa que se está evaluando.
- **Ritmo sostenible:** Se debe trabajar en un ritmo que se pueda mantener indefinidamente, es decir, que no debe haber tiempos extensos de ocio entre los desarrolladores o excesos de horas extras de trabajo. Al tener claro semana a semana lo que debe hacerse, el equipo se enfoca para lograr el objetivo cercano de finalizar una historia de usuario o versión.

La herramienta a desarrollar contará con esta metodología de desarrollo, basando su principal argumento en la adaptabilidad de las necesidades de ésta, como por ejemplo las comprobaciones de accesibilidad que se pueden ir incorporando incrementalmente. Sin embargo, la metodología XP fue aplicada parcialmente en el desarrollo de la herramienta, ya que se incorporan otros artefactos en las fases de análisis y diseño, tales como los diagramas de casos de uso, diagramas de componentes, diagramas de secuencia y diagramas de clases.

1.2. Casos de Uso.

Es una técnica de Ingeniería del Software utilizado para capturar una secuencia de acciones realizadas por una entidad externa sobre el sistema, cuyo fin es lograr un objetivo cuantificable. Un caso de uso describe únicamente una característica del sistema.

Un caso de uso describe un medio en el que un actor del mundo real (una persona, organización o sistema externo) interactúa con su organización. Los actores se muestran como figuras, los casos de uso son elipses y el sistema es una caja. Las flechas indican que actor está implicado en qué casos de uso y la dirección de la flecha indica el flujo de información.

Los principales beneficios de los casos de uso son:

- Capturar los requerimientos del sistema.
- Fundamento para el diseño del software.
- Sirven para validar el diseño del software realizado.
- Sustentan las pruebas de la implementación, ya que cada caso de uso es un elemento a verificar y validar.
- Son la base de la documentación en línea y del manual de usuario (Ambler, 2004).

El diagrama de casos de uso de la herramienta Web se presenta de la siguiente manera:

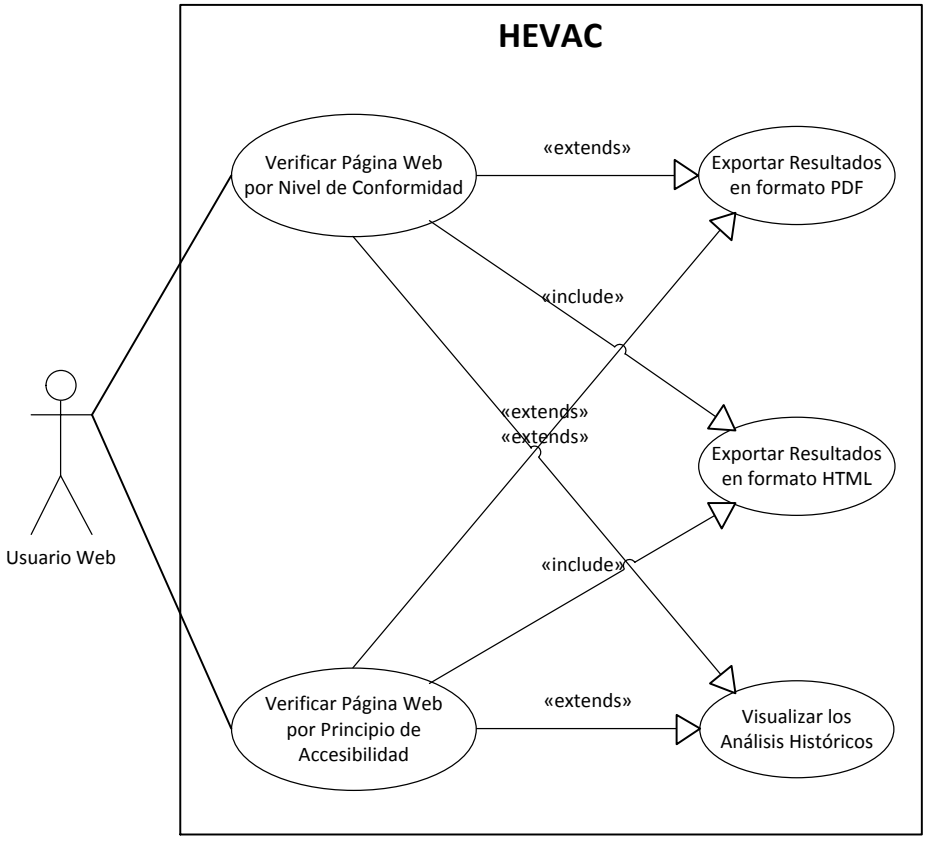


Imagen 2.1: Diagrama de Casos de Uso del Sistema.
Fuente: Elaboración Propia.

1.2.1. Documentación de los casos de uso

Caso de Uso:	Verificar Página Web por Nivel de Conformidad.
Actores:	Usuario Web.
Tipo:	Flujo Básico.
Propósito:	El Usuario requiere que se verifiquen los Principios de Accesibilidad dado su Nivel de Conformidad (Ejemplo: Nivel A, Nivel AA o Nivel AAA).
Resumen:	El sistema verifica el Nivel de Conformidad seleccionado por el Usuario, luego obtiene el URL de la página Web a verificar y busca las incidencias presentes por parte del incumplimiento de las técnicas de accesibilidad aplicadas según el Nivel de Conformidad elegido.
Precondiciones:	El contenido del URL debe ser válido y tiene que estar disponible como un recurso HTML.
Flujo Principal:	• El Usuario ha accedido página principal de la herramienta Web. • El Usuario introduce el URL del sitio Web a verificar. • El Usuario selecciona alguna opción presentada del "Nivel de Conformidad". • El Usuario envía la petición al sistema.
Flujo Alternativo:	Ninguno.
Excepciones	Si la URL introducida está mal formada o no puede ser localizada.

Tabla 2.1: Documentación de Caso de Uso 1.
Fuente: Elaboración propia.

Caso de Uso:	Verificar Página Web por Principio de Accesibilidad.
Actores:	Usuario Web.
Tipo:	Flujo Básico.
Propósito:	El Usuario requiere que se verifiquen los criterios de éxito dado su Principio de Accesibilidad.
Resumen:	El sistema verifica el (los) Principio (s) de Accesibilidad seleccionado (s) por parte del usuario, luego obtiene el URL de la página Web a verificar y busca las incidencias presentes por parte del incumplimiento de las técnicas de accesibilidad aplicadas según el (los) Principio (s) de Accesibilidad elegidos.
Precondiciones:	El contenido de la URL debe ser válido y tiene que estar disponible como un recurso HTML.
Flujo Principal:	• El Usuario ha accedido página principal de la herramienta Web. • El Usuario introduce el URL del sitio Web a verificar. • El Usuario selecciona alguna de las opciones en la sección "Principios de Accesibilidad". • El Usuario envía la petición al sistema.
Flujo Alternativo:	Ninguno.
Excepciones	Si la URL introducida está mal formada o no puede ser localizada.

Tabla 2.2: Documentación de Caso de Uso 2.
Fuente: Elaboración propia.

Caso de Uso:	Exportar Resultados en formato HTML.
Actores:	Usuario Web.
Tipo:	Flujo de Inclusión.
Propósito:	El Usuario requiere que los resultados de las verificaciones sean visualizadas en una nueva página Web.
Resumen:	La herramienta mostrará luego de la verificación de accesibilidad los resultados en una nueva página HTML con los problemas o advertencias encontrados en la página Web analizada.
Precondiciones:	El usuario presiona el botón de verificación y seleccionó la opción "Mostrar los resultados en una nueva página".
Flujo Principal:	• El Usuario ha accedido página principal de la herramienta Web. • El Usuario introdujo el URL con las opciones seleccionadas y pulso el botón "Analizar". • El Usuario visualiza los resultados en una nueva página HTML. • El Usuario tiene la opción de ver resultados detallados con un enlace en la parte inferior de la página mostrada.
Flujo Alternativo:	Ninguno.
Excepciones	Si se interrumpe la operación durante la verificación.

Tabla 2.3: Documentación de Caso de Uso 3.
Fuente: Elaboración propia.

Caso de Uso:	Exportar Resultados en formato PDF.
Actores:	Usuario Web.
Tipo:	Flujo de Extensión.
Propósito:	El Usuario requiere que los resultados de las verificaciones sean

	visualizadas en un archivo en formato PDF.
Resumen:	La herramienta mostrará, luego de la verificación de accesibilidad los resultados en un archivo PDF con los problemas o advertencias encontrados en la página Web analizada. El documento PDF se obtendrá a través de un enlace para descargar el archivo.
Precondiciones:	El usuario realizó la verificación de un URL y luego presionó la opción "Exportar a PDF".
Flujo Principal:	• El Usuario ha accedido página principal de la herramienta Web. • El Usuario introdujo el URL con las opciones seleccionadas y pulso el botón "Analizar". • El Usuario selecciona la vista detallada de los resultados. • El Usuario presiona el botón "Exportar en PDF". • El Usuario descarga el archivo que se le envía desde el servidor.
Flujo Alternativo:	Ninguno
Excepciones	Si se interrumpe la operación durante la verificación.

Tabla 2.4: Documentación de Caso de Uso 4.
Fuente: Elaboración propia

Caso de Uso:	Visualizar los resultados de los análisis históricos.
Actores:	Usuario Web.
Tipo:	Flujo de Extensión.
Propósito:	El Usuario requiere que se muestren los resultados de las verificaciones previamente analizadas a un mismo URL.
Resumen:	El sistema mostrará, luego de la verificación de accesibilidad,
Precondiciones:	El URL verificado debe estar analizado por la herramienta al menos una vez.
Flujo Principal:	• El Usuario ha accedido página principal de la herramienta Web. • El Usuario introdujo el URL con las opciones seleccionadas y pulso el botón "Analizar". • El Usuario selecciona la vista detallada de los resultados y visualiza los resultados históricos si previamente se le ha hecho un análisis a la página consultada.
Flujo Alternativo:	Ninguno
Excepciones	Si se interrumpe la operación durante la verificación.

Tabla 2.5: Documentación de Caso de Uso 5.
Fuente: Elaboración propia

1.3. Componentes del Aplicativo

Un componente de software es una unidad modular con interfaces bien definidas, que es reemplazable dentro del contexto. Así, un componente define su comportamiento en términos de interfaces proporcionadas y requeridas; y dicho componente será totalmente reemplazable por otro que cumpla con las interfaces declaradas. UML (ver glosario) no coloca ninguna restricción respecto a la granularidad del componente, de esta forma un componente podrá ser tan simple como un convertidor de moneda o tan complejo como un sistema de ayuda semántico (OMG, 2011).

Los diagramas de componentes muestran los componentes de software que construyen una pieza de software reutilizable, sus interfaces y sus relaciones. Las cajas representan componentes, en este caso, o bien aplicaciones o subsistemas internos y las líneas punteadas representan dependencias entre componentes. Uno de los objetivos principales del modelado arquitectónico es dividir un sistema en componentes cohesivos que tienen interfaces estables, creando un núcleo que no necesita cambiar en respuesta a los cambios a nivel de subsistema.

Los diagramas de paquetes son una parte de los diagramas de componentes. Un paquete puede contener clases, interfaces u otros paquetes, pudiéndose estos anidar en profundidad. Las flechas punteadas muestran las dependencias entre paquetes.

Un componente puede contener clases, interfaces o incluso módulos o código procedimental, como por ejemplo aplicaciones heredadas.

Los componentes generales que contemplará la herramienta están representados en el siguiente diagrama:

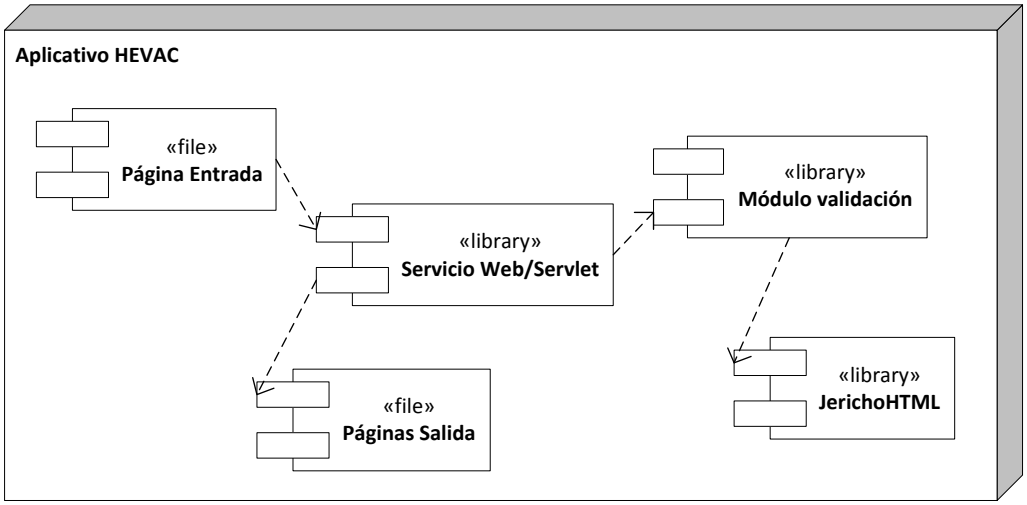


Imagen 2.2: Diagrama de Componentes del Sistema.
Fuente: Elaboración Propia.

El aplicativo fue estructurado con cuatro (4) componentes principales representados por librerías estáticas. Sus funciones son las siguientes:

- **Servlets**: El componente de procesamiento de peticiones y respuestas HTTP en Java (mejor conocido como *Servlets*), forman la capa de entrada y salida. El componente único del Servlet puede ser usado también como servicio Web, así el mismo componente permitirá dos formas distintas de entrada y salida HTTP.

- **Componente de Validación**: Compone todas las clases que realizan la evaluación y la orquestación entre ellas. Este componente representa el 80% del aplicativo.
- **Librería Jericho HTML**: La librería de soporte externa seleccionada para leer las etiquetas HTML y recorrerlas de manera eficiente, forma una parte importante del aplicativo.
- **Páginas de entrada y salida**: Estas páginas forman la interfaz de interacción con el usuario, también se incluyen las hojas de estilos y *Scripts* de validaciones y tareas.

1.4. Estructura de clases

Un **diagrama de clases** es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Los diagramas de clases son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema, y los componentes que se encargaran del funcionamiento y la relación entre uno y otro (EUMED, 2004).

Los diagramas de clases son útiles para representar:

- Requerimientos en entidades y actuaciones.
- La arquitectura conceptual de un dominio.
- Soluciones de diseño en una arquitectura.
- Componentes de software orientados a objetos.

Para la aplicación HEVAC la Imagen 2.3 muestra el Componente de Validación. La clase principal es **GestorValidacion**[1], el cual distribuye según sea el tipo de elemento a evaluar a las distintas clases **ValidadorXXXHTML**, las cuales son clases hijas del **ValidadorHTML**. Cada clase de validación se ocupa de un elemento y de sus sub-elementos, por ejemplo, la clase **ValidadorFormularioHTML** evalúa todas etiquetas que pertenecen a la etiqueta `<form>` (ésta inclusive), esto incluye los botones, listas, campos de texto, etc.

La clase **ServicioValidacion** es una implementación de un *Servlet* e invoca directamente al **GestorValidacion** para la continuación de su flujo con los parámetros leídos desde la página o el Servicio Web. Esta clase además inicializa las variables de configuración a través de un evento que dispara el servidor cuando se inicia el aplicativo.

En este caso representará el modelo conceptual de la herramienta a desarrollar, la cual se muestra a continuación:

[1] La nomenclatura representa al nombre de una clase.

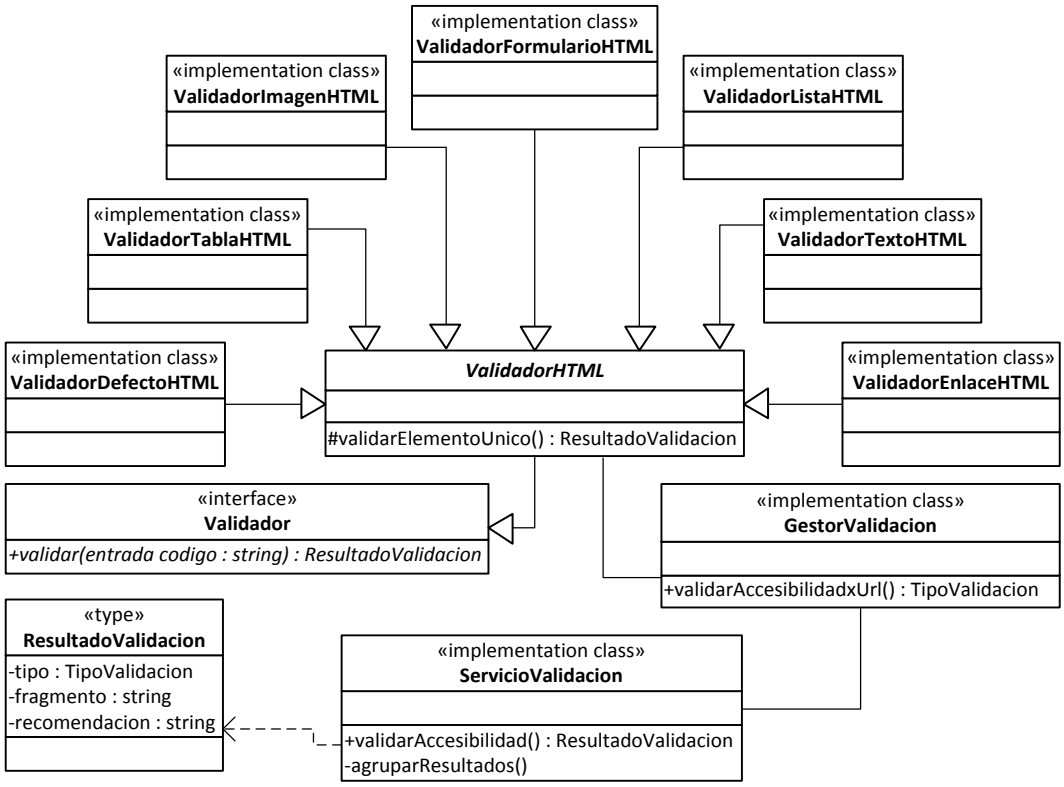


Imagen 2.3: Diagrama de Clases del Sistema HEVAC.
Fuente: Elaboración Propia.

1.4.1. Participantes

Se describe a continuación la lista de participantes que intervienen en la implementación del componente de verificación desplegado en la Imagen 2.3.

- **ServicioValidacion**: Clase que recibe las invocaciones desde los distintos métodos de entrada. Se encarga de recolectar y agrupar los resultados obtenidos para ser desplegados como indique la petición (resultado HTML o en PDF).
- **GestorValidacion**: Clase de contexto que delega la jerarquía de las validaciones una vez que es leído el URL de la página a analizar. Se encarga de recoger las distintas comprobaciones realizadas para calcular los datos finales a ser presentados.
- **Validador**: Declara una interfaz común para todos los algoritmos de comprobaciones.
- **ValidadorHTML**: Clase abstracta que define el proceso de verificación de la tecnología HTML y los componentes involucrados en el proceso.
- **ValidadorXXXHTML**: Conjunto de clases concretas que heredan de la clase anterior para implementar los algoritmos de comprobaciones de los elementos HTML que intervienen en la accesibilidad.
- **ResultadoValidacion**: Objeto de transporte que contiene los datos de una incidencia encontrada.

1.5. Secuencia de Objetos

Es un diagrama que se utiliza para definir rigurosamente la lógica de un escenario de un caso de uso. Un diagrama de secuencia sirve para validar casos de uso durante el diseño con la finalidad de comprender la lógica de su aplicación (EUMED, 2004).

Los diagramas de secuencia muestran los tipos de objetos implicados en el caso de uso, los mensajes que se envían entre si y cualquier valor de retorno asociado con los mensajes. Los objetos (instancias) en UML se muestran subrayados para distinguirlos de las clases.

Las líneas verticales representan la línea de vida de objetos. Las cajas verticales indican si los objetos son activos. Las flechas representan mensajes enviados de una instancia a otra. Un evento ocurre cuando llega un mensaje.

Dado que todos los casos de uso que posee el sistema tienen el mismo flujo de información y misma dirección de la secuencia eventos entre los objetos, se puede representar el sistema con un solo diagrama de secuencia, tal como se muestra a continuación:

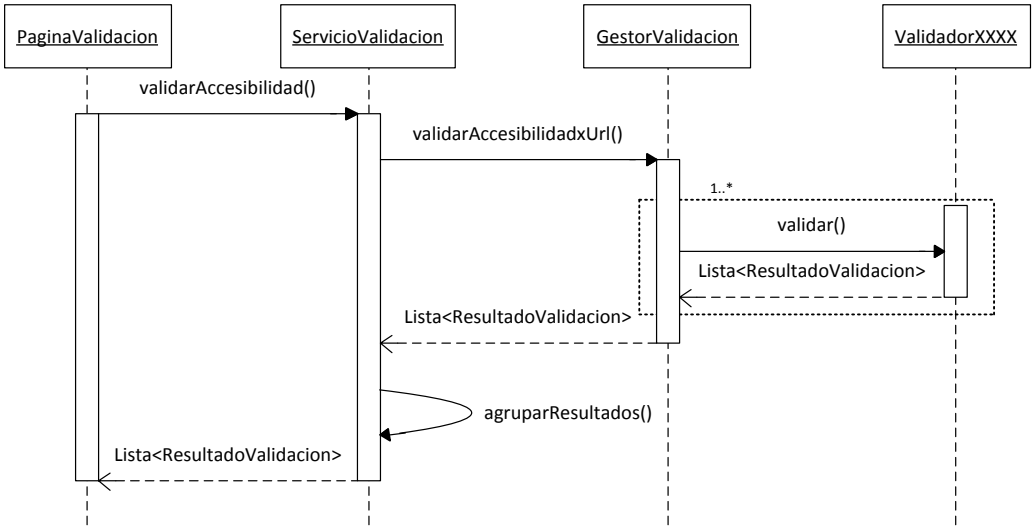


Imagen 2.4: Diagrama de Secuencia del Sistema HEVAC.
Fuente: Elaboración Propia.

La página de validación invoca al método **validarAccesibilidad** vía HTTP, llega al *ServicioValidacion* con los parámetros de la URL y los niveles y principios a evaluar, una vez que se analizan todos estos parámetros se invoca a la clase *GestorValidacion* para que iterativamente dirija cada evaluación de etiqueta HTML al validador correspondiente para que éste indique si la etiqueta posee una o más incidencias de accesibilidad.

1.6. Selección del lenguaje, tecnología y ambiente a implementar

En este punto, la selección del lenguaje es sumamente crítico y objetivo, ya que la herramienta debe ser implementada con un alto nivel de escalabilidad, debido a que el empleo de las verificaciones de accesibilidad irán incrementando según lo hagan los lenguajes de marcado o estándares de accesibilidad. Es por esta razón que el lenguaje a utilizar es necesario que sea orientado a objetos, de manera que puedan usarse las bondades de los patrones de diseño para realizar aplicaciones escalables en base al modelo de estructura de objetos del sistema. El lenguaje que posee más ventajas para realizaciones aplicaciones escalables, con un uso más sencillo de patrones de diseño orientado a objetos y una alta mantenibilidad del código para versiones futuras, es **Java**, el cual será en el que se base el desarrollo de la herramienta. También se evaluaron otros lenguajes para desarrollo Web como PHP o ASP, sin embargo se encontraron dificultades en hallar las librerías necesarias para el análisis del código interno de las páginas Web, donde estás librerías conforman el motor principal de la herramienta.

1.7. Tecnologías de Implementación

Para lograr las funcionalidades propuestas en el alcance de la herramienta, se menciona a continuación las tecnologías y aplicaciones con licencia de software libre para ser utilizados como soporte y que piezas fundamentales en el desarrollo de la herramienta.

Las peticiones HTTP y el despliegue dinámico de las páginas Web de la herramienta serán procesadas por las tecnologías Java *Servlets* y JSP. En el Anexo 4.2 se describe con mayor detalle el funcionamiento de estas tecnologías.

Debido a su simplicidad, completo soporte de los Servlets y páginas JSP, alta escalabilidad y rendimiento y poca o ninguna configuración para su arranque, el servidor de implementación en esta herramienta de desarrollo es el servidor Tomcat. En el Anexo 4.3 se encuentra la información del servidor Apache Tomcat utilizado por la herramienta desarrollada.

1.7.1. Persistencia de Datos

Para cumplir con el objetivo de registrar un historial de verificaciones de los sitios Web analizados por la herramienta, es necesario contar con un sistema manejador de Base de Datos. Al analizar las características más destacadas que debe poseer un manejador de Base de Datos para este tipo de proyectos: eficiente, con alta concurrencia de acceso, multi-plataforma, con licencia de software libre y fácil de instalar y administrar; se tiene que el sistema manejador de Base de Datos elegido para cumplir con este objetivo es **PostgreSQL**.

PostgreSQL está publicado bajo la licencia BSD de software libre y sus ventajas residen en la amplia variedad de tipos de datos nativos y tipos personalizados, los cuales pueden ser indexados. También cuenta con características básicas como claves foráneas, disparadores, vistas, integridad transaccional, herencia de tablas y otras complejas como los tipos de datos en operaciones geométricas y transacciones distribuidas.

La base de datos PostgreSQL fue utilizada para implementar la funcionalidad de registro de análisis históricos, este proceso utiliza una tabla llamada **historico_analisis** y su estructura es la siguiente:

HISTORICO_ANALISIS	
PK **PK**	**url** **fecha_analisis**
	nivel problemas advertencias accesibilidad

Imagen 2.5: Tabla de base de datos para registros históricos.
Fuente: Elaboración Propia

1.8. Selección de la librería de soporte

Para analizar las secciones de código de las páginas HTML, es necesario utilizar alguna herramienta que permita recorrer el código HTML del documento y seleccionar aquellas partes que sean de utilidad. Una de las herramientas que permite hacer lo que se ha explicado anteriormente son los programas denominados *Parsers* HTML. En el Anexo 4.4 se describe con mayor detalle el funcionamiento de los *Parsers* HTML, su estructura y las distintas librerías evaluadas para la utilización en la herramienta.

1.8.1. Características a considerar en el Parser HTML a elegir

Una vez visto las distintas opciones para escoger un *Parser* o analizador de HTML, el siguiente paso es escoger de entre todos ellos uno que permita hacer la tarea de limpiar los documentos de la manera más eficiente posible intentando no sobrecargar la plataforma.

Antes de escoger un *Parser* es necesario definir una serie de características que serían deseables que fuesen cumplidas. Las características son las siguientes:

- Capacidad de leer HTML mal formado.
- Bajo uso de memoria, ya que se analiza una pequeña porción de código.

- Facilidad de uso durante la codificación.
- Extracción de los datos del documento fácilmente.

Observando las características antes definidas que deben cumplir los *Parser* y verificando que las cumplen los *Parsers* anteriores, se observa que la librería **Jericho HTML Parser** cumple todos estos requisitos.

Dado que es un *Parser* específico para HTML es de suponer que la primera característica si la cumpla. La segunda característica también la cumple ya que el uso de memoria puede ser radicalmente menor que un *Parser* basado en árbol como DOM si lo que se necesita analizar son secciones pequeñas de código.

Se han hecho pruebas con documentos HTML sencillos para comprobar el funcionamiento y se comprueba que la librería **Jericho HTML Parser** es muy fácil su uso y bastante intuitivo. También es posible mediante esta librería extraer todos los datos incluidos en el código HTML. Además, se han probado otros *Parsers*, sin embargo son más difíciles de usar que éste o no cumplen las características más esenciales, por lo que el analizador de HTML que se usará en el aplicativo será **Jericho HTML Parser**.

1.8.2. Instalación de la librería de soporte

Como ya se ha descrito anteriormente, el *Parser* HTML a utilizar será Jericho HTML Parser, y antes de empezar el desarrollo de la herramienta Web que verifique los criterios de accesibilidad, es necesario descargar e instalar el *Parser* en el sistema. La última versión disponible es 3.1. Para poder utilizar el *Parser* hay que realizar los siguientes pasos:

1) Descargar el *Parser* del sitio Web del proyecto (Jericho, 2009).
2) Como la descarga anterior es un archivo comprimido es necesario descomprimirlo mediante alguna herramienta que lo permita.
3) Incluir en el programa la **librería jericho-html-3.1.jar** que se encuentra en el directorio /lib del archivo descomprimido.
4) Importar en el proyecto Java el archivo **jericho-html-3.1.jar** para tener acceso a Jericho HTML Parser.

1.8.3. Librerías Adicionales

Para cumplir con el requerimiento funcional que implica la exportación de los resultados en formato PDF, se seleccionó la librería **iText**, el cual es una biblioteca código abierto para crear y manipular archivos PDF, RTF y HTML en Java; está distribuida bajo la licencia *Affero General Public*

License. El mismo documento puede ser exportado en múltiples formatos, o múltiples instancias del mismo formato. Los datos pueden ser escritos a un archivo o, por ejemplo, desde un *Servlet* a un navegador Web (Lowagie, 2012).

1.9. Fase de Desarrollo

Una vez realizado la selección de los módulos esenciales para iniciar el desarrollo de la herramienta, se procede a integrar y organizar los componentes en el ambiente de desarrollo para ir completando los requerimientos propuestos en el inventario de funcionalidades.

1.9.1. Desarrollo por paquetería

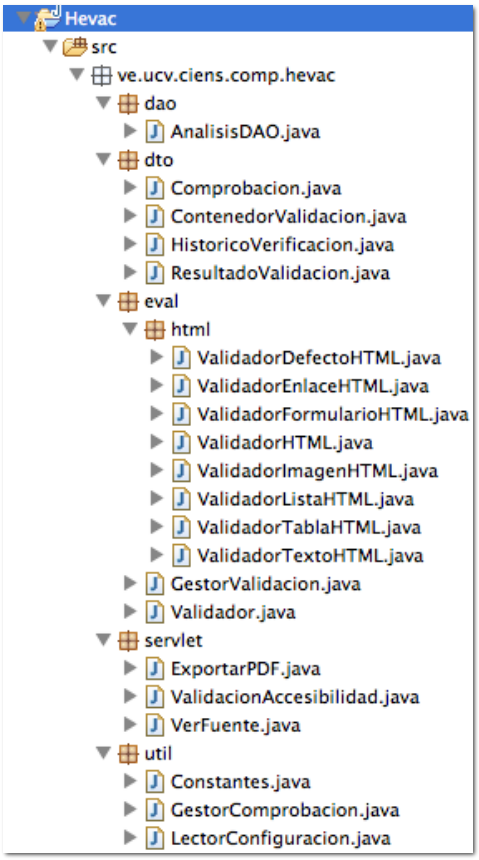
Un Paquete en Java es un contenedor de clases que permite agrupar las distintas partes de un programa cuya funcionalidad tienen elementos comunes. El uso de paquetes proporciona las siguientes ventajas:

- Agrupamiento de clases con características comunes.
- Reutilización de código.
- Mayor seguridad al existir niveles de acceso.

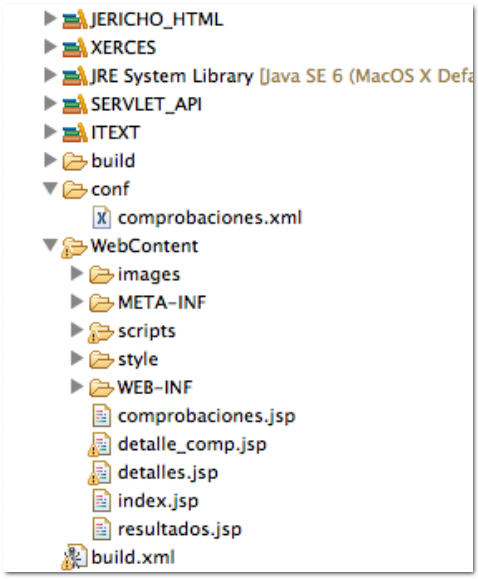
El paquete superior se denomina **ve.ucv.ciens.comp.hevac**. Su nomenclatura proviene de una lectura inversa desde el dominio de red de la institución el cual realiza el desarrollo finalizando en el nombre del aplicativo. A partir de este paquete se encuentra los siguientes sub-paquetes que conforman la separación de funcionalidades y componentes conformados de la siguiente manera:

- **dao**: Posee las clases que realizan acceso a datos internos. En este caso posee acceso a la base de datos que guarda los históricos de las verificaciones.
- **dto**: Posee las clases que almacenan información a transportar (también llamado objetos de valor) entre los componentes de la herramienta usando el patrón *Data Transfer Object*.
- **eval**: Poseen las clases superiores que conforman el componente de las comprobaciones.
- **eval.html**: Poseen las clases base de las comprobaciones de los elementos HTML usando las funcionalidades que provee la librería *Jericho HTML*.
- **servlet**: Posee las clases que forman los *Servlets* de Java para procesar las peticiones de la interfaz Web.
- **util**: Contiene las clases que poseen funcionalidades utilitarias de la herramienta.

La herramienta usa los paquetes para realizar la distinción de las funcionalidades, y se encuentran clasificados como se muestra en la siguiente imagen:



Fuentes de la Herramienta



Librerías y Contenido Web

Imagen 2.6: Vista del Proyecto de la Herramienta Web.
Fuente: Elaboración Propia.

1.9.2. Patrones de Diseño Implementados

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

Los patrones de diseño implementados en la herramienta, usados para solventar el problema con miras a obtener la mayor adaptabilidad para futuras implementaciones, fueron los siguientes:

- Patrón DAO (*Data Access Object*): es un patrón que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de

datos o un archivo. La implementación de este patrón fue utilizado para solventar la gestión histórica de los URL verificados por la herramienta.

- Patrón DTO (*Data Transfer Object*): es un patrón que se usa para transferir datos entre subsistemas de aplicaciones de software. Las clases de DTO que usan este patrón, son usadas a menudo con el patrón DAO para recuperar los datos de una base de datos.

  Se implementa este patrón para trasferir los mensajes en objetos enviados por el gestor de comprobaciones de la herramienta hacia la interfaz Web. También se usa para recuperar y hacer persistente los datos correspondientes al registro histórico de los URL verificados por la herramienta.
- Patrón *Strategy* (Estrategia): En programación orientada a objetos, es un patrón de diseño para el desarrollo de software. Se clasifica como patrón de comportamiento porque determina cómo se debe realizar el intercambio de mensajes entre diferentes objetos para resolver una tarea. El patrón estrategia permite mantener un conjunto de algoritmos de entre los cuales el objeto cliente puede elegir aquel que le conviene e intercambiarlo dinámicamente según sus necesidades.

  Es el patrón más importante implementado en la herramienta. Se usa para distribuir las verificaciones de los elementos HTML en verificaciones más complejas.

  Para los elementos verificables como las imágenes, los formularios, las tablas, entre otros, se crea una clase concreta que realiza las verificaciones de accesibilidad sobre ese elemento mientras una clase abstracta superior que posee un método de estrategia el cual recorre cada uno de los elementos a medida que va iterando por todas las etiquetas HTML de la URL recuperada. Las Imagen 2.3 mostradas anteriormente poseen el detalle visual de la implementación de este patrón.

1.9.3. Técnicas de Suficiencia utilizadas en las comprobaciones

Para asegurarse del cumplimiento de los Criterios de Éxito, se utilizan un conjunto de Técnicas de Suficiencia pertenecientes a estos Criterios, estas Técnicas indican una referencia exitosa para llevar a cabo el cumplimiento de accesibilidad en el elemento HTML evaluado.

En el Anexo 3 se indica la lista de Técnicas utilizadas en la herramienta, que a su vez pueden incrementarse incorporando nuevos algoritmos en las distintas clases de implementación basadas en los elementos HTML. Además, se puede crear una nueva jerarquía para realizar la verificación de otras tecnologías como CSS o *Javascript* utilizando el mismo patrón de diseño *Strategy*.

1.9.4. Principales Interfaces de la Herramienta y Navegabilidad

La interacción con la herramienta es a través de un sitio Web desarrollado con páginas dinámicas JSP, el cual posee los datos de entrada necesarios para realizar las verificaciones de las páginas HTML.

La herramienta posee un esquema de navegación simple (ver Imagen 2.7) con el objetivo de realizar la verificación de una URL en la página principal o mostrar los recursos más accedidos desde la herramienta. La siguiente imagen muestra el detalle de la navegación del Sitio Web.

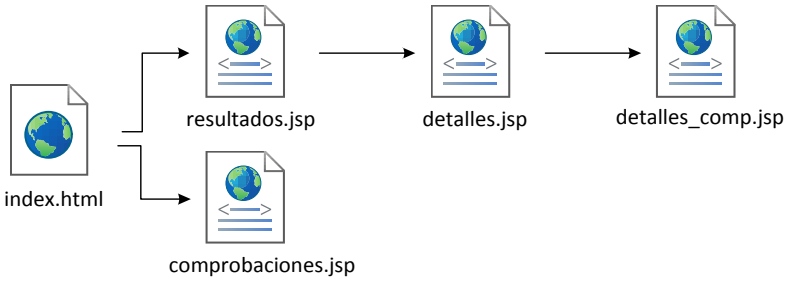


Imagen 2.7: Diagrama de Navegación del Sitio Web.
Fuente: Elaboración Propia.

El sitio Web muestra en su página principal (ver imagen 2.8) un formulario donde se solicita la dirección URL del recurso que desear verificarse, además se solicitan las opciones de comprobación como el nivel y principios de accesibilidad sobre los cuales se tomará en cuenta la verificación del recurso. También posee un enlace en la parte inferior de la página para acceder a la lista de comprobaciones con las que opera la herramienta.

Imagen 2.8: Página principal del Sitio Web.
Fuente: Elaboración Propia.

Al realizar el análisis del recurso Web introducido, se invoca al *Servlet* de verificación el cual al realizar el proceso completo, devuelve un conjunto de datos con los detalles del análisis completado. El Servlet redirige el resultado hacia la página de resultados (ver Imagen 2.9) donde se muestra un breve resumen de la verificación previamente ejecutada.

Imagen 2.9: Página de resultados completados del Sitio Web.
Fuente: Elaboración Propia.

En la imagen anterior se muestra un resumen de los resultados obtenidos por el análisis ejecutado, donde se muestran los tres campos principales que orientan al usuario principal a identificar el problema que posee el sitio Web previamente verificado. Estos campos son los Problemas, las Advertencias y el Grado de Accesibilidad.

Los **Problemas** son todas aquellas verificaciones que no fueron superadas de prioridad uno. Las **Advertencias** son todas aquellas verificaciones que no fueron superadas de prioridad dos o tres, en ambos casos estas verificaciones no cumplen con las recomendaciones expuestas en las Pautas citadas en la WCAG versión 2.0. El **Grado de Accesibilidad** ga se calcula según la cantidad de problemas y advertencias encontradas con respecto a la cantidad de elementos que posea la página HTML y se representa mediante la siguiente ecuación:

$$ga = 1 - \frac{p_1 * 7 \ + \ p_2 * 4 \ + p_3}{e}$$

Imagen 2.10: Ecuación para el cálculo del Grado de Accesibilidad.
Fuente: Elaboración Propia.

Donde p_1, p_2 y p_3 representan las incidencias de las Técnicas prioridades uno (Nivel A), dos (Nivel AA) y tres (Nivel AAA) respectivamente, mientras que e representa la cantidad total de elementos del recurso. Si ga es negativo, se dice que el Grado de Accesibilidad de recurso es **cero**.

El usuario que intenta verificar qué tan accesible puede llegar a ser una página HTML en particular, puede guiarse con los valores que arroje el Grado de Accesibilidad.

Una vez que el usuario decide continuar con la exploración de los resultados arrojados por la herramienta, puede dirigirse a través del enlace inferior a los resultados detallados (ver Imagen 2.11).

Imagen 2.11: Página de resultados completos del análisis del Sitio Web.
Fuente: Elaboración Propia.

Este enlace está orientado a usuarios desarrolladores y/o diseñadores de sitios Web que intentan entregar páginas HTML orientadas a la accesibilidad. Al hacer clic en los títulos de las comprobaciones realizadas en el recurso, se accede a una nueva página con el detalle de la verificación mostrando un consejo de diseño para poder cumplir con la comprobación desplegada. Al ingresar a la página del detalle de la comprobación (Imagen 2.12), se le indica al usuario todas las incidencias correspondientes a dicha comprobación, lo cua lo orienta mostrándole el número de la línea resaltada junto con el código fuente completo.

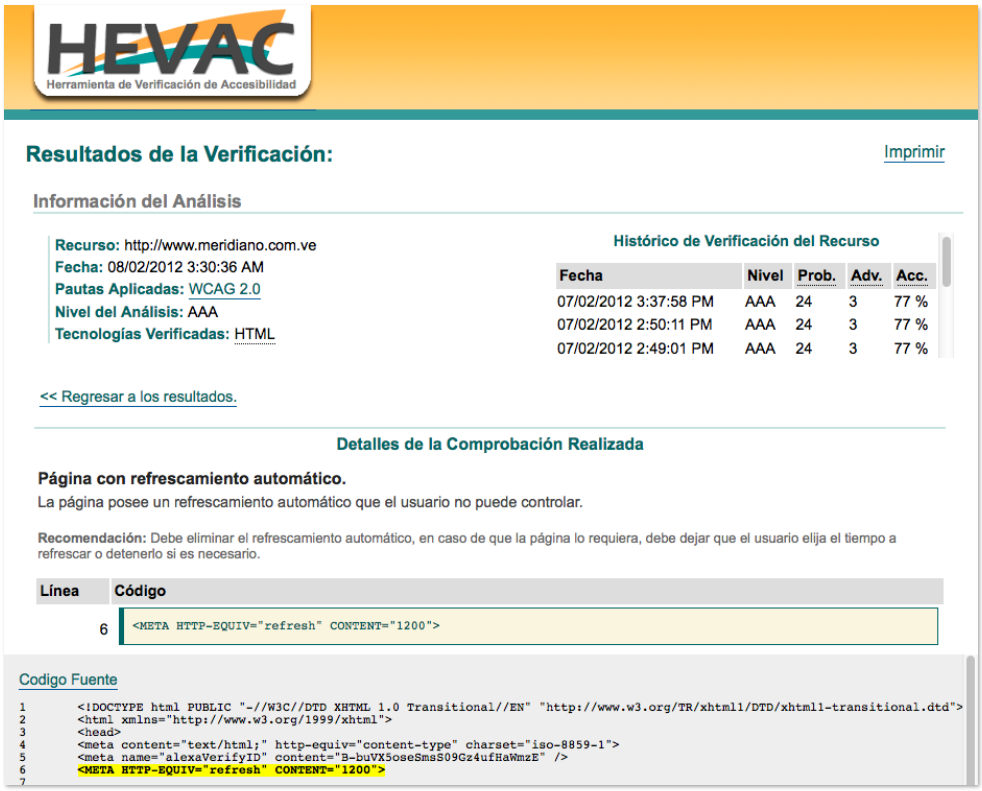


Imagen 2.12: Página de resultados de la comprobación seleccionada en el Sitio Web. Fuente: Elaboración Propia.

2. Consideraciones de Diseño y Desarrollo con otros aplicativos

A continuación se describe el proceso de selección de ideas, patrones de diseño de interfaz y el origen de los principales elementos de las interfaces de la herramienta HEVAC.

Las interfaces de usuario de la herramienta se diseñaron integrando ideas de los estilos visuales de la aplicación en línea Test de Accesibilidad Web (TAW) (CTIC, 2009), sin reutilizar el código interno de esa aplicación, cuya página de inicio se muestra en la Imagen 3.1.1.

Imagen 3.1.1: Página principal de TAW.
Fuente: Tawdis.net

HEVAC presenta a su vez un conjunto de variaciones en el diseño en sus secciones (Imagen 3.1.2) las cuales se describirán a continuación y han sido destacadas a través de los siguientes recuadros:

- **Recuadro Rojo:** Indica que se incorporaron ideas del diseño del sitio Web de TAW.
- **Recuadro Verde:** Indica que en la sección marcada consideró parcialmente el estilo visual del sitio Web de TAW y se realizó un conjunto de modificaciones para adaptarlas a los intereses de la herramienta HEVAC.
- **Recuadro Violeta:** Indica que en la sección marcada se utilizó un estilo visual propio, no considerando ideas de la aplicación TAW. Se corresponde con una nueva sección o presenta información que TAW no presenta.

En la pantalla principal se incluyeron variantes presentadas en la siguiente imagen:

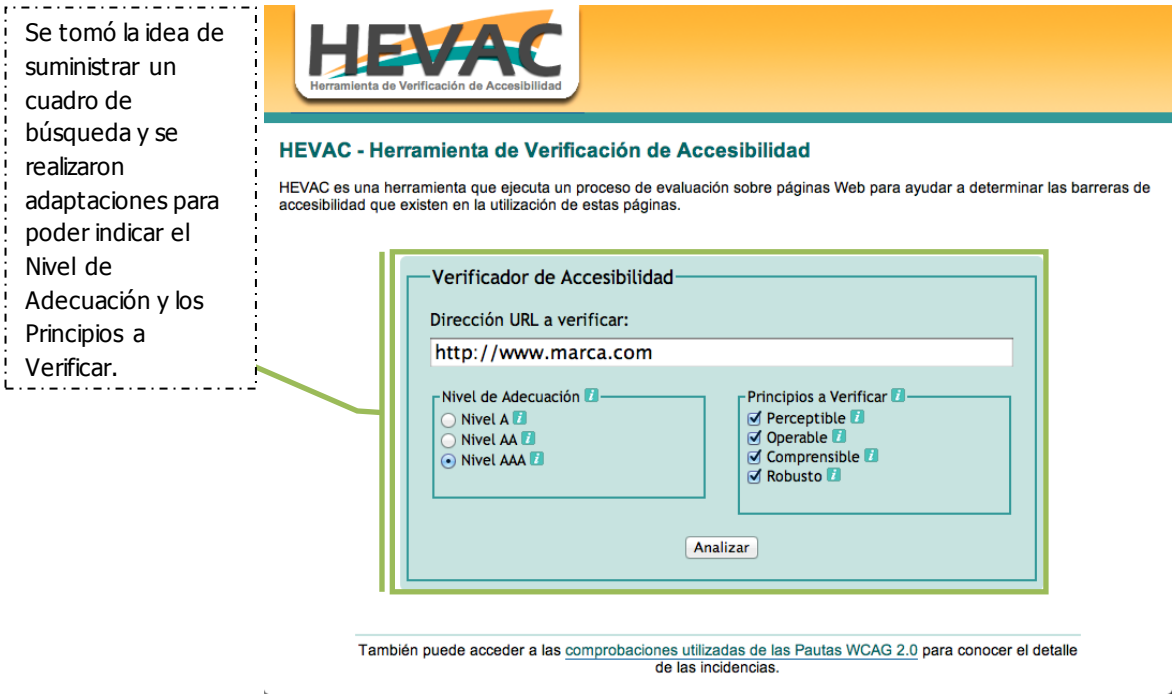


Imagen 3.1.2: Consideraciones en la Página principal de HEVAC.
Fuente: Elaboración Propia.

Una vez que es analizado el URL se pasa a la siguiente pantalla que contiene el primer nivel de resultados, destacando al igual que en TAW una sección de resumen en la parte superior izquierda.

Imagen 3.1.3: Consideraciones en la Página del primer nivel de resultados.
Fuente: Elaboración Propia.

Al realizar clic en el enlace que accede al análisis detallado, se presenta la siguiente página con el detalle de las incidencias organizada por principio verificado.

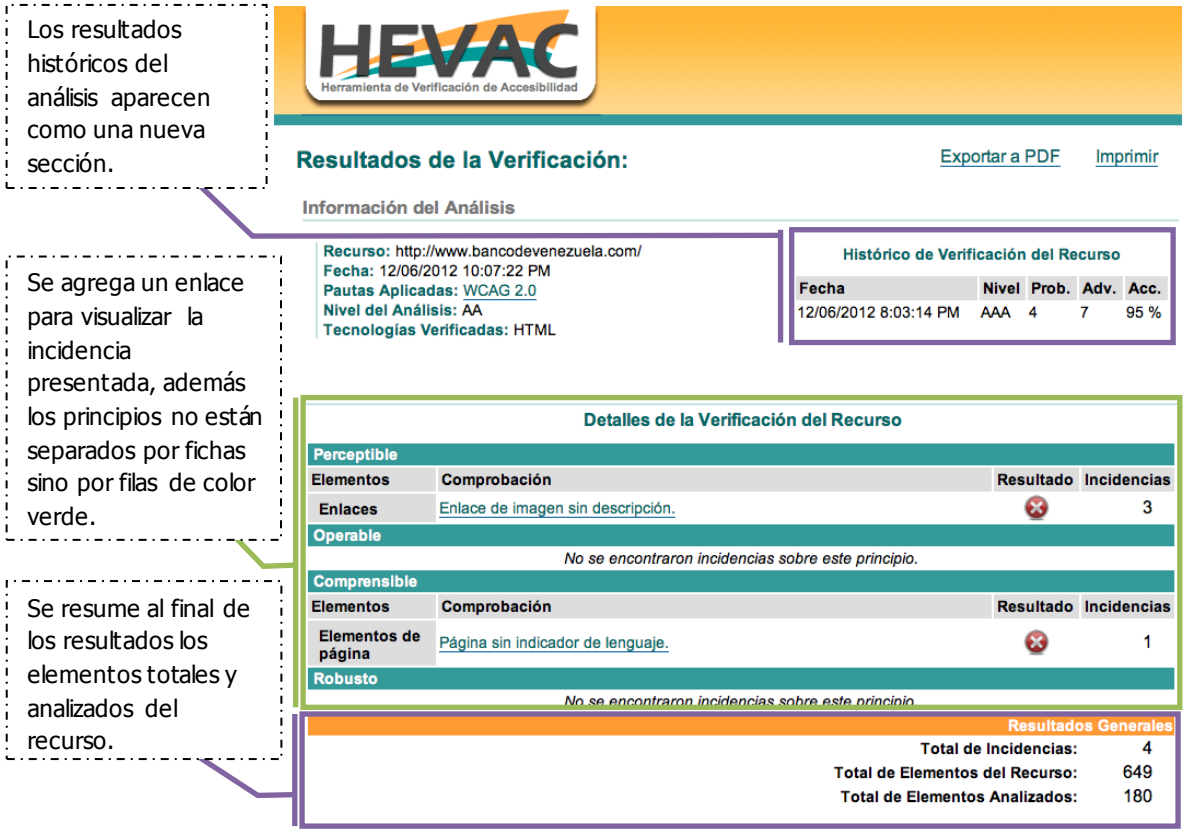


Imagen 3.1.4: Consideraciones en la Página del detalle de la verificación.
Fuente: Elaboración Propia.

Una vez que el usuario accede al detalle de las incidencias presentadas, se muestra una página con el código fuente y las líneas con problemas. El diseño, despliegue de información y navegación es propio de la herramienta HEVAC.

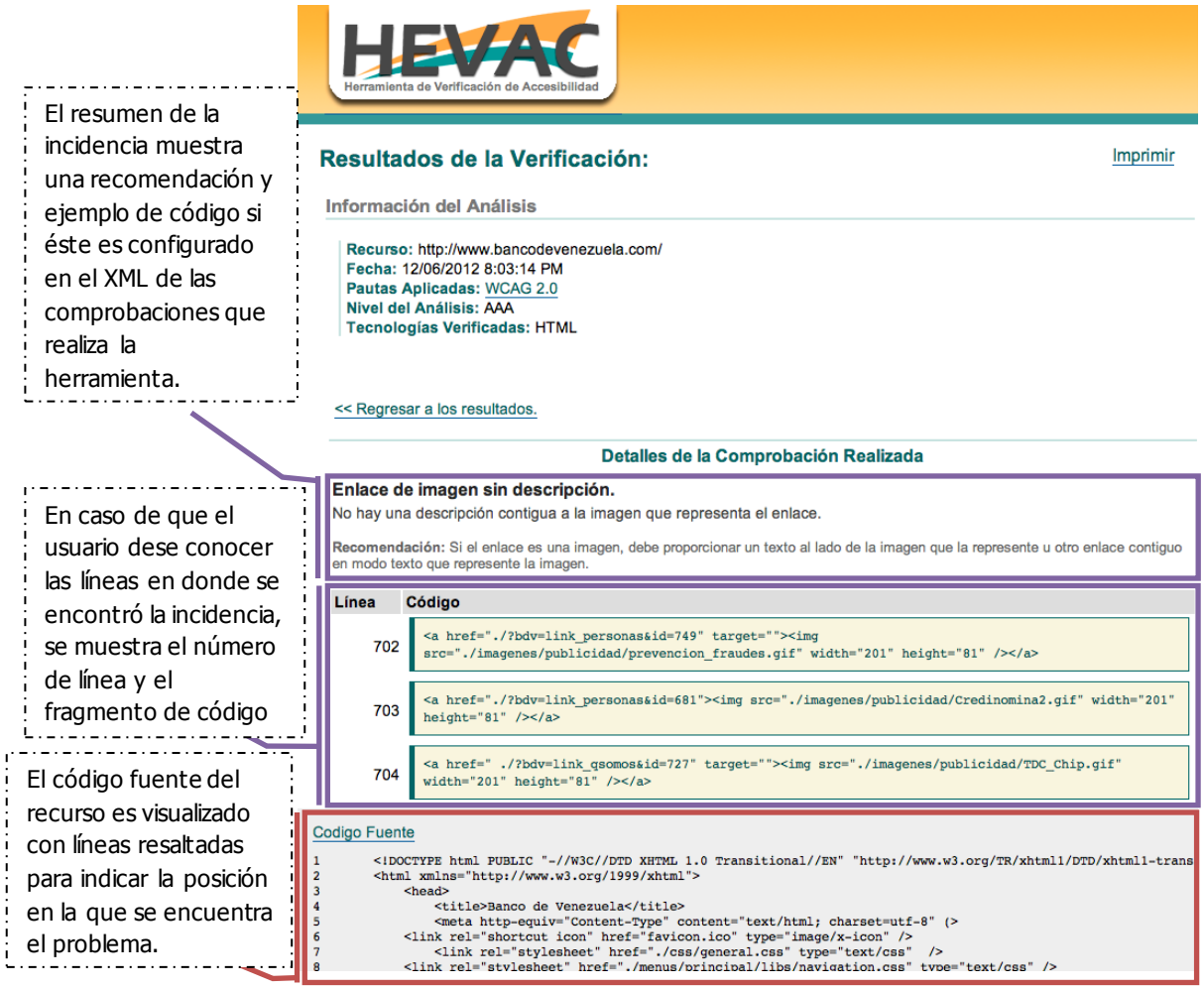


Imagen 3.1.5: Consideraciones en la Página del detalle de la incidencia seleccionada.
Fuente: Elaboración Propia.

Se destaca, que tanto las herramientas HEVAC como TAW tienen elementos en común en su interfaz que responden al uso de metáforas, iconografía y patrones de diseño de interfaz comúnmente aceptados e incluso considerados buenas prácticas, entre los que se encuentran:

- Cajas de búsqueda (*search box*).
- Enlace a la página principal (*home link*).
- Despliegue de información extra bajo demanda (*extras on demand*).
- Información contenida en Paneles desplegables (*closable panels*).
- Información detallada (*overview by detail*).
- Sección de pie de página (*footer bar*).

En el aspecto del desarrollo del motor de evaluación de accesibilidad, la herramienta HEVAC posee un conjunto de diferencias con respecto al aplicativo TAW, las cuales son mostradas en la siguiente tabla:

Características	**HEVAC**	**TAW**
Pautas	WCAG 2.0	WCAG 1.0, WCAG 2.0, mobileOK[2]
Tecnologías Soportadas	HTML	HTML, CCS, Javascript (parcialmente)
Técnicas HTML	Parcial (Basado en criterios de discapacidad visual)	Completa
Selección del Nivel de Análisis	Si	Si
Selección de Principios de Accesibilidad (sólo WCAG 2.0)	Si	No
Clasificación de comprobaciones	Por nomenclatura de técnicas (ver Anexo 3)	Por numeración de criterios de éxitos.
Indicación de comprobaciones no realizadas	No	Si
Agrupación de resultados por Principios	Si	Si
Consejos de desarrollo	Si	No (Posee un enlace a la página de la técnica en el sitio del WCAG)
Indicación del Grado de Accesibilidad	Si	No
Pre-visualización de la página evaluada	No	Si
Indicación del problema en el código fuente	Si	Si
Visualización en formato PDF	Si	No
Registró histórico de verificaciones	Si	No

Tabla 2.6: Diferencias entre HEVAC y TAW.
Fuente: Elaboración Propia.

La esencia las características de HEVAC está ligada a la simplicidad en la presentación de los resultados obtenidos, sin embargo, el desarrollo de la herramienta en sus módulos bases están adaptados para soportar una ampliación en el conjunto de características.

3. Proceso de Pruebas

En este proceso se ejecutan dos tipos de pruebas: las pruebas realizadas a la herramienta, para obtener los requerimientos de Software alcanzados a través de los Casos de Uso planteados,

[2] Es un analizador de buenas prácticas para web móvil. En concreto comprueba la adecuación de un contenido móvil a nivel *mobileOK Basic* según se define en *W3C mobileOK Basic Tests 1.0* que está basado en *W3C Mobile Web Best Practices 1.0*.

además del cumplimiento de usabilidad y accesibilidad; y las pruebas realizadas a un conjunto de sitios Web con el fin de obtener los resultados de accesibilidad arrojados por la herramienta.

3.1. Pruebas sobre la Herramienta HEVAC

Las pruebas realizadas a la herramienta HEVAC intentan corroborar el cumplimiento de un conjunto aspectos fundamentales para la entrega y utilización de esta herramienta por parte de los usuarios finales. Las pruebas ejecutadas fueron: Las pruebas de funcionalidad, las pruebas de usabilidad y las pruebas de accesibilidad.

3.1.1. Pruebas de Funcionalidad

Las pruebas de funcionalidad tienen como objetivo asegurar el trabajo apropiado de los requisitos funcionales (Casos de Uso), los cuales incluyen la navegación, entrada de datos, procesamiento y obtención de resultados. Estas pruebas realizadas a la herramienta fueron llevadas a cabo a través de pruebas unitarias sobre cada uno de los Casos de Uso planteados. El detalle de las pruebas se presenta a continuación basado en cada Caso de Uso.

3.1.1.1. Caso de Uso 1

El Caso de Uso 1 indica el siguiente requerimiento: "Verificar Página Web por Nivel de Conformidad". Para realizar la prueba unitaria se selecciona una Técnica de Suficiencia perteneciente a un Criterio de Éxito de Nivel AAA, la Técnica que se toma para este caso será **Área de Imagen sin título** (ver Anexo 3). Se crea un código HTML de ejemplo que incumpla esta técnica, el cual se presenta a continuación:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html lang="es">
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
<title>Prueba Unitaria - Caso de Uso</title>
</head>
<body>
  <img src="images/accesibility.png" usemap="#map1"  alt="Simbolo de
accesibilidad"/>
  <map id="map1" name="map1">
    <area shape="rect" coords="0,0,30,30"
      href="referencia.html" alt="Referencia" />
    <area shape="rect" coords="34,34,100,100" href="medios.html"/>
  </map>
</body>
</html>
```

En la línea sombreada con color Amarillo, se crea intencionalmente la incidencia ya que la etiqueta `<area>` no presenta el atributo `alt` como lo contiene la etiqueta anterior. Este código se lleva a un archivo HTML de prueba que luego es colocado en la raíz del servidor HEVAC para que

pueda ser accedido por URL y a su vez analizado por la misma herramienta (ver imagen 3.2.1) seleccionando el Nivel AAA y todos los Principios presentados. Los resultados son los siguientes:

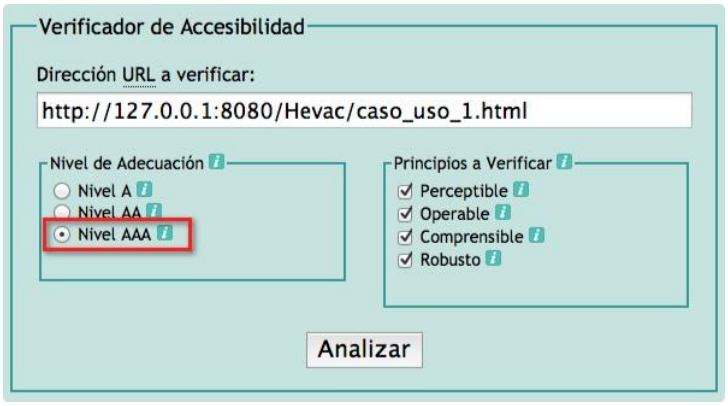


Imagen 3.2.1: Paso 1 para las pruebas del Caso de Uso 1.
Fuente: Elaboración Propia.

Al analizar los resultados del archivo de pruebas se puede observar que se encuentra la incidencia.

Imagen 3.2.2: Paso 2 para las pruebas del Caso de Uso 1.
Fuente: Elaboración Propia.

Al acceder al detalle de la incidencia se puede observar la línea que contiene el código con el problema de Nivel AAA encontrado, igual al código colocado anteriormente.

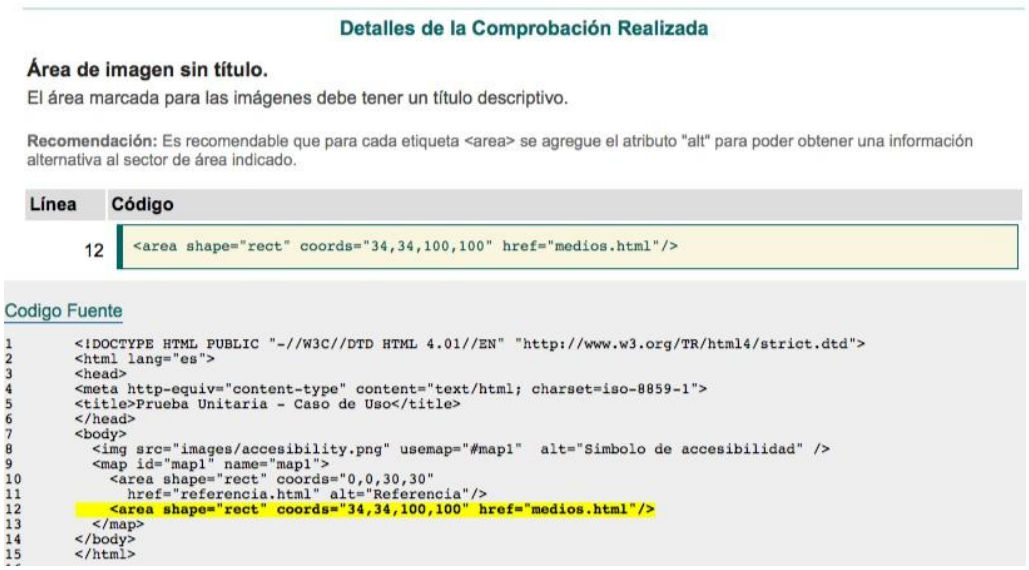


Imagen 3.2.3: Paso 3 para las pruebas del Caso de Uso 1.
Fuente: Elaboración Propia.

Luego se procede a probar con un Nivel de conformidad inferior, en este caso se selecciona Nivel AA y todos los Principios presentados.

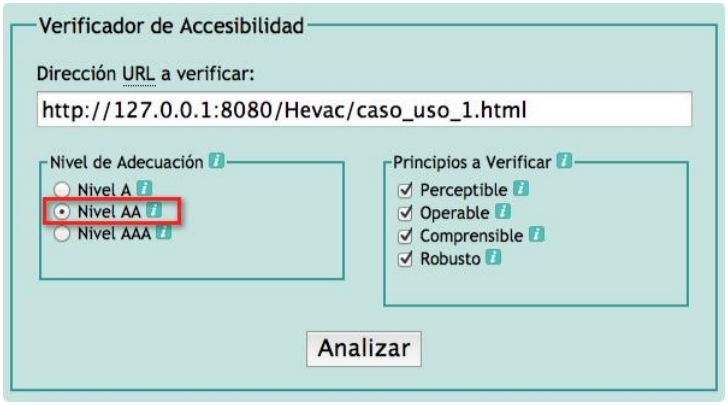


Imagen 3.2.4: Paso 4 para las pruebas del Caso de Uso 1.
Fuente: Elaboración Propia.

Una vez analizado el mismo recurso con un Nivel menor, en este caso el Nivel AA, se observa que no se encontraron incidencias:

Imagen 3.2.5: Paso 5 para las pruebas del Caso de Uso 1.
Fuente: Elaboración Propia.

Con las evidencias presentadas se puede observar que se cumple el Caso de Uso 1: "Verificar Página Web por Nivel de Conformidad", distinguiendo las incidencias según su Nivel de Conformidad.

3.1.1.2. Caso de Uso 2

El Caso de Uso 2 indica el siguiente requerimiento: "Verificar Página Web por Principio de Accesibilidad". Para realizar la prueba unitaria se selecciona la misma Técnica utilizada en la prueba del Caso de Uso 1, la Técnica corresponde a **Área de Imagen sin título** y esta pertenece al Principio de Accesibilidad **Operable,** se agrega además una nueva incidencia de otro Principio, esta corresponde a **Página sin indicador de lenguaje**, que pertenece al principio **Comprensible**. El código HTML de ejemplo que incumple con estas Técnicas es similar al anterior:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
<title>Prueba Unitaria - Caso de Uso</title>
</head>
<body>
  <img src="images/accesibility.png" usemap="#map1"  alt="Simbolo de
accesibilidad"/>
  <map id="map1" name="map1">
    <area shape="rect" coords="0,0,30,30" href="referencia.html"/>
    <area shape="rect" coords="34,34,100,100" href="medios.html" alt="Medios
Alternativos"/>
  </map>
</body>
</html>
```

Se mantiene la etiqueta que presenta la incidencia pero en este caso el problema lo posee la primera etiqueta `<area>` con la ausencia del atributo `alt`, además se elimina el atributo `lang`

de la etiqueta `<html>` para obtener la segunda incidencia. Este código se lleva a un archivo HTML de prueba que luego es colocado en la raíz del servidor HEVAC para que pueda ser accedido por URL y a su vez analizado por la misma herramienta (ver imagen 3.3.1), con un nombre diferente seleccionando el Nivel AAA y todos los Principios presentados.

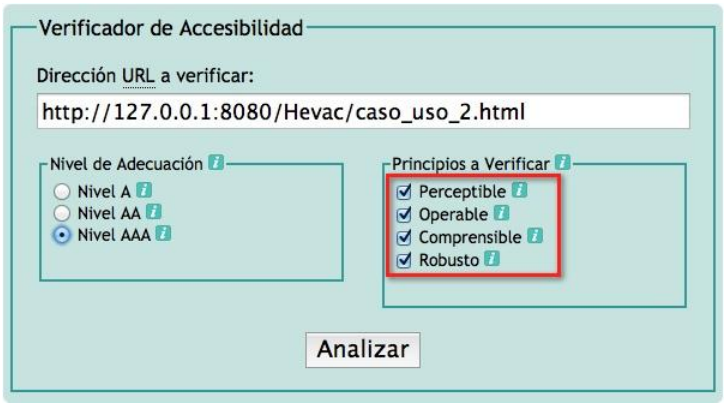


Imagen 3.3.1: Paso 1 para las pruebas del Caso de Uso 2.
Fuente: Elaboración Propia.

Una vez analizado el nuevo recurso HTML creado, se obtienen las dos incidencias que fueron colocadas como pruebas, las cuales se muestran en la siguiente imagen:

Imagen 3.3.2: Paso 2 para las pruebas del Caso de Uso 2.
Fuente: Elaboración Propia.

Al acceder a la incidencia se puede mostrar el código con el problema en la etiqueta `<html>`.

Imagen 3.3.3: Paso 3 para las pruebas del Caso de Uso 2.
Fuente: Elaboración Propia.

Ahora se intenta realizar la prueba deseleccionando la opción del principio Operable, para que sólo tome en cuenta una incidencia de las dos presentadas manteniendo el Nivel AAA.

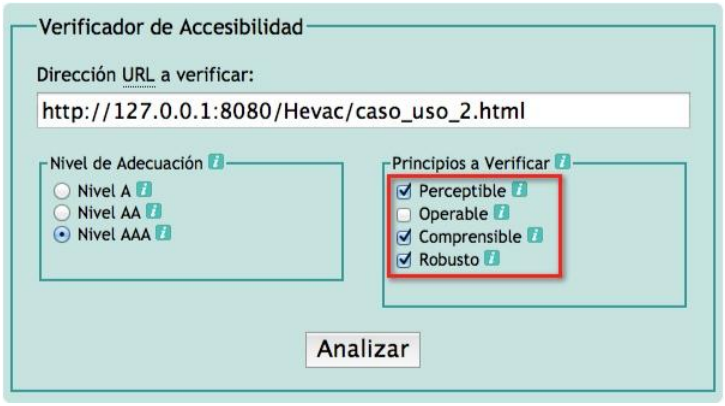
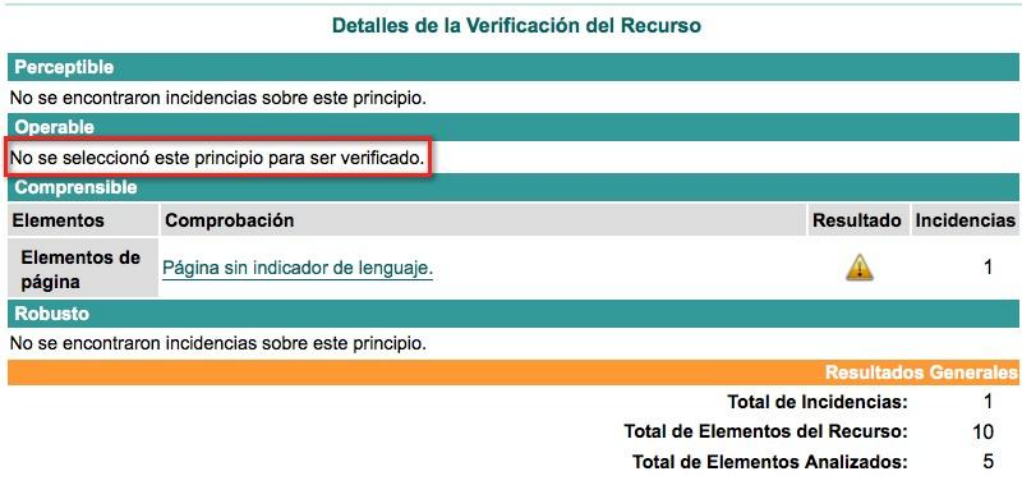


Imagen 3.3.4: Paso 4 para las pruebas del Caso de Uso 2.
Fuente: Elaboración Propia.

Al realizar el análisis se observa que no toma en cuenta el principio Operable para ser evaluado, mostrando el mensaje que aparece a continuación en la siguiente imagen:



Detalles de la Verificación del Recurso

Perceptible

No se encontraron incidencias sobre este principio.

Operable

No se seleccionó este principio para ser verificado.

Comprensible

Elementos	Comprobación	Resultado	Incidencias
Elementos de página	Página sin indicador de lenguaje.	⚠	1

Robusto

No se encontraron incidencias sobre este principio.

Resultados Generales

Total de Incidencias:	1
Total de Elementos del Recurso:	10
Total de Elementos Analizados:	5

Imagen 3.3.5: Paso 5 para las pruebas del Caso de Uso 2.
Fuente: Elaboración Propia.

Con las evidencias presentadas se puede observar que se cumple el Caso de Uso 2: "Verificar Página Web por Principio de Accesibilidad", filtrando las verificaciones de las incidencias según su Principio de Accesibilidad.

3.1.1.3. Caso de Uso 3

El Caso de Uso 3 indica el siguiente requerimiento: "Exportar resultados en formato HTML". El requerimiento se presenta como un flujo de inclusión donde se accede a través de cada verificación, mostrando una página HTML generada con el resumen de las incidencias encontradas, la lista de incidencias ordenadas por Principio de Accesibilidad y el detalle de cada incidencia seleccionada mostrando el código fuente; todas estas páginas son creadas dinámicamente por los *Servlets* y páginas JSP a través de las resultados obtenidos, generando un código HTML final el cual procesado y desplegado por los navegadores Web. Por esta razón, se considera que las pruebas unitarias de los Casos de Uso 1 y 2 comprueban el correcto funcionamiento del presente Caso de Uso.

3.1.1.4. Caso de Uso 4

El Caso de Uso 4 indica el siguiente requerimiento: "Exportar resultados en formato PDF". Para realizar la siguiente prueba se puede utilizar el código de las pruebas unitarias del Caso de Uso 2, el cual contiene las incidencias de las Técnicas **Área de Imagen sin título** y **Página sin indicador de lenguaje.**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
<title>Prueba Unitaria - Caso de Uso</title>
```

```
</head>
<body>
  <img src="images/accesibility.png" usemap="#map1" alt="Simbolo de
accesibilidad"/>
  <map id="map1" name="map1">
    <area shape="rect" coords="0,0,30,30" href="referencia.html"/>
    <area shape="rect" coords="34,34,100,100" href="medios.html" alt="Medios
Alternativos"/>
  </map>
</body>
</html>
```

El código sombreado en Amarillo representa las incidencias de accesibilidad colocadas. Este código se lleva de igual manera a un archivo HTML de prueba que luego es colocado en la raíz del servidor HEVAC para que pueda ser accedido por URL y a su vez analizado por la misma herramienta (ver imagen 3.4.1), con un nombre diferente seleccionando el Nivel AAA y todos los Principios presentados.

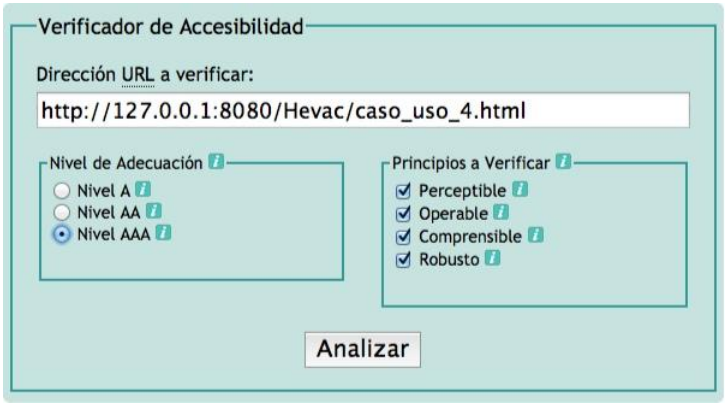


Imagen 3.4.1: Paso 1 para las pruebas del Caso de Uso 4.
Fuente: Elaboración Propia.

Al realizar el análisis del recurso se muestra el resumen con los resultados obtenidos como se muestra en la imagen 3.4.2.

Imagen 3.4.2: Paso 2 para las pruebas del Caso de Uso 4.
Fuente: Elaboración Propia.

Una vez analizado los resultados y accediendo a la vista detallada, aparece la opción "Exportar a PDF" en la parte superior izquierda como se muestra en la siguiente imagen.

Imagen 3.4.3: Paso 3 para las pruebas del Caso de Uso 4.
Fuente: Elaboración Propia.

Al presionar el enlace "Exportar a PDF" aparece un cuadro de diálogo del navegador indicando el archivo PDF a descargar como aparece en la imagen 3.4.4.

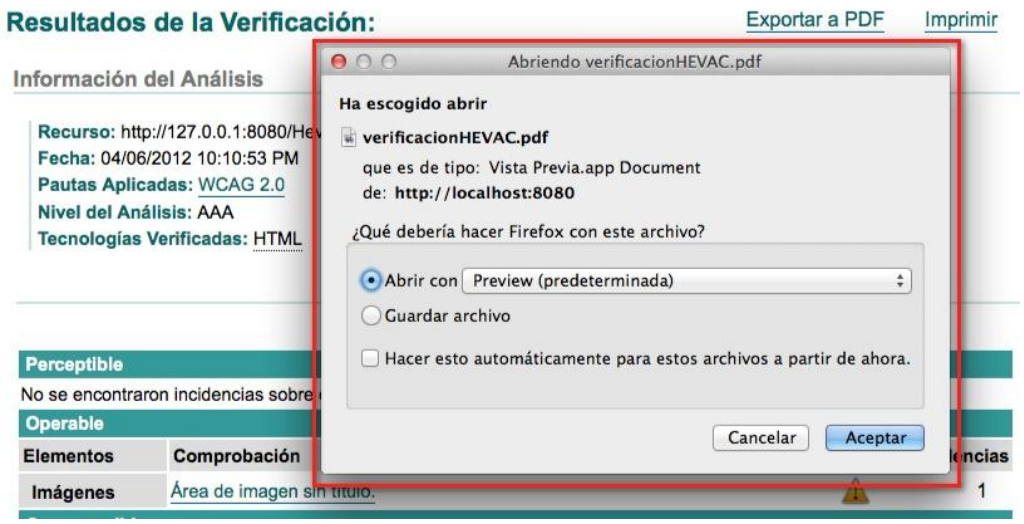


Imagen 3.4.4: Paso 4 para las pruebas del Caso de Uso 4.
Fuente: Elaboración Propia.

Al descargar y abrir el PDF se muestran los resultados que se obtienen con la vista HTML.

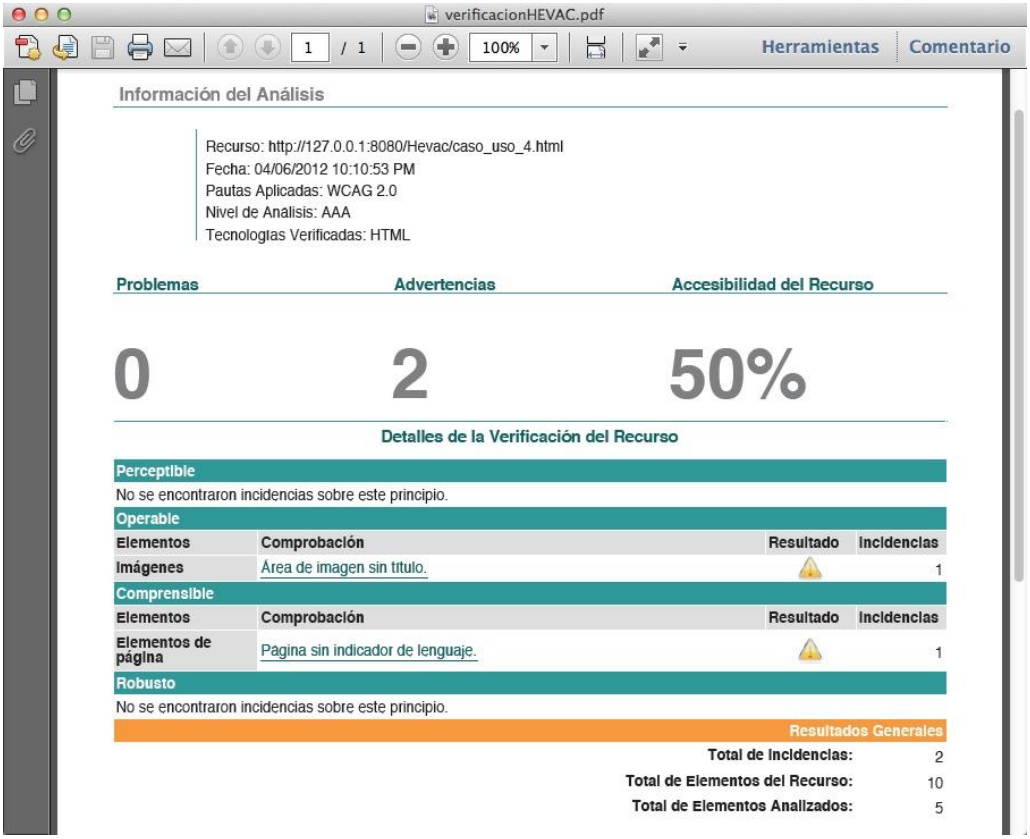


Imagen 3.4.5: Paso 5 para las pruebas del Caso de Uso 4.
Fuente: Elaboración Propia.

Presentando las evidencias anteriores, se puede observar que se cumple el requerimiento que representa el Caso de Uso 4: "Exportar resultados en formato PDF".

3.1.1.5. Caso de Uso 5

El Caso de Uso 5 indica el siguiente requerimiento: "Visualizar los resultados de los análisis históricos". Para realizar la siguiente prueba se puede utilizar el mismo archivo del Caso de Uso 4, de esta manera al realizar una nueva verificación sobre el mismo recurso generará un nuevo registro en el historial que podrá ser visualizado. Para generar una leve diferencia en los resultados, se procede a verificar con el Nivel AA y todos los principios seleccionados.

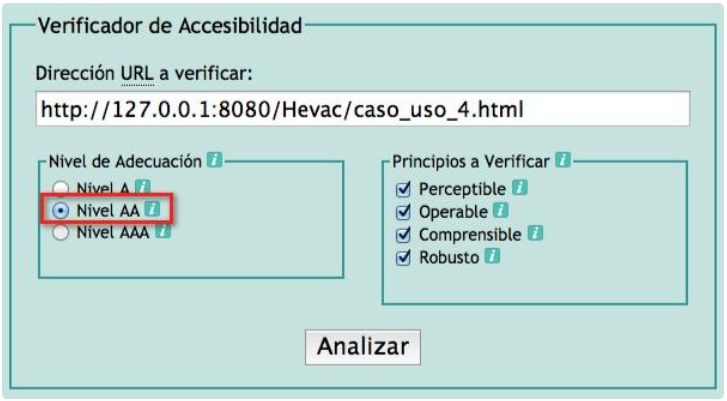


Imagen 3.5.1: Paso 1 para las pruebas del Caso de Uso 5.
Fuente: Elaboración Propia.

Luego de ejecutar el análisis se obtienen distintos los cuales corresponden a las pruebas unitarias del Caso de Uso 4, esto se debe a que el Nivel AA no incluye la comprobación **Área de Imagen sin título**, presentando además una nueva fecha que se muestra a continuación:

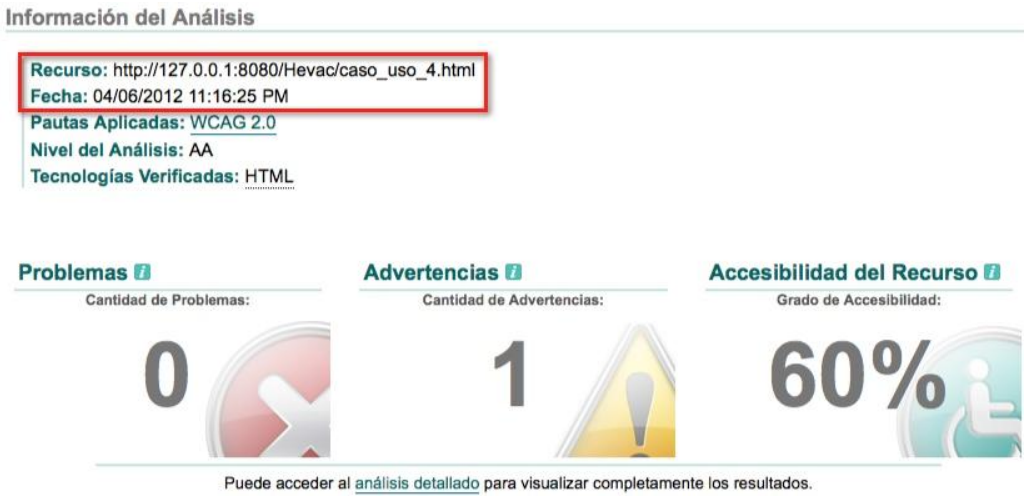


Imagen 3.5.2: Paso 2 para las pruebas del Caso de Uso 5.
Fuente: Elaboración Propia.

Una vez accedido a la vista detallada se puede observar que existe una nueva sección que indica las verificaciones históricas, esto sucede porque se ha realizado la verificación en más de una

ocasión al mismo recurso, en este caso arroja la fecha y los resultados de las pruebas del Caso de Uso 4 (ver imagen 3.5.3).

Imagen 3.5.3: Paso 3 para las pruebas del Caso de Uso 5.
Fuente: Elaboración Propia.

Observadas las evidencias presentadas, se puede observar que se comprueba el correcto funcionamiento en el requerimiento del Caso de Uso 5: "Visualizar los resultados de los análisis históricos", llevando a cabo el cumplimiento de todos los Casos de Uso planteados como requerimientos de Software para el desarrollo de la herramienta.

3.1.2. Pruebas de Usabilidad

Las Pruebas de Usabilidad consisten en seleccionar a un grupo de usuarios de una aplicación y solicitarles que lleven a cabo las tareas para las cuales fue diseñada, mientras se toma nota de la interacción, particularmente de los errores y dificultades con las que se encuentren los usuarios. Para ello es necesario crear sesiones de pruebas con los usuarios en un lugar especial para probar la aplicación.

Para efectos de la herramienta desarrollada, las Pruebas de Usabilidad no fueron alcanzadas por la dificultad de reunir a los distintos tipos de usuarios finales y crear las sesiones necesarias para llevar a cabo las tareas adaptadas a las necesidades de cada tipo de usuario.

3.1.3. Pruebas de Accesibilidad

Las Pruebas de Accesibilidad realizadas a la herramienta fueron llevadas a cabo a través de un aplicativo Web llamado *Web Accesibility Evaluation Tool* (WAVE) realizado por la comunidad libre WebAIM, se encuentra en línea desde el año 2001, su versión actual es la 4.0 y ha sido usada para evaluar la accesibilidad de millones de documentos Web (WebAIM, 2001). Se seleccionó esta aplicación motivado a una funcionalidad que permite cargar los archivos HTML directamente al evaluador de accesibilidad, esto facilita el proceso de pruebas ya que no es necesario mantener la herramienta a verificar desplegada en un dominio público.

Para completar el proceso de evaluación, se guardaron todas las páginas que genera la herramienta (estáticas o dinámicas) como archivos HTML en el disco local y fueron subidas a través del cargador de archivos del aplicativo WAVE. Los resultados son los siguientes:

- **Página Inicial**: Es la página que solicita los datos de entrada (ver Imagen 2.8). Se verificó en el aplicativo y no presenta errores de accesibilidad. Los detalles se muestran en la siguiente imagen:

Imagen 4.1: Prueba de Accesibilidad a la página inicial de HEVAC.
Fuente: Elaboración Propia.

- **Página de resultados resumidos**: Es la página que muestra los resultados de las incidencias (ver Imagen 2.9) .Se verificó en el aplicativo y no presenta errores de accesibilidad. Los detalles se muestran en la siguiente imagen:

Imagen 4.2: Prueba de Accesibilidad a la página de resultados resumidos de HEVAC.
Fuente: Elaboración Propia.

- **Página de resultados detallados**: Es la página que contiene el listado de las comprobaciones con incidencias ordenadas por los Principios de Accesibilidad (ver Imagen 2.11). Se verificó en el aplicativo y no presenta errores de accesibilidad. Los detalles se muestran en la siguiente imagen:

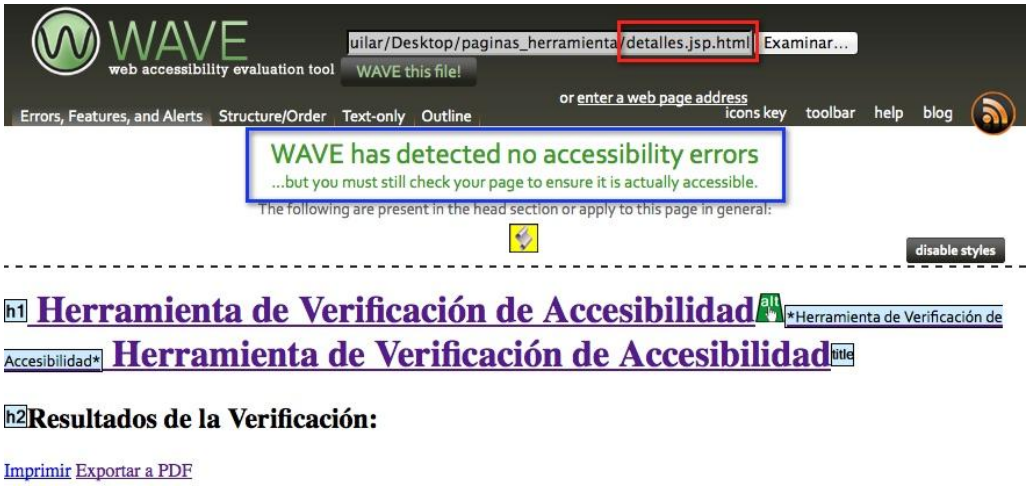


Imagen 4.3: Prueba de Accesibilidad a la página de resultados detallados de HEVAC.
Fuente: Elaboración Propia.

- **Página del detalle de la comprobación en las incidencias**: Es la página que contiene el detalle del problema presentado en el código HTML dado una comprobación seleccionada (ver Imagen 2.12). Se verificó en el aplicativo y no presenta errores de accesibilidad. Los detalles se muestran en la siguiente imagen:

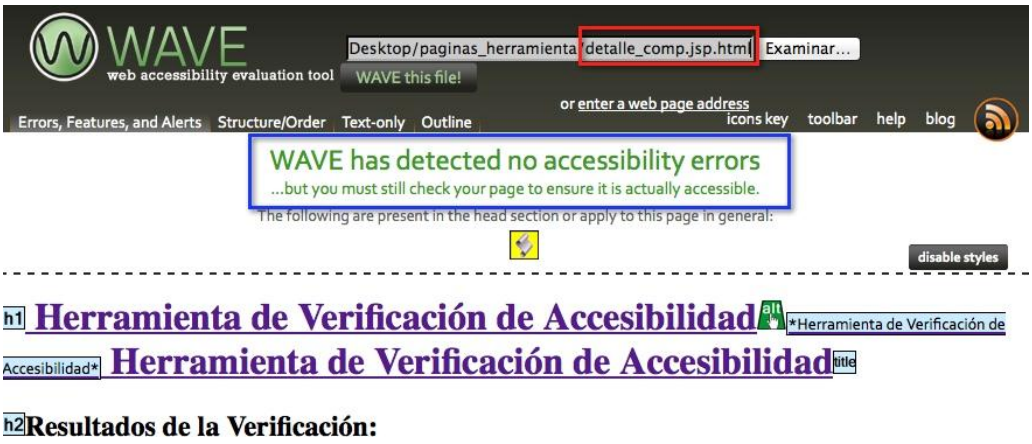


Imagen 4.4: Prueba de Accesibilidad a la página de los detalles de comprobación de HEVAC. Fuente: Elaboración Propia.

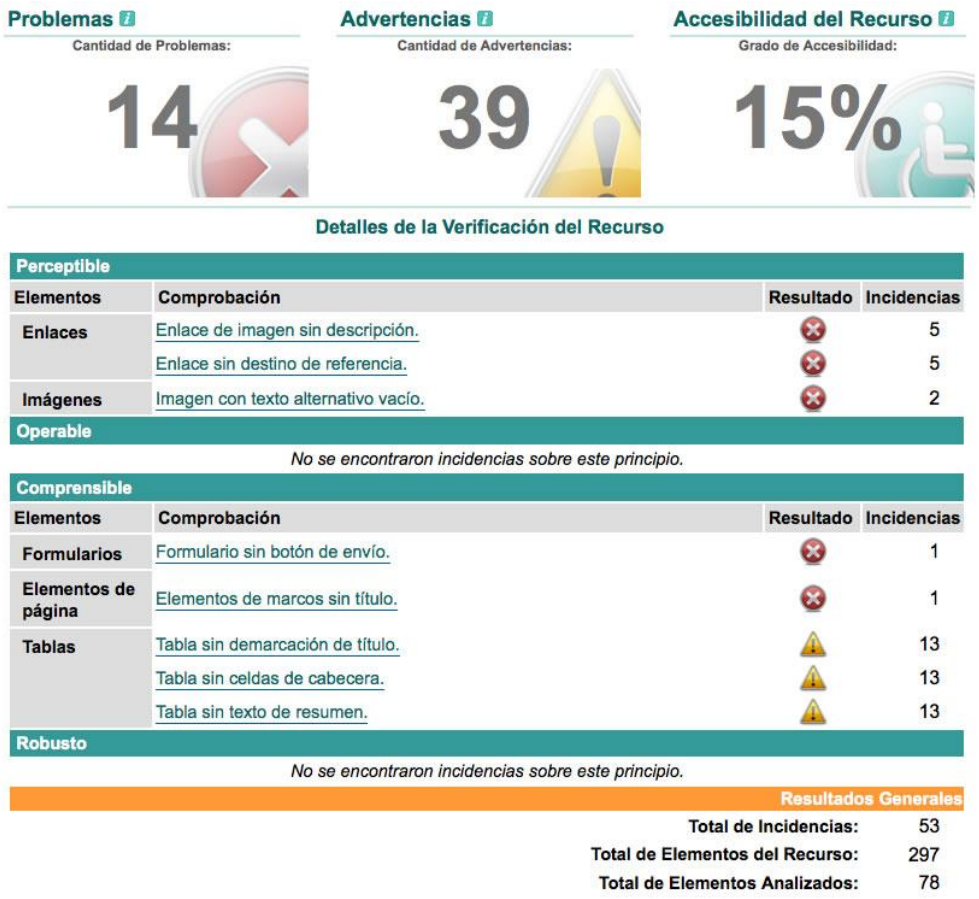
Se observa en cada una de las pruebas realizadas con el aplicativo WAVE que ninguna de las páginas evaluadas presenta errores de accesibilidad, por lo que se cumplen los requisitos de accesibilidad para la herramienta HEVAC.

3.2. Pruebas realizadas en Sitios Web

Este proceso de pruebas está enfocado en realizar la evaluación de los sitios Web destacados del Sector del Gobierno Venezolano, los cuales deberían estar orientados a entregar accesibilidad, del cual sólo se tomará en cuenta la página con la que inicia el sitio Web. Es por esto que para estas pruebas se ha realizado la selección de un conjunto de sitios Web pertenecientes al sector del Gobierno considerados como relevantes y con frecuencia de usuarios.

3.2.1. Servicio Nacional Integrado de Administración Aduanera y Tributaria (SENIAT)

Una página corta en tamaño vertical y contenido el cual su principal objetivo es mostrar el enlace a SENIAT en Línea y un *banner* o anuncio publicitario con gráficos en Flash que notifica las noticias más destacadas. Debajo de estos elementos se encuentra el menú principal de forma horizontal. El color predominante es el rojo, con dos cabeceras de noticias de color amarillo y azul para presentar una zona como la bandera venezolana. En la siguiente imagen mostrada se presentan los resultados de la verificación de accesibilidad de la página y de los detalles de comprobación realizada.



Problemas	Advertencias	Accesibilidad del Recurso
Cantidad de Problemas:	Cantidad de Advertencias:	Grado de Accesibilidad:
14	39	15%

Detalles de la Verificación del Recurso

Perceptible

Elementos	Comprobación	Resultado	Incidencias
Enlaces	Enlace de imagen sin descripción.		5
	Enlace sin destino de referencia.		5
Imágenes	Imagen con texto alternativo vacío.		2

Operable

No se encontraron incidencias sobre este principio.

Comprensible

Elementos	Comprobación	Resultado	Incidencias
Formularios	Formulario sin botón de envío.		1
Elementos de página	Elementos de marcos sin título.		1
Tablas	Tabla sin demarcación de título.		13
	Tabla sin celdas de cabecera.		13
	Tabla sin texto de resumen.		13

Robusto

No se encontraron incidencias sobre este principio.

Resultados Generales

Total de Incidencias:	53
Total de Elementos del Recurso:	297
Total de Elementos Analizados:	78

Imagen 5.1: Evaluación de la página principal del SENIAT.
Fuente: Elaboración Propia.

Al ejecutar el análisis se destacan una de las incidencias más comunes en el ámbito de accesibilidad, el cual es el enlace de imagen sin descripción, además intenta entregar la presentación de los elementos de la página a través de tablas HTML, el cual no es su función principal de esta última, con lo que genera muchas incidencias de accesibilidad sólo por tablas. Con los pocos elementos HTML que posee, las incidencias son elevadas, lo que arroja un Grado de Accesibilidad extremadamente bajo y por ende un diseño con inconvenientes para usuarios con discapacidad visual.

3.2.2. Banco de Venezuela

Posee un diseño limpio en presentación, similar al sitio Web del SENIAT, corto en tamaño vertical, a la izquierda presenta el acceso a Clavenet donde los clientes realizan una gran cantidad de operaciones bancarias en línea y un *banner* en Flash que muestra noticias destacadas y publicidad interna de la empresa, por encima de estos elementos se encuentra el menú principal en horizontal del sitio Web en color gris. Predomina el rojo, como antiguamente ha sucedido.

Problemas	Advertencias	Accesibilidad del Recurso
Cantidad de Problemas:	Cantidad de Advertencias:	Grado de Accesibilidad:
3	8	96%

Detalles de la Verificación del Recurso

Perceptible			
Elementos	Comprobación	Resultado	Incidencias
Enlaces	Enlace de imagen sin descripción.		3
Operable			
Elementos	Comprobación	Resultado	Incidencias
Enlaces	Conjunto de enlaces sin indexacion tabulada.		7
Comprensible			
Elementos	Comprobación	Resultado	Incidencias
Elementos de página	Página sin indicador de lenguaje.		1
Robusto			
No se encontraron incidencias sobre este principio.			

Resultados Generales	
Total de Incidencias:	11
Total de Elementos del Recurso:	649
Total de Elementos Analizados:	180

Imagen 5.2: Evaluación de la página principal del Banco de Venezuela.
Fuente: Elaboración Propia.

Una vez evaluada la página Web a través de la herramienta se obtienen excelentes resultados con pocas incidencias de baja prioridad. El sitio fue desarrollado con un despliegue de elementos que favorecen a las ayudas técnicas para personas con discapacidad. En resumen, un sitio Web con soporte amplio de accesibilidad.

3.2.3. Consejo Nacional Electoral (CNE)

Posee una interfaz de sencilla con un menú principal a la izquierda, un *banner* de imágenes en el centro y el acceso la consulta de datos en el registro electoral por cédula a la derecha. La mayoría de las fuentes son de color azul pero el resto de los fondos son de color blanco, color que destaca en la página.

Detalles de la Verificación del Recurso

Perceptible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Enlace de imagen sin descripción.	✗	11
	Enlace sin aviso de nueva página.	✗	2
Imágenes	Imagen con texto alternativo vacío.	✗	15
Elementos de página	Página con elemento en movimiento.	✗	1
Operable			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Conjunto de enlaces sin indexacion tabulada.	⚠	20
Comprensible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Formularios	Campo del formulario sin etiqueta referenciada.	✗	3
Robusto			

No se encontraron incidencias sobre este principio.

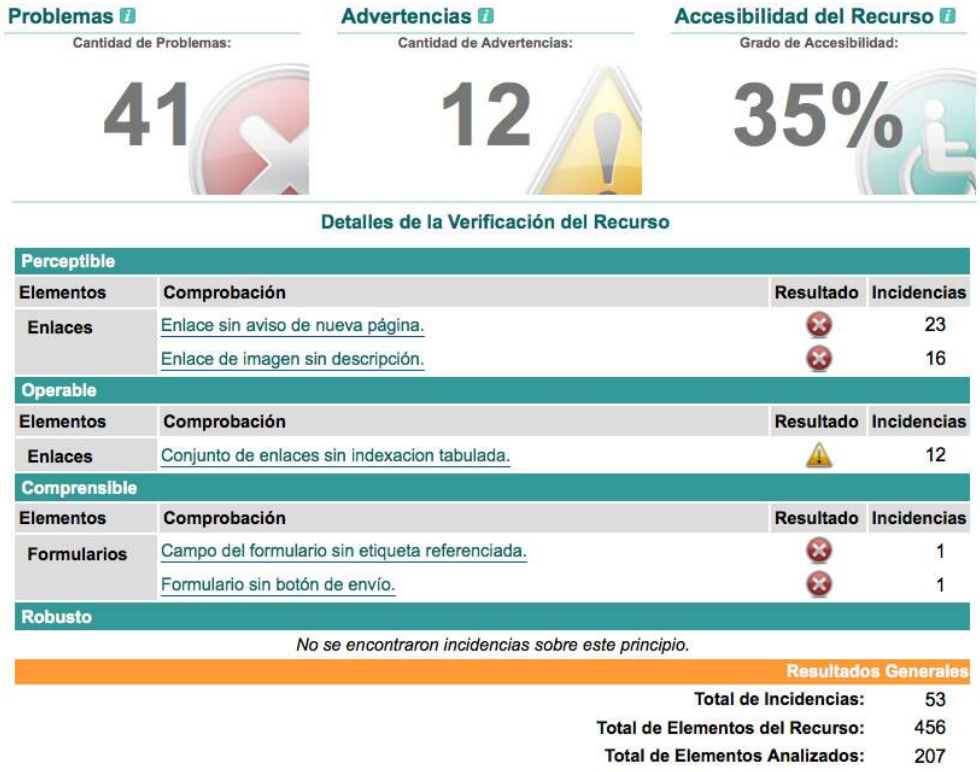
Resultados Generales	
Total de Incidencias:	52
Total de Elementos del Recurso:	363
Total de Elementos Analizados:	117

Imagen 5.3: Evaluación de la página principal del CNE.
Fuente: Elaboración Propia.

Luego de la verificación se halla otro de los problemas que más afectan la accesibilidad en los sitios Web, el cual es la ausencia del texto alternativo para las imágenes, estas incidencias imposibilitan a las ayudas técnicas a indicar la descripción textual de la imagen desplegada y cobran importancia cuando el navegador del usuario es basado en solo texto. Por estas razones tiene una caída sustancial en accesibilidad y ofrece un índice muy bajo en la misma.

3.2.4. Comisión de Administración de Divisas (CADIVI)

Al abrir la página principal, hay elementos flotantes informativos que aparecen al iniciar la página sin informar al usuario, el cual se hace a través de elementos de *JavaScripts*, lo cual no está validado por la herramienta. El fondo del contenido es gris, los menús verticales y horizontales son azul y rojo respectivamente y ambos colores son los que destacan. También posee *banners* de anuncios, uno superior para publicidades con imágenes y otro mostrar noticias textuales de forma progresiva.



Problemas	Advertencias	Accesibilidad del Recurso
Cantidad de Problemas:	Cantidad de Advertencias:	Grado de Accesibilidad:
41	12	35%

Detalles de la Verificación del Recurso

Perceptible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Enlace sin aviso de nueva página.	✖	23
	Enlace de imagen sin descripción.	✖	16
Operable			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Conjunto de enlaces sin indexacion tabulada.	⚠	12
Comprensible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Formularios	Campo del formulario sin etiqueta referenciada.	✖	1
	Formulario sin botón de envío.	✖	1
Robusto			
No se encontraron incidencias sobre este principio.			

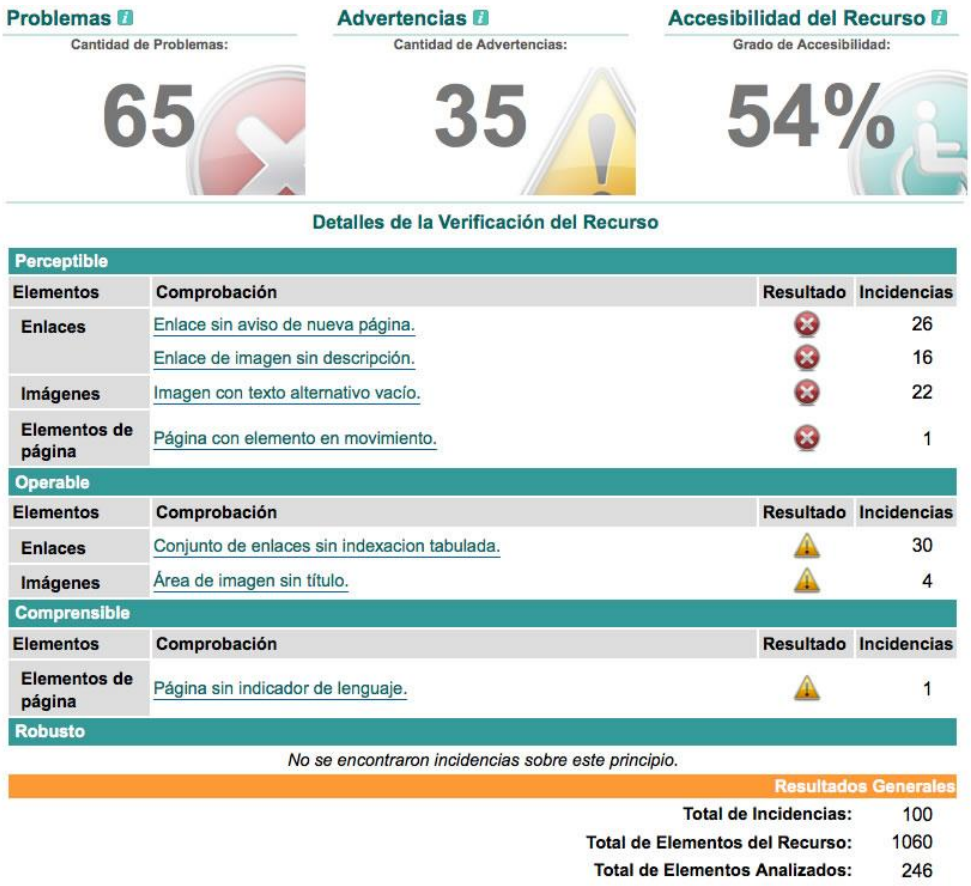
Resultados Generales	
Total de Incidencias:	53
Total de Elementos del Recurso:	456
Total de Elementos Analizados:	207

Imagen 5.4: Evaluación de la página principal de CADIVI.
Fuente: Elaboración Propia.

Las verificaciones en la página inicial de CADIVI arrojan al problema principal como los enlaces que se dirigen a una página nueva sin avisar de este comportamiento, esto provoca una desorientación a los usuarios, por tanto las ayudas técnicas no pueden informar de su presencia, tambien cuenta con incidencias comunes como los enlaces representados por imágenes que no poseen una descripción. Este conjunto de incidencias merma el grado de accesibilidad dejándolo en un nivel bastante bajo.

3.2.5. Servicio Administrativo de Identificación, Migración y Extranjería (SAIME)

Está cargada de mucha información poco resumida con un tamaño vertical considerable, varios *banners* gráficos de información y/o publicidad, muchos comunicados de prensa; contiene los tradicionales menús verticales y horizontales y una gran cantidad de enlaces externos en forma de imágenes que se encuentran ubicados a ambos lados. El azul predomina en los enlaces de texto y menús, pero los *banners* animados y las imágenes en forma de enlaces distorsionan sobre el color principal.



Problemas	Advertencias	Accesibilidad del Recurso
Cantidad de Problemas:	Cantidad de Advertencias:	Grado de Accesibilidad:
65	35	54%

Detalles de la Verificación del Recurso

Perceptible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Enlace sin aviso de nueva página.		26
	Enlace de imagen sin descripción.		16
Imágenes	Imagen con texto alternativo vacío.		22
Elementos de página	Página con elemento en movimiento.		1
Operable			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Conjunto de enlaces sin indexacion tabulada.		30
Imágenes	Área de imagen sin título.		4
Comprensible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Elementos de página	Página sin indicador de lenguaje.		1
Robusto			

No se encontraron incidencias sobre este principio.

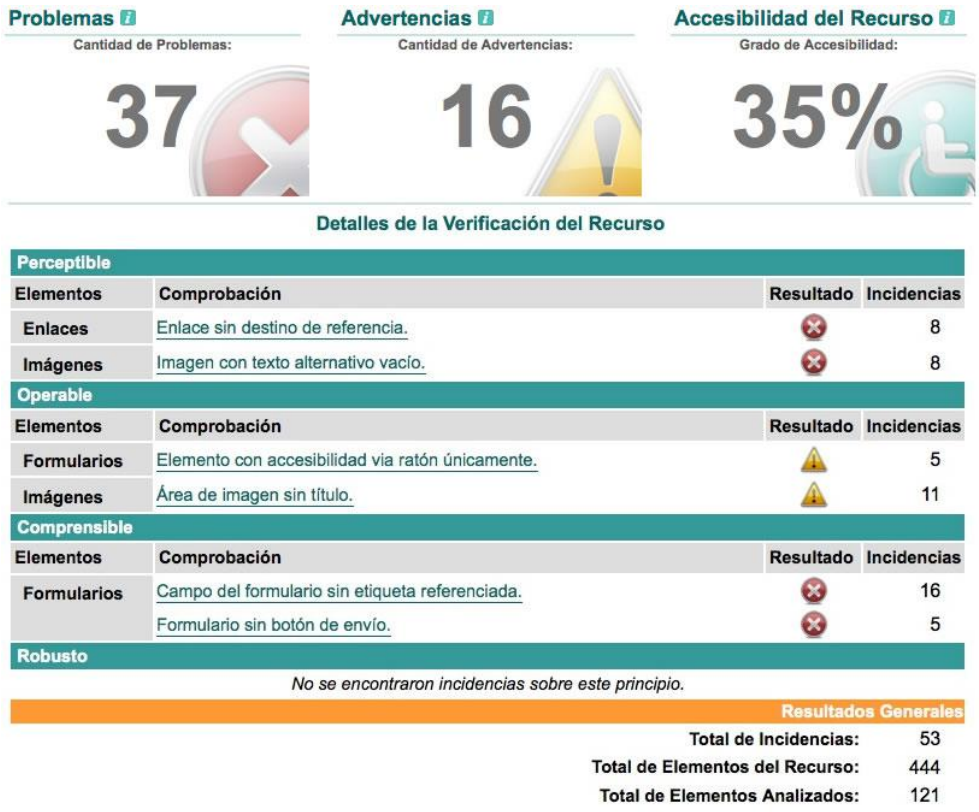
Resultados Generales	
Total de Incidencias:	100
Total de Elementos del Recurso:	1060
Total de Elementos Analizados:	246

Imagen 5.5: Evaluación de la página principal del SAIME.
Fuente: Elaboración Propia.

La evaluación de la página principal posee una cantidad relevante de incidencias, las que más destacan son referentes a la ausencia de descripciones en imágenes, enlaces de imágenes y enlaces a páginas nuevas. Sin embargo, la página evaluada posee una cantidad de elementos HTML que no intervienen en la accesibilidad, por lo que la afectación en este aspecto no es tan grave, pero estos problemas comúnes representan un gran obstáculo para los usuarios con discapacidad visual.

3.2.6. Instituto Venezolano de los Seguros Sociales (IVSS)

Una página con una variedad de colores azules en todos los renglones. Un *banner* central que abarca todo el ancho y que muestra imágenes de enlaces con títulos de noticias, por encima un menú horizontal resumido, y por debajo de éste se encuentran los menús desplegados con todos los enlaces a las opciones que este sitio ofrece. La opción principal del sitio, el sistema en línea, no destaca por encima del resto de las opciones como ofrecen los sitios anteriores.



Problemas	Advertencias	Accesibilidad del Recurso
Cantidad de Problemas:	Cantidad de Advertencias:	Grado de Accesibilidad:
37	16	35%

Detalles de la Verificación del Recurso

Perceptible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Enlace sin destino de referencia.		8
Imágenes	Imagen con texto alternativo vacío.		8
Operable			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Formularios	Elemento con accesibilidad via ratón únicamente.		5
Imágenes	Área de imagen sin título.		11
Comprensible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Formularios	Campo del formulario sin etiqueta referenciada.		16
	Formulario sin botón de envío.		5
Robusto			
No se encontraron incidencias sobre este principio.			

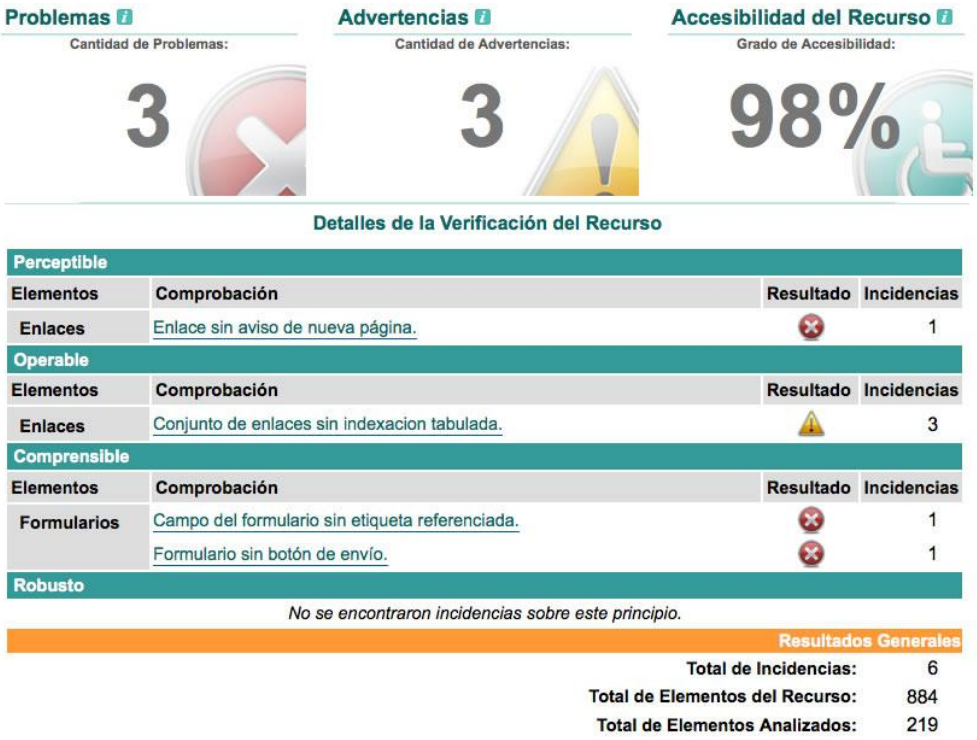
Resultados Generales	
Total de Incidencias:	53
Total de Elementos del Recurso:	444
Total de Elementos Analizados:	121

Imagen 5.6: Evaluación de la página principal del IVSS.
Fuente: Elaboración Propia.

Las páginas que recopilan algún tipo de información del usuario estan basadas principalmente de elementos de formularios, cuando éstos no son usados para tal fin, conllevan a confundir el comportamiento de estos elementos por parte de las ayudas técnicas que utilizan los usuarios con discapacidad. Este es el caso del sitio Web del Instituto Venezolano de los Seguros Sociales, donde no se está realizando el uso correcto de los formularios y afecta considerablemente la accesibilidad a usuarios que necesitan enviar información a este sitio Web. También se presentan los problemas comúnes cuando se despliegan imágenes, además de poseer áreas de imágenes sin un texto descriptivo. Por estos motivos la accesibilidad de esta página posee niveles excesivamente bajos para la cantidad de accesos que ofrece.

3.2.7. Defensoría del Pueblo

Al abrir la página principal, se presenta un *banner* superior animado en Flash del ancho total de la página con imágenes. Debajo de éste posee un menú horizontal y otro en la parte izquierda. Tiene en el contenido otros *banners* de noticias y a la derecha algunos videos informativos. Las noticias no se encuentran resumidas y posee enlaces externos a la izquierda de la página. El color naranja es el que toma protagonismo en este sitio Web.



Problemas	Advertencias	Accesibilidad del Recurso
Cantidad de Problemas: 3	Cantidad de Advertencias: 3	Grado de Accesibilidad: 98%

Detalles de la Verificación del Recurso

Perceptible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Enlace sin aviso de nueva página.		1
Operable			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Conjunto de enlaces sin indexacion tabulada.		3
Comprensible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Formularios	Campo del formulario sin etiqueta referenciada.		1
	Formulario sin botón de envío.		1
Robusto			

No se encontraron incidencias sobre este principio.

Resultados Generales	
Total de Incidencias:	6
Total de Elementos del Recurso:	884
Total de Elementos Analizados:	219

Imagen 5.7: Evaluación de la página principal de la Defensoría del Pueblo.
Fuente: Elaboración Propia.

Para la cantidad de elementos que posee la página principal de la Defensoría del Pueblo al ser evaluada, sorprende que contenga sólo 3 problemas. Con las técnicas utilizadas por la herramienta HEVAC el grado de accesibilidad se acerca al 100%, catalogando los problemas y advertencias presentados como detalles de accesibilidad que pueden haberse escapado en el diseño de la página HTML. En resumen, una nota sobresaliente en accesibilidad.

3.2.8. Compañía Anónima Nacional de Telefonos de Venezuela (CANTV)

En la página principal de CANTV.net se encuentra en la parte superior un *banner* publicitario al lado del logo del sitio, una barrar de menús horizontal roja con los principales enlaces, luego se observa una presentación de información poco común mostrando en el medio de la página una lista de noticias en vertical de tamaño pequeño que se desplazan hasta el fondo de la página, de lado izquierdo las noticias resumidas por categoría y del lado derecho un conjunto de objetos publicitarios en Flash. El rojo es el color más llamativo aunque con menos abundancia en elementos que en otras páginas evaluadas.

Detalles de la Verificación del Recurso

Perceptible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Enlace de imagen sin descripción.	✖	16
	Enlace sin aviso de nueva página.	✖	7
Imágenes	Imagen con texto alternativo vacío.	✖	4
Objetos	Página con elemento en movimiento.	✖	1
Operable			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Conjunto de enlaces sin indexacion tabulada.	⚠	9
Imágenes	Área de imagen sin título.	⚠	7
Comprensible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Formularios	Campo del formulario sin etiqueta referenciada.	✖	4
	Formulario sin botón de envío.	✖	3
Elementos de página	Página sin indicador de lenguaje.	⚠	1
Robusto			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Objetos	Objeto o plug-in sin elemento de sustitución.	⚠	1
			Resultados Generales
		Total de Incidencias:	53
		Total de Elementos del Recurso:	824
		Total de Elementos Analizados:	219

Imagen 5.8: Evaluación de la página principal de la CANTV.
Fuente: Elaboración Propia.

El análisis realizado en la página inicial de CANTV.net muestra incidencias en distintas comprobaciones presentadas, donde cobra mayor relevancia los enlaces de imágenes sin descripción. El grado de accesibilidad no se ve gravemente afectando en gran parte por la cantidad de elementos que la página principal presenta. Posee un elemento en movimiento en la zona cental superior que puede afectar a un grupo de usuarios.

3.2.9. Banco Bicentenario

Dos grandes *banners* que ocupan el ancho de la página, uno estático en la parte superior, y el otro dinámico en la parte inferior con imágenes rotativas. En el centro el menú principal en horizontal. Posee marquesinas de texto que pueden disgustar a ciertos usuarios con discapacidad visual. El color más destacado es el contraste de los grises.

Problemas	Advertencias	Accesibilidad del Recurso
Cantidad de Problemas:	Cantidad de Advertencias:	Grado de Accesibilidad:
29	10	0%

Detalles de la Verificación del Recurso

Perceptible			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Enlace de imagen sin descripción.		12
	Enlace sin destino de referencia.		1
	Enlace sin aviso de nueva página.		3
Imágenes	Imagen con texto alternativo vacío.		13
Operable			
Elementos	**Comprobación**	**Resultado**	**Incidencias**
Enlaces	Conjunto de enlaces sin indexacion tabulada.		10
Comprensible			
No se encontraron incidencias sobre este principio.			
Robusto			
No se encontraron incidencias sobre este principio.			

Resultados Generales	
Total de Incidencias:	39
Total de Elementos del Recurso:	186
Total de Elementos Analizados:	77

Imagen 5.9: Evaluación de la página principal del Banco Bicentenario.
Fuente: Elaboración Propia.

Una vez realizado el análisis del Banco Bicentenario, se halla nuevamente problemas con la falta de descripciones en imágenes y enlaces de imágenes como las comprobaciones a destacar. Su grado de accesiblidad cae al nivel mas bajo posible, un problema mayúsculo para un sitio que presenta una alta cuota de acceso de usuarios al día, como lo es una entidad financiera.

3.2.10. Facultad de Ciencias

Para complementar el conjunto de pruebas realizadas se tomó como objetivo realizar la evaluación a un sitio Web académico y éste fue la página principal de la Facultad de Ciencias de la UCV. Este sitio esta diseñado con un margen ancho muy corto, los menus de enlaces se encuentran a los lados, en el centro están anuncios que han sido diseñados con elementos de parpadeo y marquesinas desplazables hacia arriba que en ambos casos pueden ser inconvenientes para algunos usuarios, y, al final de la página se encuentran enlaces de imágenes que contrasta con el fondo que entrega la página.

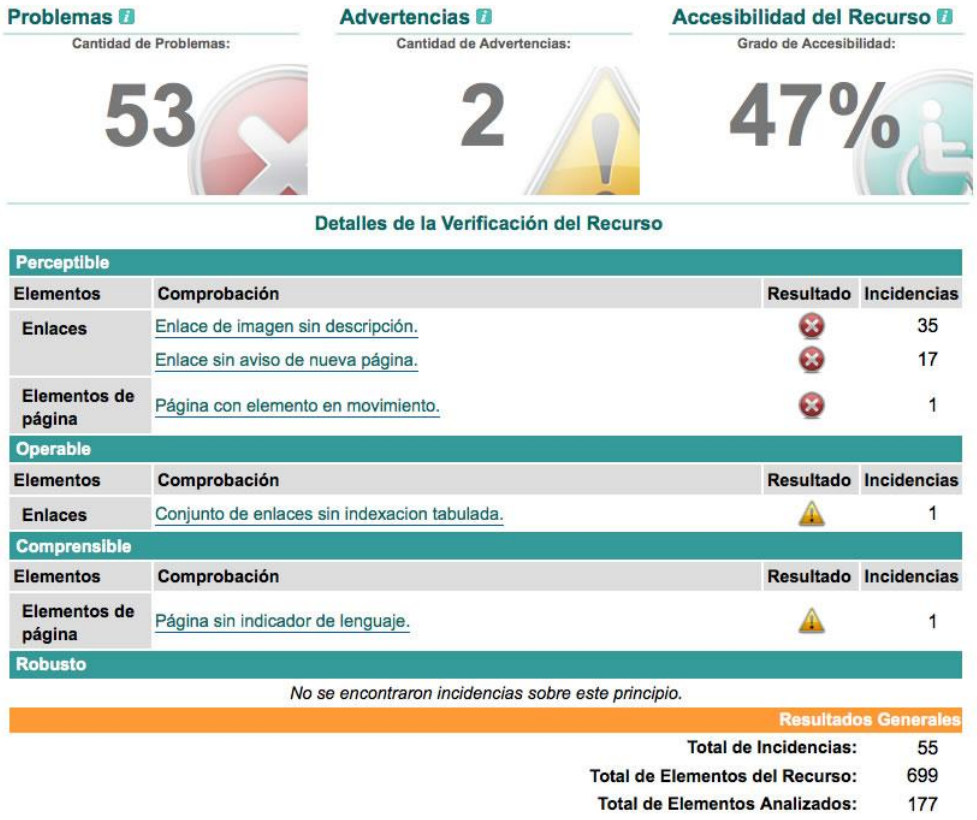


Imagen 5.10: Evaluación de la página principal de la Facultad de Ciencias de la UCV. Fuente: Elaboración Propia.

Se observa que al realizar la verificación de la página anterior, el problema principal reside en la utilización de los enlaces, donde una cantidad de enlaces que son de imágenes no poseen descripción y en algunos casos no se informa cuando este enlace va dirigido a una página nueva. El resto de las comprobaciones poseen una cantidad pequeña de incidencias, y en este caso de no ser por los problemas que presentan los enlaces poseería un nivel adecuado de accesibilidad.

3.3. Resultados Obtenidos

A continuación se muestra una tabla comparativa con los resultados obtenidos:

Página Principal	Problemas	Advertencias	Grado de Accesibilidad	Incidencia más destacada
SENIAT	14	39	15%	Información en Tablas.
Banco de Venezuela	3	8	96%	Enlaces sin indexación tabulada.
CNE	32	20	33%	Imágenes sin texto alternativo
CADIVI	41	12	35%	Enlace sin aviso de nueva pagina
SAIME	65	35	54%	Enlace sin aviso de nueva pagina
IVSS	37	16	35%	Campo de formulario sin referencia a etiqueta
Defensoría del Pueblo	3	3	98%	Enlaces sin indexación tabulada.

CANTV	35	18	68%	Enlace de imagen sin descripción
Banco Bicentenario	29	10	0%	Imágenes sin texto alternativo
Facultad de Ciencias	53	2	47%	Enlace de imagen sin descripción

Tabla 3.1: Resultados obtenidos por la evaluación a los sitios seleccionados con HEVAC. Fuente: Elaboración Propia.

Una vez finalizado los análisis de los sitios Web seleccionados, se puede observar que el 80% de las páginas Web evaluadas no cuentan con el mínimo necesario para el soporte de accesibilidad, destacan la página principal del Banco de Venezuela y de la Defensoría del Pueblo como las páginas con un excelente soporte de accesibilidad. En los problemas más frecuentes encontrados, resalta la comprobación de los enlaces de imágenes sin descripción como las principales incidencias, seguido de las imágenes con texto alternativo vacío y los enlaces sin avisos a páginas nuevas. Es importante mencionar que estos tres tipos de problemas pueden suceder al mismo tiempo si el diseñador o desarrollador de la página coloca una imagen de enlace y omite las descripciones necesarias para su presentación.

3.4. Plataforma de Desarrollo para la herramienta

Java posee una máquina virtual o intérprete de ejecución, la cual debe ser previamente instalada en el sistema operativo (en algunos sistemas operativos ya viene preinstalado) para poder ejecutar aplicaciones de Java. Con herramienta desarrollada se realizaron las pruebas en un ambiente de la siguiente especificación:

- **Sistema Operativo**: Mac OSX versión 1.7.4 (versión de 64 bits)
- **Procesador**: Intel® Core™ i5 CPU 2.4 GHz.
- **Velocidad del bus de sistema**: 1066 MHz.
- **Memoria Física del Sistema**: 8192 MB.
- **Disco duro principal**: Unidad de Estado Sólido de 256 Gb.
- **Versión de Java**: Java 6 actualización 22, versión de 64 bits.
- **Versión de servidor Tomcat**: Tomcat 7.0, especificación de Servlet 3.0, especificación de JSP 2.2.

Aunque se realizó la implementación y pruebas en este ambiente, cabe destacar que puede migrarse en un archivo WAR (comprimido final que representa físicamente la aplicación, el cual incluye el código y las librerías utilizados) a un otra plataforma distinta a la indicada que posea al menos las versiones necesarias de Java y de un servidor de aplicaciones Web o contenedor de Servlets que puedan ejecutar la aplicación satisfactoriamente.

RESULTADOS Y CONCLUSIONES

Los resultados obtenidos en el desarrollo del presente Trabajo Especial de Grado, permiten resaltar las siguientes conclusiones:

1. Se analizaron las Pautas de Accesibilidad en Contenido Web planteados por el W3C (versión 2.0), indicando sus antecedentes, organización, estructura y un enfoque basado en las tecnologías, principalmente HTML, que intervienen en la accesibilidad.
2. Se seleccionaron las Pautas de Accesibilidad citadas por el W3C a partir de la recomendación informativa de las "Técnicas de Éxito y Fallos Comunes" de cada pauta, especialmente se seleccionaron las pautas relacionadas con el uso de la tecnología HTML.
3. Se empleó una metodología de desarrollo de software *ad-hoc*, fundamentada por la utilización de los artefactos UML (diagrama de componentes, diagrama de secuencia y diagrama de clases) para el análisis y diseño, además de la metodología XP para el proceso de desarrollo de la herramienta.
4. Desde el punto de vista técnico, los logros obtenidos con el desarrollo de la herramienta HEVAC abarcan los aspectos que se describen a continuación:
    a. La herramienta ha sido desarrollada bajo el lenguaje **Java** versión 1.6, instalado en un servidor **Apache Tomcat** con las tecnologías **Servlet** y **JSP** para el procesamiento de peticiones HTTP. Esto favorece el uso de la herramienta a través de la Web sin la necesidad de realizar instalación alguna.
    b. Se seleccionó la librería adecuada para el análisis de las páginas o recursos HTML a través de un *Parser* fundamentado en la rapidez, facilidad de uso y eficiencia. La librería **Jericho HTML** permitió realizar el análisis basado en Niveles de Conformidad (Nivel AAA, Nivel AA y Nivel A) y los Principios de Accesibilidad Web citados en la especificación WCAG versión 2.0. Este análisis abarcó la verificación de la tecnología HTML de una página Web, sin embargo, no incluyó verificaciones a elementos de las tecnologías CSS y *JavaScript*.
    c. Los resultados obtenidos por el análisis de la herramienta a una página HTML muestra una vista sencilla que informa el Grado de Accesibilidad del recurso analizado. Esta vista presenta los problemas que pueden afectar principalmente a los usuarios con discapacidad y a continuación una vista detallada que informa las comprobaciones realizadas con las incidencias encontradas. También está orientada esta vista a desarrolladores y diseñadores de sitios Web, ya que permite determinar los problemas de accesibilidad presentes en una página.

d. Al realizar un análisis de una página Web se guarda automáticamente un registro en la base de datos **PostgreSQL** la cual fue seleccionada principalmente para este propósito. Al realizar más de un análisis al mismo recurso HTML, se empieza a desplegar los análisis mostrados anteriormente bajo el recurso en cuestión.
e. Como soporte alterno al resultado de la evaluación, la herramienta permite exportar los resultados de las verificaciones en formato PDF, donde se muestran las vistas simples y detalladas en un formato adaptado para la impresión. Este formato de exportación ha sido alcanzado a través del uso de librería **iText** para Java.

5. Las Pruebas de Funcionalidad realizadas a la herramienta indican que los requerimientos de Software planteados en los Casos de Uso se cumplieron en su totalidad; por otro lado, las Pruebas de Accesibilidad llevadas a cabo por la aplicación en línea WAVE demuestran que las páginas de la herramienta HEVAC no poseen errores de accesibilidad. Sin embargo, no se lograron ejecutar las Pruebas de Usabilidad debido a la dificultad de reunir a los distintos tipos de usuarios finales en sesiones de pruebas.
6. El proceso de pruebas utilizando HEVAC para analizar una muestra de sitios Web, deja entrever que aplicaciones en Venezuela sometidas al proceso de pruebas se enfocan y/o se esfuerzan en entregar un excelente diseño visual, mientras que se descuidan los aspectos de accesibilidad que merman la navegación de usuarios con discapacidad, sobre todo visual.
7. Las funcionalidades presentes en la aplicación desarrollada en el marco de este Trabajo Especial de Grado pueden brindar un aporte esencial para la verificación de accesibilidad en contenidos Web para personas con discapacidad visual y personas que desean conocer las facilidades de acceso que le puede ofrecer un portal de Internet, además de facilitar la verificación de páginas Web por parte de los desarrolladores en fase de diseño y desarrollo, y hacer los correctivos necesarios previos a la fase de entrega.

Como observación resultante del proceso de pruebas, se percibe que el enfoque y visión de la entrega de los portales Web están orientados a cumplir principalmente con los objetivos planteados por la ejecución del proyecto y estándares de diseños del sitio Web, pero, aunque un sitio Web posea un diseño pulcro y con una excelente presentación del contenido gráfico y textual, puede estar limitado en la incorporación de elementos de accesibilidad. Esto ocurre especialmente si no se toma en cuenta que un grupo de usuarios con algún tipo de discapacidad puede acceder al portal. El error parte de no aplicar el conjunto mínimo requerido de Pautas de Accesibilidad planteados por el W3C.

RECOMENDACIONES

En base a los resultados obtenidos en la solución de verificación de accesibilidad implementada, se muestran a continuación unas recomendaciones para desarrollar la solución a una escala más amplia y de esta manera cubrir con una mayor cantidad de funcionalidades, además de extender las Técnicas de Éxito que presenta el W3C. Estas recomendaciones abren la oportunidad del desarrollo de otros Trabajos Especiales de Grado en esta área.

Las recomendaciones que se presentan a continuación para ampliar las funcionalidades del prototipo actual son las siguientes:

- Se debe contemplar en su totalidad los Criterios de Éxito y por tanto sus Técnicas de Éxito planteados por el W3C en la especificación WCAG versión 2.0, de esta manera se debe ir desarrollando cada uno de estos criterios como una comprobación de la herramienta a través del componente de validación. Entre los criterios de éxito faltantes se puede citar e
- En la herramienta prototipo se pueden agregar a futuro nuevas comprobaciones en el código según la etiqueta HTML a verificar, la inclusión de estas comprobaciones conlleva a realizar creaciones o modificaciones de clases correspondientes del árbol de validación mostrada en la imagen 2.6. Los metadatos de estas comprobaciones deben crearse en el XML **conf/comprobaciones.xml** que se encuentra en el proyecto, el cual tiene un identificador con ejemplos y textos descriptivos que ayudan al desarrollador a mostrar de manera simple el detalle de la comprobación.
- Los elementos de marcos o *Frames* en HTML no son soportados por la herramienta, por lo que su incorporación como funcionalidad futura puede cubrir la necesidad de analizar las páginas que se encuentran inmersas en los marcos internos de una página Web superior solicitada para el análisis, siempre que estas páginas formen parte del mismo sitio Web.
- Una página que contiene sólo HTML tiende a conceder generalmente un carácter estático, sin embargo existen requerimientos de accesibilidad que intentan evitar que elementos con dinamismo en la página atenten contra la percepción visual o auditiva del usuario, para lograr este dinamismo es necesario emplear la tecnología de *JavaScript* o plug-Ins de terceros como *Adobe Flash*, lo que implica que la herramienta debe poseer validaciones de elementos de *JavaScripts* dentro de la página para poder advertir a los usuarios de la presencia de estos elementos con dinamismo.
- Los diseños de los sitios Web pueden poseer elementos accesibles a nivel de código en las tecnologías HTML y *JavaScripts*, pero existen sitios con contenidos visuales que pueden tener un contraste alto o bajo en la colocación de los elementos que se entregan en la página Web, para ello es necesario implementar la verificación de la tecnología CSS, el cual

dicta el patrón de diseño de los sitios Web realizando comprobaciones que contengan validaciones de fuentes, colores, tamaño mínimo de elementos, etc.

- La herramienta prototipo realiza verificaciones de un sólo recurso HTML o verificaciones página por página, en caso de que se desee evaluar la accesibilidad del sitio Web completo, es necesario que crear un proceso en segundo plano que evalúe el mapa del sitio Web y obtenga todos los recursos HTML que sean estáticos y realizar el análisis en cascada de todos estos recursos, una vez finalizado se envía un reporte vía correo electrónico o notificación posterior, ya que el proceso puede demorar un tiempo considerable para ser mostrado desde una petición HTTP.

REFERENCIAS BIBLIOGRÁFICAS

Ambler, S. W. (2004). *The Object Primer 3rd Edition: Agile Model Driven Development with UML 2.* Cambridge: Cambridge University Press.

CIF. (22 de Mayo de 2001). Clasificación Internacional de Funcionamiento, Discapacidad y Salud. *Organización Mundial de la Salud*. Ginebra, Suiza: OMS.

Clark, A. (24 de 5 de 2007). *NekoHTML*. Recuperado el 10 de 12 de 2010, de NekoHTML: http://nekohtml.sourceforge.net/

Colenbrander, A. (1999). *Guide for the Evaluation of Visual Impairment.* San Francisco, EEUU: Pacific Vision Foundation.

CTIC. (Marzo de 2009). *TAW - Servicios de Accesibilidad y Movilidad Web*. Recuperado el 10 de Agosto de 2010, de http://www.tawdis.net/index.html?lang=es

Dodds, L. (8 de 9 de 2007). *TagSoup*. Recuperado el 1 de 12 de 2010, de TagSoup: http://ccil.org/cowan/XML/tagsoup

EUMED. (Febrero de 2004). *RUP - Etapas de Diseño.* Recuperado el 21 de Mayo de 2010, de Universidad de Málaga: http://www.eumed.net/libros/2009c/584/RUP%20Diseno%20e%20implementacion%20del%20sistema.htm

ExtremeProgramming.Org. (20 de Marzo de 1999). *Extreme Programming*. Recuperado el 5 de Septiembre de 2010, de http://www.extremeprogramming.org/

González García, L. (1990). *Psicomotricidad para Deficientes Visuales (4-7 años).* Salamanca: Amarú.

Howland, E. (28 de 3 de 2006). *HotSAX*. Recuperado el 30 de 11 de 2010, de HotSAX: http://hotsax.sourceforge.net/

Hunter, J. (21 de 10 de 2009). *JDom*. Recuperado el 10 de 12 de 2010, de JDom: http://www.jdom.org/

INTECO. (Septiembre de 2009). *Instituto Nacional de Tecnologías de la Comunicación.* Recuperado el 29 de Marzo de 2012, de Guía de Transición para Evaluadores y Desarroladores WCAG 2.0: http://www.inteco.es

Jeffries, R. (Marzo de 2001). *XProgramming*. Recuperado el 21 de Septiembre de 2011, de What is Extreme Programming?: http://www.xprogramming.com/xpmag/whatisxp.htm

Jericho, M. (11 de Junio de 2009). *Proyecto Jericho HTML Parser*. Recuperado el 11 de Octubre de 2010, de http://sourceforge.net/projects/jerichohtml/

Lowagie, B. (20 de 12 de 2012). *iText*. Recuperado el 01 de 02 de 2011, de iText PDF: http://www.itextpdf.com/

Naciones Unidas. (13 de Diciembre de 2006). *United Nations Enable.* Recuperado el 31 de Octubre de 2010, de Convention on the Rights of Persons with Disabilities: http://www.un.org/disabilities/default.asp?navid=12&pid=150

NEI. (15 de Enero de 2008). *National Eye Institue*. Recuperado el 20 de Mayo de 2010, de http://www.nei.nih.gov/health/

Nikic, V. (10 de 9 de 2006). *HTML Cleaner*. Recuperado el 28 de 11 de 2010, de HTML Cleaner: http://htmlcleaner.sourceforge.net/

OMG. (Mayo de 2011). *OMG Unified Modeling Language (OMG UML), Superstructure, V2.4.* Recuperado el 15 de Febrero de 2012, de OMG Unified Modeling Language: http://www.omg.org/spec/UML/2.4/Superstructure/Beta2/PDF/

OMS. (20 de Julio de 2009). *Organización Mundial de la Salud*. Recuperado el 15 de Mayo de 2010, de Centro de Prensa: http://www.who.int/mediacentre/factsheets/fs282/es/index.html

ONCE. (26 de Julio de 2007). *Organización Nacional de Ciegos de España*. Recuperado el 30 de Abril de 2010, de http://www.once.es

Oswalt, D. (5 de 11 de 2006). *HTML Parser*. Recuperado el 9 de 12 de 2010, de HTML Parser: http://htmlparser.sourceforge.net/

Sandor, A. (9 de 7 de 2006). *JTidy*. Recuperado el 30 de 11 de 2010, de JTidy: http://jtidy.sourceforge.net/

Serfaty, O. (8 de 6 de 2007). *Java Mozilla Html Parser*. Recuperado el 29 de 11 de 2010, de Java Mozilla Html Parser: http://mozillaparser.sourceforge.net/

SIDAR. (Diciembre de 2002). *Curso de Diseño Accesible*. Recuperado el 21 de Mayo de 2010, de Fundación SIDAR: http://www.sidar.org/acti/cursos/cursotener/indice/index.htm

W3C-WAI. (Febrero de 1998). *WAI: Web Accessibility Initiative*. Recuperado el 25 de Mayo de 2010, de http://www.w3.org/WAI/

W3C-WAI-2. (Diciembre de 2008). *How To Meet WCAG 2.0.* Recuperado el 21 de Mayo de 2011, de W3C - Web Accesibility Initiative: http://www.w3.org/WAI/WCAG20/quickref/

W3C-WCAG. (11 de Diciembre de 2008). *Web Content Accesibility Guidelines.* Recuperado el 25 de Mayo de 2010, de http://www.w3c.org/TR/WCAG

WebAIM. (Marzo de 2008). *WAVE: Web Accesibility Evaluation Tool*. Recuperado el Agosto de 2010, de http://wave.webaim.org/

Wells, D. (28 de Septiembre de 2009). *Extreme Programming: A gentle Introduction*. Recuperado el 30 de Mayo de 2010, de Extreme Programming: A gentle Introduction: http://www.extremeprogramming.org/

ANEXOS

Para mayor comprensión del presente Trabajo, en el siguiente apartado se lleva a cabo la definición o explicación de algunos términos a través del Glosario de Acrónimos y Términos, además de complementar la información de algunos temas tratados en el documento.

1. Anexo 1: Glosario de Acrónimos y Términos

Apache. *Apache Software Foundation* (ASF) es una organización no lucrativa (en concreto, una fundación) creada para dar soporte a los proyectos de software bajo la denominación Apache, incluyendo el popular servidor HTTP Apache. La ASF se formó a partir del llamado Grupo Apache y fue registrada en Delaware (Estados Unidos), en junio de 1999.

API. *Application Programming Interface*. Interfaz de programación de aplicaciones. Es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas (también denominadas comúnmente "librerías").

Applet. Es un componente de una aplicación que se ejecuta en el contexto de otro programa, por ejemplo un navegador Web. El *Applet* debe ejecutarse en un contenedor, que lo proporciona un programa anfitrión, mediante un Plug-In, o en aplicaciones como teléfonos móviles que soportan el modelo de programación por "applets".

ASCII. *American Standard Code for Information Interchange*. Código Estándar Estadounidense para el Intercambio de Información. Es un código de caracteres basado en el alfabeto latino, tal como se usa en inglés moderno y en otras lenguas occidentales. Fue creado en 1963 por el Comité Estadounidense de Estándares como una refundición o evolución de los conjuntos de códigos utilizados entonces en telegrafía. Más tarde, en 1967, se incluyeron las minúsculas, y se redefinieron algunos códigos de control para formar el código conocido como US-ASCII.

ASP. *Active Server Pages*. Páginas de Servidor Activas. También conocido como ASP clásico, es una tecnología de la empresa Microsoft del tipo "lado del servidor" para páginas Web generadas dinámicamente, que ha sido comercializada como un anexo al servidor *Internet Information Services* (IIS).

Banner. Es un formato publicitario en Internet. Esta forma de publicidad online consiste en incluir una pieza publicitaria dentro de una página Web. Prácticamente en la totalidad de los casos, su objetivo es atraer tráfico hacia el sitio Web del anunciante que paga por su inclusión.

Braille. Es un sistema de lectura y escritura táctil pensado para personas ciegas. Fue ideado por el francés Louis Braille a mediados del siglo XIX, que se quedó ciego debido a un accidente durante su niñez mientras jugaba en el taller de su padre. Cuando tenía 13 años, el director de la escuela de ciegos y sordos de París –donde estudiaba el joven Braille– le pidió que probara un sistema de lecto-escritura táctil inventado por un militar llamado Charles Barbier para transmitir órdenes a puestos de avanzada sin tener necesidad de delatar la posición durante las noches. Louis Braille descubrió al cabo de un tiempo que el sistema era válido y lo reinventó utilizando un sistema de ocho puntos. Al cabo de unos años lo simplificó dejándole en el sistema universalmente conocido y adoptado de 6 puntos.

BSD. *Berkeley Software Distribution*. Distribución de software Berkeley. Es un sistema operativo derivado del sistema Unix nacido a partir de los aportes realizados a ese sistema por la Universidad de California en Berkeley.

Buffer. En informática es un espacio de memoria, en el que se almacenan datos para evitar que el programa o recurso que los requiere, ya sea hardware o software, se quede sin datos durante una transferencia.

Bus. En arquitectura de computadores, el bus (o canal) es un sistema digital que transfiere datos entre los componentes de una computadora o entre computadoras. Está formado por cables o pistas en un circuito impreso, dispositivos como resistores y condensadores además de circuitos integrados.

Byte. Para fines correctos, un byte debe ser considerado como una secuencia de bits contiguos, cuyo tamaño depende del código de información o código de caracteres en que sea definido.

Clic/Click. En informática, se denomina clic, hacer clic, "clicar", "cliquear" o pinchar a la acción de pulsar cualquiera de los botones de un ratón de computadora. Como resultado de esta operación, el sistema aplica alguna función o proceso al objeto señalado por el cursor o el puntero en el momento de realizarla.

Clip. Se refiere a un elemento multimedia de audio o video de forma acortada que funciona para indicar un anuncio o una secuencia específica.

CPU. *Central Processing Unit*. Unidad central de procesamiento. Simplemente procesador o microprocesador, es el componente del computador y otros dispositivos programables, que interpreta las instrucciones contenidas en los programas y procesa los datos.

CSS. *Cascading Style Sheets*. Hojas de estilo en cascada. Es un lenguaje usado 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.

DOM. *Document Object Model*. Modelo de Objetos del Documento. Es esencialmente una API que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos. A través del DOM, los programas pueden acceder y modificar el contenido, estructura y estilo de los documentos HTML y XML, que es para lo que se diseñó principalmente.

DTD. *Document Type Definition*. Definición de Tipo de Documento. Es una descripción de estructura y sintaxis de un documento XML o SGML. Su función básica es la descripción de la estructura de datos, para usar una estructura común y mantener la consistencia entre todos los documentos que utilicen la misma DTD. De esta forma, dichos documentos pueden ser validados, conocen la estructura de los elementos y la descripción de los datos que trae consigo cada documento, y pueden además compartir la misma descripción y forma de validación dentro de un grupo de trabajo que usa el mismo tipo de información.

Flash. *Adobe Flash Player* es una aplicación en forma de reproductor multimedia creado inicialmente por Macromedia y actualmente distribuido por *Adobe Systems*. Permite reproducir archivos SWF que pueden ser creados con la herramienta de autoría Adobe Flash, con Adobe Flex o con otras herramientas de Adobe y de terceros. Estos archivos se reproducen en un entorno determinado. En un sistema operativo tiene el formato de aplicación del sistema, mientras que si el entorno es un navegador, su formato es el de un Plug-In u objeto ActiveX.

GNOME. *GNU Network Object Model Environment.* Es un entorno de escritorio e infraestructura de desarrollo para sistemas operativos Unix y derivados Unix como GNU/Linux, BSD o Solaris; compuesto enteramente de software libre. El proyecto fue iniciado por los mexicanos Miguel de Icaza y Federico Mena y forma parte oficial del proyecto GNU. Actualmente además del español se encuentra disponible en 166 idiomas.

GNU. Es un acrónimo recursivo que significa GNU No es Unix (GNU is Not Unix). Puesto que en inglés "gnu" (en español "ñu", referido al antílope africano). GNU se identifica como un movimiento y comunidad de software y conocimientos libres con el objetivo de promocionar el desarrollo colaborativo de software y conocimiento mediante el uso de licencias libres.

Hardware. Corresponde a todas las partes tangibles de un sistema informático sus componentes son: eléctricos, electrónicos, electromecánicos y mecánicos.

HTML. *HyperText Markup Language*. Lenguaje de marcado de hipertexto. Es el lenguaje de marcado predominante para la elaboración de páginas Web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de etiquetas, rodeadas por corchetes angulares (<,>).

HTTP. *HyperText Transfer Protocol.* Protocolo de Transferencia de Hipertexto. Es el protocolo usado en cada transacción de la WWW. Fue desarrollado por el W3C y la IETF (*Internet Engineering Task Force)*, colaboración que culminó en 1999 con la publicación de una serie de RFC, el más importante de ellos es el RFC 2616 que especifica la versión 1.1. Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor.

IBM. *International Business Machines*. Es una empresa multinacional estadounidense de tecnología y consultoría con sede en Armonk, Nueva York. Fabrica y comercializa hardware y software para computadoras, y ofrece servicios de infraestructura, alojamiento de Internet, y consultoría en una amplia gama de áreas relacionadas con la informática, desde computadoras centrales hasta nanotecnología.

Indentación. Es un anglicismo (de la palabra inglesa *indentation*) de uso común en informática que significa mover un bloque de texto hacia la derecha insertando espacios o tabuladores para separarlo del texto adyacente, lo que en el ámbito de la imprenta se ha denominado siempre como sangrado o sangría.

Java. Es un lenguaje de programación y la primera plataforma informática creada por *Sun Microsystems* en 1995. Es la tecnología subyacente que permite el uso de programas punteros, como herramientas, juegos y aplicaciones de negocios. Java se ejecuta en más de 850 millones de ordenadores personales de todo el mundo y en miles de millones de dispositivos, como dispositivos móviles y aparatos de televisión.

JavaScript. Es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipificado y dinámico. Se utiliza principalmente en su forma del lado del cliente (*client-side*), implementado como parte de un navegador Web permitiendo mejoras en la interfaz de usuario y páginas Web dinámicas, en bases de datos locales al navegador.

JAWS. *Job Access With Speech*, es un software lector de pantalla para ciegos o personas con visión reducida. Su finalidad es hacer que ordenadores personales que funcionan con Microsoft Windows sean más accesibles para personas con alguna minusvalía relacionada con la visión. Para conseguir este propósito, el programa convierte el contenido de la pantalla en sonido, de manera que el usuario puede acceder o navegar por él sin necesidad de verlo.

JRE. *Java Runtime Environment.* Entorno de Ejecución de Java*.* Es un conjunto de utilidades que permite la ejecución de programas Java, el entorno en tiempo de ejecución de Java está conformado por una Máquina Virtual de Java, un conjunto de bibliotecas Java y otros componentes necesarios para que una aplicación escrita en lenguaje Java pueda ser ejecutada. Actúa como un "intermediario" entre el sistema operativo y Java.

JSP. *JavaServer Pages*. Páginas de Servidor de Java. Es una tecnología Java que permite generar contenido dinámico para Web, en forma de documentos HTML, XML o de otro tipo.

JSTE. *JavaScript Templating Language*. Lenguaje de plantillas basado en *JavaScript*. Es un lenguaje de plantillas basado en Java para la producción de contenido dinámico, que suele ser útil en las aplicaciones Web. El objetivo de JSTE es similar a la de JSP, con la diferencia clave que JSTE utiliza *JavaScript* para los aspectos computacionales de la expansión de la plantilla. Esto permite a los autores de la plantilla a emplear la máxima potencia de *JavaScript*, incluidas las funciones, los cierres, los objetos y la delegación, en la creación de plantillas.

Linux. Es un núcleo libre de sistema operativo basado en Unix. Es uno de los principales ejemplos de software libre. Linux está licenciado bajo la GPL v2 y está desarrollado por colaboradores de todo el mundo.

MathML. *Mathematical Markup Language*. Lenguaje de Marcado Matemático. Es un lenguaje de marcado basado en XML, cuyo objetivo es expresar notación matemática de forma que distintas máquinas puedan entenderla, para su uso en combinación con XHTML en páginas Web, y para intercambio de información entre programas de tipo matemático en general.

OCR. *Optical Character Recognition*. Reconocimiento Óptico de Caracteres. Es un tipo de aplicación dirigida a la digitalización de textos. Identifican automáticamente símbolos o caracteres que pertenecen a un determinado alfabeto, a partir de una imagen para almacenarla en forma de datos con los que podremos interactuar mediante un programa de edición de texto o similar.

OMS. La Organización Mundial de la Salud, es el organismo de la Organización de las Naciones Unidas (ONU) especializado en gestionar políticas de prevención, promoción e intervención en salud a nivel mundial.

ONCE. La Organización Nacional de Ciegos Españoles es una entidad de Derecho Público de carácter social y democrático "sin ánimo de lucro" que tiene el propósito fundamental de mejorar la calidad de vida de los ciegos y deficientes visuales de toda España.

Optometría. Es la ciencia encargada del cuidado primario de la salud visual, a través de acciones de prevención, diagnóstico, tratamiento y corrección de defectos refractivos, musculares y enfermedades del segmento anterior. También se ocupa del diseño, cálculo, adaptación y control de lentes de contacto y lentes oftálmicas.

Parser. Un analizador sintáctico; es una de las partes de un compilador que transforma su entrada en un árbol de derivación. El análisis sintáctico convierte el texto de entrada en otras estructuras (comúnmente árboles), que son más útiles para el posterior análisis y capturan la jerarquía implícita de la entrada.

PDF. *Portable Document Format*. Formato de documento portátil. Es un formato de almacenamiento de documentos, desarrollado por la empresa *Adobe Systems*. Este formato es de tipo compuesto (imagen vectorial, mapa de bits y texto).

PHP. Es un acrónimo recursivo que significa *PHP Hypertext Pre-processor* (inicialmente *PHP Tools*, o, *Personal Home Page Tools*). PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas Web dinámicas. Se usa principalmente para la interpretación del lado del servidor (*server-side scripting*) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas.

Plug-In. Un complemento de software que se relaciona con una aplicación para aportarle una función nueva y generalmente muy específica. Esta aplicación adicional es ejecutada por la aplicación principal e interactúan por medio de la API. También se lo conoce como *plug-in* (del inglés "enchufable"), *add-on* (agregado), complemento, conector o extensión.

RFC. *Request for Comments.* Petición De Comentarios. Son una serie de notas sobre Internet, y sobre sistemas que se conectan a internet, que comenzaron a publicarse en 1969. Cada una de ellas individualmente es un documento cuyo contenido es una propuesta oficial para un

nuevo protocolo de la red Internet (originalmente de ARPANET), que se explica con todo detalle para que en caso de ser aceptado pueda ser implementado sin ambigüedades.

RPM. Una revolución por minuto es una unidad de frecuencia que se usa también para expresar velocidad angular. En este contexto, se indica el número de rotaciones completadas cada minuto por un cuerpo que gira alrededor de un eje.

RTF. *Rich Text Format*. Formato de texto enriquecido. Es un formato de archivo informático desarrollado por Microsoft en 1987 para el intercambio de documentos multiplataforma. La mayoría de procesadores de texto son capaces de leer y escribir documentos RTF.

SAX. *Simple API for XML*, originalmente, una API únicamente para el lenguaje de programación Java, que después se convirtió en la API estándar de facto para usar XML en JAVA. Existen versiones de SAX no sólo para Java, sino también para otros lenguajes de programación. Es un analizador lineal de un documento XML que dispara eventos sin almacenamiento de una estructura jerárquica en memoria.

Schema. XML Schema es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML. Se consigue así una percepción del tipo de documento con un nivel alto de abstracción. Fue desarrollado por el W3C y alcanzó el nivel de recomendación en mayo de 2001.

Script. En informática un guión, archivo de órdenes o archivo de procesamiento por lotes, vulgarmente referidos con el barbarismo script, es un programa usualmente simple, que por lo regular se almacena en un archivo de texto plano. Los guiones son casi siempre interpretados, pero no todo programa interpretado es considerado un guión. El uso habitual de los guiones es realizar diversas tareas como combinar componentes, interactuar con el sistema operativo o con el usuario. Por este uso es frecuente que los *shells* sean a la vez intérpretes de este tipo de programas.

Servlet. Son objetos de Java que corren dentro y fuera del contexto de un contenedor de este tipo de objetos (ejemplo: Tomcat) y extienden su funcionalidad. El uso más común de éstos es generar todas páginas Web de forma dinámica a partir de los parámetros de la petición que envíe el navegador Web.

Sistema Operativo. Es un programa o conjunto de programas que en un sistema informático gestiona los recursos de hardware y provee servicios a los programas de aplicación, y corre en modo privilegiado respecto de los restantes.

Software. Se conoce como software al equipamiento lógico o soporte lógico de un sistema informático; comprende el conjunto de los componentes lógicos necesarios que hacen posible la realización de tareas específicas, en contraposición a los componentes físicos, que son llamados hardware.

Spiders. Son buscadores Web jerárquicos que funcionan recopilando información sobre los contenidos de las páginas. Cuando se busca una información en los motores, éstos consultan su base de datos y presentan resultados clasificados por su relevancia. Ejemplos de Spiders: Google, Bing, Hotbot.

TAW. Es un validador de accesibilidad y calidad Web desde hace 10 años y se ha convertido en el validador automático de referencia en España, y Merkur, herramienta para la movilización de contenidos Web que permite la adaptación de las páginas Web a dispositivos móviles.

TIC. Tecnologías de Información y Comunicación. Agrupan los elementos y las técnicas usadas en el tratamiento y la transmisión de las informaciones, principalmente de informática, Internet y telecomunicaciones.

Ubuntu. Es un sistema operativo mantenido por la empresa Canonical y la comunidad de desarrolladores. Utiliza un núcleo Linux, y su origen está basado en Debian. Ubuntu está orientado en el usuario promedio, con un fuerte enfoque en la facilidad de uso y mejorar la experiencia de usuario. Está compuesto de múltiple software normalmente distribuido bajo una licencia libre o de código abierto.

UML. *Unified Modeling Language*. Lenguaje Unificado de Modelado. Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (*Object Management Group*). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.

Unix. (Registrado oficialmente como UNIX®) es un sistema operativo portable, multitarea y multiusuario; desarrollado, en principio, en 1969 por un grupo de empleados de los laboratorios Bell de AT&T.

URL. *Uniform Resource Locator*. Localizador de recursos uniforme. Es una secuencia de caracteres, de acuerdo a un formato modélico y estándar, que se usa para nombrar recursos en Internet para su localización o identificación, como por ejemplo documentos textuales, imágenes, videos, presentaciones digitales, etc.

W3C. *World Wide Web Consortium*. Consorcio Web Mundial. Abreviado W3C, es un consorcio internacional que produce recomendaciones para la World Wide Web. Está dirigida por Tim Berners-Lee, el creador original de URL, HTTP y HTML que son las principales tecnologías sobre las que se basa la Web.

WCAG. *Web Content Accesibility Guidelines*. Son un conjunto de pautas de accesibilidad Web publicados por el grupo WAI (*Web Accesibility Initiative*) perteneciente a el W3C. Estos consisten en un conjunto de pautas para desarrollar contenido accesible en las páginas Web, principalmente para los usuarios con discapacidad, pero también para los agentes de usuario o componentes de navegación, incluyendo dispositivos altamente limitados, como los teléfonos móviles**.** La versión actual es 2.0.

WWW. *World Wide Web*. Red Informática Mundial. Es un sistema de distribución de información basado en hipertexto o hiper-medios enlazados y accesibles a través de Internet. Con un navegador Web, un usuario visualiza sitios Web compuestos de páginas Web que pueden contener texto, imágenes, videos u otros contenidos multimedia, y navega a través de ellas usando hiperenlaces.

Xerces. Es una colección de bibliotecas para el análisis validación, serialización y manipulación de documentos XML de la Apache Software Foundation. Entre ellas se incluyen DOM, SAX y SAX2.

XHTML. *Extensible HyperText Markup Language*. Es básicamente HTML expresado como XML válido. Es más estricto a nivel técnico, pero esto permite que posteriormente sea más fácil al hacer cambios o buscar errores entre otros.

XML. *Extensible Markup Language*. Lenguaje de Marcas Extensible. Es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C).

XPath. XML Path Language. Es un lenguaje que permite construir expresiones que recorren y procesan un documento XML. La idea es parecida a las expresiones regulares para seleccionar partes de un texto sin atributos. XPath permite buscar y seleccionar teniendo en cuenta la estructura jerárquica del XML.

2. Anexo 2: Elementos que intervienen en las Pautas WCAG 2.0

En este apartado se complementa la información correspondiente a los Elementos de las Pautas WCAG presentando a continuación algunos de estos.

2.1. Imágenes

Las Imágenes como elementos destacados para las Pautas WCAG 2.0 presentan las siguientes variaciones:

2.1.1. Alternativas a Imágenes

Al igual que en las Pautas WCAG 1.0, las nuevas Pautas contienen diversos criterios relativos al uso de imágenes y a la inclusión de alternativas accesibles a este tipo de elementos, al tiempo que son más específicas sobre las condiciones que deben reunir dichas alternativas en función del tipo de elemento que se quiere hacer accesible.

Existen criterios de éxito donde se establecen las características de las alternativas textuales para algunos elementos no textuales usados con frecuencia en el contenido Web, como pueden ser imágenes, audio o vídeo. Para el caso de las imágenes, algunos de estos criterios son:

- Los controles que responden a la entrada del usuario deben disponer de un nombre que identifique el propósito del elemento.
- Si el elemento no textual consiste en algún tipo de test o ejercicio que perdería validez al presentarlo como texto, la alternativa debe al menos proporcionar una descripción que identifique el contenido no textual.
- En el caso de los mecanismos de control destinados a distinguir a un humano de una máquina o programa de ordenador (CAPTCHA), se deben proporcionar distintos métodos alternativos para acceder a la información, adaptados a diferentes capacidades sensoriales.
- También se hace una referencia explícita a las imágenes o elementos puramente decorativos, cuya alternativa debe implementarse de tal modo que los productos de apoyo puedan ignorarla por completo.
- El ASCII Art se considera igualmente como un elemento no textual, por lo que debe proporcionarse una alternativa siguiendo los mismos criterios que para las imágenes o elementos de audio o vídeo.

La mayor parte de estas indicaciones ya existían de una u otra manera en el punto de verificación 1.1 de las WCAG 1.0, si bien referidos a la implementación concreta de alternativas textuales con HTML, y mencionando por tanto atributos como `alt` o `longdesc`.

En la nueva versión de las Pautas, no se indican implementaciones concretas, sino las líneas generales de actuación, por lo que la manera de proporcionar las alternativas estará condicionada a la tecnología de base usada. Por ejemplo, si se usa HTML, la alternativa para una imagen se puede proporcionar mediante el atributo `alt`, pero si la imagen es un clip de película

insertado en Flash, la alternativa se puede indicar usando la pestaña de accesibilidad de Adobe Flash. Por tanto, es importante señalar que cuando las Pautas WCAG 2.0 hablan de alternativas accesibles no se refieren necesariamente a texto en el código HTML, sino a cualquier método que proporcione la información textual requerida, siempre que el método tenga soporte para la accesibilidad.

Además, esta neutralidad tecnológica de las Pautas WCAG 2.0 implica que una página Web puede seguir siendo conforme a las Pautas aunque use una tecnología que no está directamente incluida en los navegadores Web, siempre y cuando esta circunstancia esté indicada en la Declaración de Conformidad y la tecnología usada tenga soporte para la accesibilidad. Por ejemplo, una página Web que use PDF para presentar información (y sin alternativa textual al documento PDF en sí) puede ser conforme al WCAG 2.0, ya que la tecnología PDF tiene soporte para las características de accesibilidad y es posible crear un PDF accesible.

Técnica de Ejemplo.

Dentro de las propiedades de algunas etiquetas HTML están `"alt"`, `"longdesc"` y `"title"`, éstas deberán ser utilizadas para la descripción alternativa de medios no alfanuméricos, como se ejemplifica a continuación:

```
<img src="paleta.gif" alt="paleta de acuarelas">
```

Aquí el atributo `alt` describirá la imagen como una "paleta de acuarelas" al pasar un lector de pantalla sobre ella. Si es necesario, se debe usar un enlace descriptivo (D) para identificar el medio.

 D

```
<img src="paleta.gif" alt="paleta de acuarelas">
```

Al acceder a este enlace el usuario deberá llegar a una página que contiene la descripción extensa de la imagen y su propósito.

2.1.2. Imágenes de Texto

Los criterios de éxito referentes a las imágenes de texto mencionan explícitamente algunos casos en los que se admite como válido el uso de imágenes que contengan texto:

- **Características personalizables**: según la tecnología usada, es posible proporcionar mecanismos que permitan al usuario ajustar las características del texto a sus preferencias; en estos casos, es admisible la inclusión de texto en la imagen, ya que el usuario tiene forma de ajustarlo a sus preferencias.
- **Uso esencial**: se refiere a aquellos usos donde la inclusión del texto en la propia imagen es clave para transmitir información (por ejemplo, muestras de tipografías, imagen de marca, tipografías especiales, símbolos).
- **Logotipos**: se consideran siempre como excepción al ser esencial para transmitir la información la inclusión del texto del logotipo con un formato, color o apariencia concreta e identificativos de la imagen de marca.

Las Pautas WCAG 2.0 recogen también algunos casos de uso válido de texto en imágenes para lograr efectos visuales concretos que no son reproducibles de otra manera, si bien habría que considerar la tecnología específica que se esté usando. Por ejemplo, en el caso de HTML y CSS, para crear efectos de sombra o de texto reflejado puede ser necesario recurrir a imágenes que representen estos efectos, mientras que si se usa Flash es posible manipular el texto directamente para crear dichos efectos visuales.

En cualquier caso, si es posible lograr el mismo efecto visual mediante la tecnología concreta que se esté usando (por ejemplo, mediante CSS), las Pautas WCAG 2.0 siguen manteniendo que debe usarse texto en lugar de imágenes para representar la información.

2.1.3. Mapas de Imagen

Respecto a los mapas de imagen, además de lo ya expuesto sobre las alternativas textuales, en las Pautas WCAG 2.0 se contemplan algunos criterios referidos a la operabilidad con teclado de los mapas de imagen.

Por un lado, los criterios de éxito que indican que la funcionalidad (en este caso la ofrecida por el mapa de imagen) debe ser accesible mediante teclado; dado que en HTML los mapas de imagen de cliente son accesibles directamente mediante teclado, mientras que los mapas de servidor no lo son, las Pautas WCAG 2.0 consideran no válidos los mapas de imagen de servidor, y por lo tanto debe proporcionarse una alternativa textual a los mismos.

Por otro lado, en las WCAG 2.0 se especifica que cada vínculo debe estar asociado con un texto que indique el propósito del vínculo, por lo que sigue siendo necesario proporcionar alternativas para cada zona activa de los mapas de imagen.

Técnica de Ejemplo

El siguiente ejemplo muestra un mapa de imagen cliente accesible, en éste se enlaza la imagen con las secciones de Golf, Tenis, Motociclismo y Natación de la página. Es recomendable que para cada etiqueta `<area>` se agregue el atributo `title`, con esto al posarse sobre el área del mapa se recibe una indicación de donde lleva el enlace.

```
<map name="entrada">
<area alt="Golf" title="Informaci&oacute;n para Golfistas." shape="poly"
coords="147, 36, 167, 36, 183, 53, 197, 68, 197,82, 198, 178, 167, 175,
159, 164, 173, 153, 178, 146, 177, 140, 172, 130, 172, 122, 172, 118,
154, 123, 162, 86, 159, 65, 152, 61, 146, 51" href="ejmap.htm#golf">
<area alt="Tenis." title="Informaci&oacute;n para tenistas." shape="poly"
coords="76, 39, 85, 37, 92, 37, 102, 46, 101,54, 107, 54, 124, 67, 134,
83, 132, 99, 123, 112, 132, 133, 137, 143, 129, 160, 99, 160, 102, 147,
109, 142, 97,120, 96, 108, 89, 116, 82, 116, 72, 105, 78, 88, 78, 78, 78,
67, 80, 64, 75, 58, 75, 56, 70, 48" href="ejmap.htm#tenis">
<area alt="Motociclismo." title="Informaci&oacute;n para Motociclistas"
shape="poly" coords="34, 102, 28, 115, 27, 127, 26, 142, 32, 156, 34,
177, 45, 181, 59, 171, 68, 176, 85, 178, 92, 181, 105, 185, 105, 175, 93,
164, 96, 151, 81, 130, 71, 119, 61, 118, 55, 115, 52, 105, 45, 102, 40,
102" href="ejmap.htm#motos">
<area alt="Nataci&oacute;n." title="Informaci&oacute;n para nadadores."
shape="poly" coords="19, 7, 17, 22, 25, 38, 25, 49, 25, 53, 21, 62, 16,
75, 10, 89, 7, 100, 8, 122, 8, 137, 4, 160, 4, 176, 12, 179, 20, 173, 22,
160, 26, 147, 25, 120, 27, 109, 35, 99, 41, 96, 42, 84, 46, 73, 58, 63,
68, 53, 65, 43, 49, 19, 35, 19" href="ejmap.htm#nata">
</map>
<img src="depor.jpg" width="200" height="200" border="0"
usemap="#entrada" alt="Nuestras secciones: Golf, Motociclismo,
Nataci&oacute;n y Tenis; seleccione una de las fotos o el enlace en texto
de debajo."><p>
<a href="ejmap.htm#golf" title="Informaci&oacute;n para
golfistas.">Golf</a>|
<a href="ejmap.htm#moto" title="Informaci&oacute;n para
motociclistas">Motos</a>|
<a href="ejmap.htm#nata" title="informaci&oacute;n para
nadadores.">Nataci&oacute;n</a>|
<a href="ejmap.htm#tenis" title="Informaci&oacute;n para
tenistas.">Tenis</a>
</p>
```

El código expuesto anteriormente se vería de la siguiente manera:

Imagen 6.1: Imágenes con descripciones por área.
Fuente: Elaboración Propia.

Se puede observar que cada vez que se está sobre el enlace, el atributo `title` indica de qué trata el área o el enlace.

2.2. Multimedia

En los elementos multimedia existen diferentes contextos que son relevantes a la hora de entregar contenidos accesibles, los cuales se presentan a continuación.

2.2.1. Alternativas textuales al contenido multimedia

Al igual que en el caso de las imágenes, el criterio de éxito 1.1.1 (Nivel A) también recoge algunas características que deben poseer las alternativas para algunos elementos multimedia:

- En elementos multimedia que dependen del tiempo, la alternativa textual debe proporcionar al menos una descripción del elemento.
- En elementos que están pensados para provocar un efecto que sólo es perceptible si se dispone de una determinada capacidad sensorial (por ejemplo, la vista o el oído), la alternativa debe al menos identificar el elemento con una etiqueta descriptiva.

Además, si se trata de elementos exclusivamente de audio o vídeo pregrabado (por ejemplo, un programa de radio, o un vídeo sin sonido), el criterio 1.2.1 (Nivel AA) establece que se debe proporcionar una alternativa que ofrezca la misma información que el elemento multimedia, por ejemplo mediante una transcripción textual; en el caso del vídeo esta alternativa también puede consistir en una pista de audio sincronizada con el vídeo.

2.2.2. Audio-descripción y subtitulado

En las Pautas WCAG 2.0 existen varios criterios de éxito relativos a estas alternativas, todos ellos agrupados bajo la **Pauta 1.2** (alternativas para contenido multimedia).

Los criterios de éxito referentes a este aspecto indican que se ha de proporcionar una alternativa textual o audio-descripción para los contenidos de vídeo pregrabados, salvo cuando estos sean una alternativa o apoyo para un contenido textual, en cuyo caso no necesitan alternativa. En el caso de optar por proporcionar una alternativa textual en lugar del audio-descripción, ésta debe consistir en una descripción completa de todo el contenido, tanto visual como auditivo, incluyendo escenarios, expresiones, gestos, diálogos, etc., como si de un guión se tratase.

En cuanto a los subtítulos, se establece que se deben incorporar subtítulos para el contenido pregrabado de vídeo con audio sincronizado, a menos que este vídeo sea una alternativa o apoyo para un contenido textual. En cambio, si el contenido de vídeo es en directo, también se indica la necesidad de proporcionar subtítulos para este tipo de contenidos.

2.2.3. Control de reproducción

Las Pautas WCAG 2.0 incluyen criterios muy específicos relativos al control que el usuario debe tener sobre la presentación y reproducción de contenidos. En este sentido, se ha incorporado un nuevo criterio de éxito, relativo a la reproducción automática de audio, relacionado con el requisito de conformidad de no-interferencia.

Este criterio de éxito indica que, si un elemento de audio se reproduce de forma automática durante más de 3 segundos, se debe proporcionar un mecanismo para detenerlo, o en su defecto un control de volumen independiente del volumen general del sistema, de modo que el audio no interfiera con la verbalización de un lector de pantalla, por ejemplo.

2.2.4. Acceso mediante teclado

En las Pautas WCAG 2.0 se incluyen diversos criterios genéricos relativos al manejo de la interfaz de usuario y los controles interactivos. Muchos de estos criterios tienen especial aplicación

en el caso de elementos como audio o video, donde se hace necesaria una interfaz que permita controlar la reproducción de los contenidos. Habitualmente, esta interfaz dispondrá de botones o vínculos para iniciar la reproducción, pausarla o detenerla, además de otros elementos como control de volumen, mostrar/ocultar subtítulos, audio descripción, etc.

Técnica de Ejemplo:

El siguiente ejemplo ayuda a comprender este criterio:

Una grabación de audio de una conferencia de prensa: Una página Web incluye un enlace a una grabación de audio de una conferencia de prensa que identifica a la grabación de audio. La página también contiene enlaces a una transcripción del texto de la conferencia de prensa. La transcripción incluye un acta literal de todo lo que dicen los oradores. Se identifica quién está hablando, así como señalar otros sonidos importantes que forman parte de la grabación, tales como los aplausos, las risas, las preguntas del público, y así sucesivamente.

2.3. Maquetación y Presentación

Existen diversas formas utilizar algunos elementos y tecnologías para maquetar el contenido sin ser confundidos como contenido de presentación y que deben ser analizados en su contexto como elementos accesibles para las Pautas WCAG 2.0.

2.3.1. Tablas de maquetación

En ciertos puntos de verificación de la versión WCAG 1.0 se contemplaba el uso de tablas para maquetar contenidos; además, se realiza también una mención expresa al uso incorrecto de celdas de encabezado para realizar efectos de presentación.

El primero de estos puntos se recoge de una manera más genérica en la versión WCAG 2.0, que se refiere a la secuencia en la que se presenta la información, que debe poder determinarse programáticamente (esto es, el marcado estructural permite a un agente de usuario determinar el orden de lectura).

2.3.2. Hojas de estilo

Existen criterios de éxito que se pueden interpretar como que no se usen características de CSS que no estén soportadas por los navegadores o por los productos de apoyo (por ejemplo, el uso de algunas propiedades de CSS 2.1 no soportadas por Internet Explorer). En cuanto a la no-interferencia, se tratará de evitar características de CSS que puedan interferir con el acceso a la información; por ejemplo, es erróneo el uso de `display:none` para ocultar información

visualmente, pero que sí debe ser leída por los lectores de pantalla, ya que esta propiedad ocultará la información también a los lectores de pantalla, y no sólo visualmente.

2.3.3. Uso del color y contraste

El uso del color para transmitir información, recogido anteriormente en los puntos de verificación de la WCAG 1.0, indican que no debe usarse el color como único medio visual de transmitir información, indicar una acción, solicitar una respuesta o distinguir un elemento visual.

Los criterios de éxito mejoran el aspecto del contraste, donde se especifica un contraste mínimo con una proporción de 4,5:1 no sólo para las imágenes, sino también para el texto, contemplando algunas salvedades:

- **Texto de gran formato**: si la tipografía usada en el texto o en las imágenes de texto es de gran tamaño, se permite una proporción de contraste de 3:1.
- **Uso incidental**: por ejemplo, en elementos de interfaz inhabilitados, en texto que se considera decorativo o que son parte de una imagen donde el contenido relevante está en el resto de la imagen. En estos casos no se necesita un contraste mínimo.
- **Logotipos**: el texto que forma parte de un logotipo o de un nombre comercial son considerados excepciones y no se exige un contraste mínimo.

Técnica de Ejemplo

Los colores utilizados para desplegar información en la página deberán contrastar de modo que el esfuerzo visual sea mínimo al tratar de distinguir la información. El contraste es un fenómeno con el que se pueden diferenciar colores atendiendo a la luminosidad, al color de fondo sobre el que se proyectan. Existen distintos tipos de contraste, pero es recomendable utilizar el contraste claro-oscuro, éste se produce al confrontar un color claro, o saturado de blanco, y un color oscuro, o saturado de negro.

Contraste de luminosidad Ejemplo de alto contraste y bajo contraste

Imagen 6.2: Algunos tipos de contraste.
Fuente: Wikipedia.org

2.3.4. Unidades de medida

En la versión anterior se indicaba la necesidad de usar unidades de medida relativas al especificar tamaños de fuente o de otros elementos. En la nueva versión de las Pautas no existe tal exigencia, requiriéndose en su lugar que el texto pueda ser escalado hasta un 200% sin necesidad de productos de apoyo. Así, las Pautas 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. No obstante, cabe mencionar que el uso de unidades relativas como "em" o los porcentajes se considera una técnica de suficiencia para cumplir este criterio.

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 funcionalidad.

2.4. Metadatos

A continuación se presentan los distintos tipos de Metadatos que forman parte del contenido de accesibilidad para las Pautas WCAG 2.0.

2.4.1. Mecanismos de Navegación

Al igual que en las WCAG 1.0, la nueva versión de las Pautas contiene diversos criterios relacionados con la accesibilidad de los mecanismos y elementos de navegación, desde los que se refieren a elementos concretos dentro de una página Web, como enlaces, botones, barras o menús, a aquellos que se extienden a secciones del sitio o a la totalidad del mismo, como pueden ser el mapa del sitio, una interfaz común de navegación o las herramientas de búsqueda.

2.4.2. Objetivo de los vínculos

En las Pautas WCAG 1.0, se establece que se debe identificar de forma adecuada el objetivo de los vínculos. De forma similar, en las WCAG 2.0 se establece que el propósito de cada vínculo debe poder determinarse programáticamente sin ambigüedad. En este sentido, las nuevas Pautas son menos restrictivas que las anteriores, y se entiende que el propósito del vínculo puede ser deducido no sólo del texto del mismo, sino también uniéndolo a su contexto, por lo que ahora los vínculos del tipo "ver más" se pueden considerar correctos, siempre y cuando el contexto permita averiguar con exactitud el destino o propósito del vínculo. Este criterio de éxito admite como excepción los casos donde el vínculo es ambiguo para todos los usuarios, por ejemplo si se

trata de un *banner* publicitario que busca llamar la atención sin desvelar la información, para que el usuario sienta la curiosidad de acceder al contenido que se oculta detrás del *banner*.

2.4.3. Estructura y organización de la navegación

En las Pautas WCAG 2.0 existen varios criterios de éxito que hacen alusión a la localización de la información y al uso de los mecanismos de navegación de una forma predecible por parte del usuario. En primer lugar, los criterios indican que deben proporcionarse múltiples vías para que los usuarios puedan localizar una página Web dentro de un conjunto de páginas, salvo cuando dicha página sea el resultado de un paso que forma parte de un proceso. Sin embargo, estos criterios de éxito no establecen ningún requisito sobre mecanismos concretos que deban usarse, sino que deja abierta la posibilidad de usar distintos sistemas en función de la complejidad y las necesidades del sitio. Algunos de estos sistemas podrían ser:

- En sitios pequeños, puede bastar un simple enlace a la página principal, que permite acceder al resto de las páginas del sitio, o un menú de navegación visible en todas las páginas con esa misma función.
- Si la complejidad del sitio aumenta, un enlace en todas las páginas al mapa del sitio o a una tabla de contenidos puede ayudar a localizar la información rápidamente.
- En sitios de gran complejidad se puede incluir un sistema de búsqueda mediante palabras clave o de cualquier otro tipo.

2.4.4. Redireccionamiento Automático

Los antiguos punto de verificación de las Pautas WCAG 1.0, relativo a los métodos de re-direccionamiento automático de páginas, se puede encontrar ahora enmarcado dentro de criterios de éxito en la nueva versión, que con un carácter mucho más amplio hace referencia a la capacidad del usuario para ajustar cualquier límite de tiempo existente en el contenido web.

Esto significa que el usuario debe poder desactivar, ajustar o extender el tiempo límite establecido, salvo en los casos donde este límite temporal sea fundamental para la correcta interpretación del contenido (por ejemplo, en procesos en tiempo real o en aquellos casos donde cambiar este límite invalidaría el proceso, tales como un examen). También se considera válido un proceso o tarea cuyo tiempo límite es superior a las 20 horas.

Por otro lado, es de señalar que las Pautas WCAG 2.0 contemplan como válido el uso de redirecciones automáticas con tiempo nulo, esto es, en aquellos casos donde la redirección se produce inmediatamente al entrar en la página, por ejemplo a través de un elemento meta de auto-refresco o mediante una función de JavaScript, entre otros métodos.

2.4.5. Ventanas Emergentes

En las Pautas WCAG 1.0, los puntos de verificación indicaban que no se debían abrir ventanas nuevas del navegador sin avisar al usuario de tal circunstancia. La nueva versión de las Pautas considera la apertura de este tipo de ventanas como "cambios de contexto", a los que hace referencia en los criterios de éxito, relativos a los cambios de contexto que se producen cuando un elemento toma el foco o cuando el usuario realiza alguna entrada de datos. El primer caso, por ejemplo, se produce cuando se abre una ventana emergente de forma automática al entrar en un control de formulario, mientras que el segundo puede producirse cuando un usuario selecciona una opción de un menú desplegable y este evento lanza la apertura de la nueva ventana.

Por otra parte, las WCAG 2.0 mencionan específicamente en los criterios de éxito varias técnicas de suficiencia de apertura de nuevas ventanas, en las que se avisa al usuario de tal evento; en este sentido, se considera válido el uso del atributo `target` siempre que el texto del vínculo avise de la apertura de la nueva ventana, así al igual que otras técnicas de mejora progresiva utilizando JavaScript no invasivo.

Técnicas de Ejemplo:

Se utiliza `target` para presentar una ventana emergente. Se debe incluir el texto a la ventana que informe a los usuarios no videntes de la aparición de ésta. En el caso que un enlace se abra en una nueva ventana es necesario informar al usuario de esta acción para mantener el contexto de la lectura, el siguiente ejemplo muestra la acción antes descrita.

```
<p>Visite el <a href="nueva_pagina.html" target="_blank">Sitio Web del W3C<a>
(Advertencia: Esta p&aacute;gina se abrir&aacute; en una nueva ventana.)</p>
```

En el navegador se vería del siguiente modo: Visite el Sitio Web del W3C (Advertencia: Esta página se abrirá en una nueva ventana.)

3. Anexo 3: Técnicas Utilizadas

A continuación se presentan las Técnicas utilizadas en los algoritmos de comprobaciones de la herramienta. Las Técnicas vienen identificadas con su nombre, descripción del suceso y recomendación de desarrollo, Nivel de Conformidad y Principio de Accesibilidad.

- **Campo de texto sin orden de tabulación**. Es necesario que los campos de texto tengan orden al tabular. Se debe utilizar el atributo `"tabindex"` en la etiqueta `<input>` que sea de tipo Campo de Texto, ya que es necesario que exista un orden al cambiar de un campo a otro con la tecla Tab. Nivel AA. Principio: Operable.

- **Campo de texto sin nombre**. Debe identificarse el nombre del campo de texto de un formulario. Se debe colocar el atributo "name" en la etiqueta <input> que sea de tipo Campo de Texto. Nivel A. Principio: Perceptible.
- **Objeto o *plug-in* sin elemento de sustitución**. El objeto debe poseer una imagen o texto que lo sustituya en caso de no encontrarse el *plug-in* en el navegador. Debe usar en los objetos alojados, imágenes o textos de sustitución en caso que falle la carga del objeto o no se encuentre disponible en el navegador. Nivel AA. Principio: Robusto.
- **Enlace sin aviso de nueva página**. Se están presentando enlaces a nuevas ventanas sin avisar debidamente al usuario. No debe provocar la llamada a ventanas emergentes sin informar al usuario. Para no confundir a los usuarios no videntes es necesario avisar cuando se abra una nueva ventana; de lo contrario, el usuario pensará que está en la ventana anterior. Nivel A. Principio: Perceptible.
- **Campo de imagen sin texto alternativo**. Debe identificarse el título del campo de imagen de un formulario. Se debe colocar el atributo "alt" en la etiqueta <input> que sea de tipo Campos de Imagen para que muestren un texto alternativo en caso de que estos campos no puedan ser visualizados. Nivel A. Principio: Perceptible.
- **Área de imagen sin título**. El área marcada para las imágenes debe tener un título descriptivo. Es recomendable que para cada etiqueta <area> se agregue el atributo "alt", con esto al posarse sobre el área del mapa se recibe una indicación de donde lleva el enlace. Nivel AAA. Principio: Operable.
- **Botón de formulario sin texto**. Es necesario que todo botón del formulario tenga un texto. Se debe colocar el texto al botón del formulario dentro de la etiqueta <button>. Nivel A. Principio: Perceptible.
- **Tabla sin texto de resumen**. Debe implementarse un resumen en cada tabla de datos. Utilizar el atributo "summary" en HTML, en la etiqueta <table>, para hacer un resumen de la tabla actual. de esa forma el contenido de la tabla será más entendible por parte de los usuarios. Nivel AA. Principio: Comprensible.
- **Página sin indicador de lenguaje**. Es necesario que se indique idioma de origen de la página. Se debe colocar el identificador del lenguaje en el atributo "lang" de la etiqueta <html>. Nivel A. Principio: Comprensible.
- **Enlace sin destino de referencia**. No se ha colocado el destino de referencia obligatorio en el enlace. Se debe colocar el destino del enlace en el atributo "href" de la etiqueta <a>. Nivel A. Principio: Perceptible.

- **Imagen sin enlace hacia su descripción**. No existe enlace URI de descripción de la imagen presentada. Se debe colocar la descripción de la imagen en un enlace con el atributo "longdesc" para la etiqueta <img>. Nivel AA. Principio: Comprensible.
- **Página con elemento en movimiento**. Se encontraron elementos en movimiento que pueden ser peligrosos a cierto grupo de usuarios. La etiqueta <marquee>, que proporciona texto en movimiento, no es controlable. En el caso de requerir texto en movimiento se debe utilizar otras tecnologías compatibles con ayudas técnicas que permitan brindar al usuario un completo control del movimiento del texto. Nivel A. Principio: Perceptible.
- **Campo del formulario sin etiqueta referenciada**. Debe identificarse la etiqueta de un formulario con su control. Es necesario utilizar la marca <label> y su propiedad "for" para identificar correctamente las etiquetas. Nivel A. Principio: Comprensible.
- **Conjunto de enlaces sin indexación tabulada**. No se ha especificado un índice para el grupo de enlaces. Se debe utilizar el atributo "tabindex" en la etiqueta <a> para indexar los enlaces por teclado, ya que es posible que dispositivos como el ratón no se puedan utilizar. Nivel AAA. Principio: Operable.
- **Página con elemento de parpadeo**. Se encontraron elementos de parpadeo que pueden ser peligrosos a cierto grupo de usuarios. La etiqueta <blink>, que proporciona el efecto de parpadeo, no está dentro de los estándares HTML, por lo que su uso no está permitido para sitios web con soporte de accesibilidad. Nivel A. Principio: Perceptible.
- **Botón de formulario sin valor**. Debe colocarse un valor (nombre) al botón del formulario. Se debe colocar el atributo "value" en la etiqueta <input> que funcionen como campos de botón ("submit", "reset" o "button"). Nivel A. Principio: Comprensible.
- **Elementos de marcos sin título**. No se han etiquetados los marcos de la página. Se debe utilizar el atributo "title" en la etiqueta <frame> o <iframe> para etiquetar los marcos de las páginas. Nivel A. Principio: Comprensible.
- **Celda de cabecera sin objetivo**. La celda de encabezado debe indicar si es una cabecera de columna o de fila. Usando el atributo "scope" se podrá identificar el tipo de celda de encabezado. Ejemplo: scope="col" indica que esta celda es la cabecera de todas las demás celdas que están en la misma columna. Nivel AA. Principio: Comprensible.
- **Página con refrescamiento automático**. La página posee un refrescamiento automático que el usuario no puede controlar. Debe eliminar el refrescamiento automático, en caso de que la página lo requiera, debe dejar que el usuario elija el tiempo a refrescar o detenerlo si es necesario. Nivel A. Principio: Perceptible.

- **Selección de formulario sin opciones**. Es necesario que toda selección del formulario tenga al menos dos opciones. Debe proporcionar dos o más opciones en las selecciones (etiqueta `<select>`) de los formularios con la etiqueta `<option>`. Nivel AA. Principio: Perceptible.
- **Enlace de imagen sin descripción**. No hay una descripción contigua a la imagen que representa el enlace. Si el enlace es una imagen, se debe proporcionar un texto al lado de la imagen que la represente u otro enlace contiguo en modo texto que represente la imagen. Nivel A. Principio: Perceptible.
- **Tabla sin demarcación de título**. La tabla se encuentra sin un título descriptivo. Dentro de la tabla, con la etiqueta `<caption>` se referencia el título de la tabla, con esto los usuarios entenderán el contexto en el que se sitúa la tabla. Nivel AA. Principio: Comprensible.
- **Imagen con texto alternativo vacío**. Debe comprobarse que la imagen es decorativa. Se debe colocar el texto alternativo en el atributo `"alt"` para la etiquetas `<img>` en imágenes no decorativas. Nivel A. Principio: Perceptible.
- **Elemento con accesibilidad vía ratón únicamente**. Hay elementos que sólo son accesibles por una única interfaz (vía ratón). El uso de atributos *JavaScript*, como `"onMouseOver"` u `"onClick"`, que exijan el uso de un dispositivo de entrada distinto al teclado (en este caso el Ratón), limitan las funcionalidades del sitio tornándolo menos accesible. Nivel A. Principio: Operable.
- **Tabla sin celdas de cabecera**. En las tablas se debe identificar los encabezados de las columnas y las filas. Con la etiqueta `<th>` se identifica los encabezados de las columnas o filas de las tablas, de este modo será fácil distinguir el tipo de información que entrega la tabla. Nivel AA. Principio: Comprensible.
- **Formulario sin botón de envío**. Es necesario que todo formulario tenga un botón de envió. Se debe insertar un botón de envío con la etiqueta `<input>` de con el valor `"submit"` para todo formulario en la página. Nivel A. Principio: Comprensible.

4. Anexo 4: Tecnologías Utilizadas en la Herramienta

A continuación se presenta una breve descripción de cada una de las Tecnologías utilizadas durante el proceso de implementación de la herramienta HEVAC.

4.1. Java

Java es un lenguaje de programación orientado a objetos, desarrollado por la empresa *Sun Microsystems* a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y

C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Las aplicaciones Java están típicamente compiladas en códigos de byte, aunque la compilación en código máquina nativo también es posible. En tiempo de ejecución, el código de bytes es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del código de bytes por un procesador Java también es posible.

4.1.1. Ventajas

- Es orientado a objetos. De esta manera, se pueden modelar las soluciones de los sistemas a implementar como objetos que poseen operaciones y que a su vez pueden tener distintas formas y comportamientos. Esto apoya el uso de patrones de diseño estandarizados para resolver los problemas de manera sencilla y escalable.
- Es código abierto. Es de libre utilización (no se necesitan licencias comerciales) tanto la máquina virtual, como la mayoría de las librerías de soporte al lenguaje y ambientes de desarrollo integrados.
- Es robusto. Dado a que se ejecuta en un ambiente de memoria cerrado que puede atrapar todo tipo de errores (excluyendo los del hardware o del sistema operativo) que se cometen normalmente durante el desarrollo, evitando así, que el aplicativo en ejecución se vea afectado.
- Es multiplataforma (portable). Por lo que se desarrolla una sola versión del código final y se ejecuta el mismo código en todas las plataformas donde se encuentra disponible la máquina virtual.
- Manejo de memoria autonómica. Así se evitan las posibles fugas de memoria sin intervención del desarrollador.
- Soporta la comunicación remota con otros sistemas. Asimismo pueden hacerse aplicaciones distribuidas para llevar la carga de los servicios implementados sin afectar a un único ente, así como también la comunicación con aplicaciones de terceros, soportados por varios protocolos de comunicación.

4.1.2. Desventajas

El lenguaje Java posee pocas desventajas. Sin embargo, dado que la máquina virtual de Java es un intérprete y redunda en una falta de rendimiento con relación a aplicaciones equivalentes escritas en código máquina nativo. Una respuesta a éste problema es el empleo de compiladores JIT (*Just In Time*). Un compilador JIT interactúa con la máquina virtual de Java para convertir el código de bytes en código nativo.

El poder reducir los problemas de acceso a memoria y liberación automática hacen de java un lenguaje poco apropiado para desarrollar aplicaciones de base como Sistemas Operativos. Sin embargo Java puede ser implementado en hardware con la tecnología JINI, que son redes adaptables y escalables.

4.2. Servlets y JSP

La comunidad de *Sun Microsystems*, encargado de realizar las especificaciones del lenguaje Java, creó en Junio de 1997 la tecnología de interacción HTTP llamada *API Java Servlets*, con sus respectivos contratos de comunicación con los servidores Web y contenedores de Servlets. Esta tecnología es la que actualmente se usan en todas las aplicaciones Web basadas en Java, y también lo será para esta herramienta.

Un Servlet es una clase Java que se ajusta a la API Java Servlet, un protocolo por el cual una clase Java puede responder a peticiones HTTP. Estas clases no están vinculadas a un protocolo específico de cliente-servidor, pero son utilizados a menudo con este protocolo. Así, un desarrollador de software puede utilizar un Servlet para añadir contenido dinámico a un servidor Web utilizando la plataforma Java. El contenido generado es comúnmente HTML, pero pueden ser otros datos como XML. Los Servlets son la contrapartida de Java a las tecnologías Web de contenido dinámico que no son de Java, como CGI, PHP, ASP.NET, etc. Los Servlets pueden mantener el estado en variables de sesión a través de muchas operaciones de servidor mediante el uso de cookies de HTTP, o la reescritura de URL.

JSP (*JavaServer Pages)* es una tecnología Java que permite generar contenido dinámico para Web, en forma de documentos HTML, XML o de otro tipo. Las páginas JSP permiten la utilización de código Java mediante scripts. Además, es posible utilizar algunas acciones JSP predefinidas mediante etiquetas. Estas etiquetas pueden ser enriquecidas mediante la utilización de Bibliotecas de Etiquetas (*TagLibs* o *Tag Libraries*) externas e incluso personalizadas.

JSP puede considerarse como una manera alternativa, y simplificada, de construir Servlets. Es por ello que una página JSP puede hacer todo lo que un Servlet puede hacer, y viceversa. Cada versión de la especificación de JSP está fuertemente vinculada a una versión en particular de la especificación de Servlets.

El funcionamiento general de la tecnología JSP es que el contenedor de las páginas JSP interpreta el código contenido en la página JSP para construir el código Java del Servlet a generar. Este Servlet será el que genere el documento (típicamente HTML) que se presentará en la pantalla del Navegador del usuario.

4.3. Tomcat

Tomcat (también llamado **Apache Tomcat**) funciona como un contenedor de Servlets desarrollado bajo el proyecto Jakarta en la empresa *Apache Software Foundation*. Tomcat es un servidor Web con soporte de Servlets y JSPs. Tomcat no es un servidor de aplicaciones, como *JBoss* o *JOnAS*. Incluye el compilador *Jasper*, que compila JSPs convirtiéndolas en Servlets. El motor de Servlets de Tomcat a menudo se presenta en combinación con el servidor Web Apache. En la imagen 6.3 puede observarse el procesamiento de las páginas JSP usando el Servidor Tomcat.

Dado que Tomcat fue escrito en Java, funciona en cualquier plataforma que disponga de la máquina virtual Java.

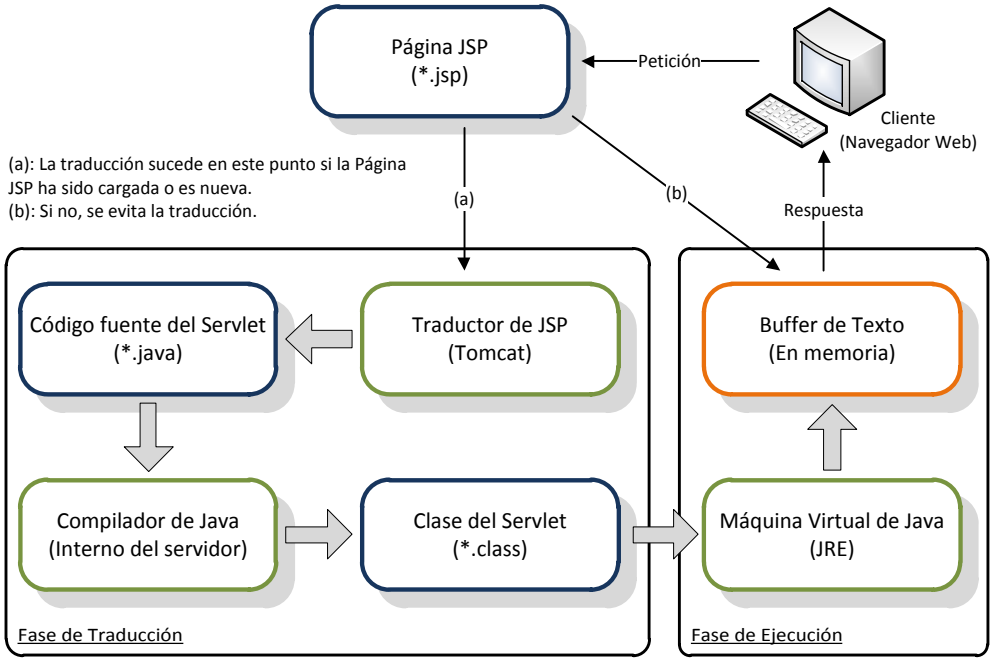


Imagen 6.3: Funcionamiento y Procesamiento de una Página JSP.
Fuente: Elaboración propia.

Tomcat es mantenido y desarrollado por miembros de la Apache Software Foundation y voluntarios independientes. Se puede disponer de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Apache Software Licence, la cual es una licencia de software libre.

4.4. Parser HTML

Un *Parser* HTML es un programa que recorre el código fuente de un documento HTML y permite extraer información del mismo de la manera que se estime oportuna para poder tratarla posteriormente. También pueden servir para manipular un documento HTML.

El *Parser* HTML es un componente clave para el desarrollo de la herramienta Web que se pretende implementar, ya que el proceso del análisis de las etiquetas HTML es el más relevante en el aplicativo, donde éstas se evalúan a fin de comprobar si poseen valores de accesibilidad.

4.4.1. Funcionamiento de un Parser

Los *Parsers* HTML están basados en los *Parsers* para el lenguaje de marcado XML (HTML es un lenguaje similar al lenguaje XML), pero están adaptados para poder recorrer código HTML en el que puede haber malformaciones tales como que haya una etiqueta de apertura de un bloque pero no de cierre, que se cierre un bloque antes que otro cuando debería ser al revés o cualquier otro tipo de error que pueda existir en un documento HTML.

Los *Parsers* cargan el archivo a escanear en memoria mediante alguna estructura de datos, una vez que el documento está cargado se puede acceder a él mediante el Modelo de Objetos del Documento, *Document Object Model* (DOM), es decir mediante las etiquetas de marcado.

Como se puede apreciar en la imagen 6.4, lo primero que se hace es leer el documento HTML, a continuación se pasa a través de un *Parser* HTML y éste lo transforma en una estructura de datos, como por ejemplo un árbol, desde el cual se puede acceder al contenido de cualquiera de las etiquetas del documento HTML.

4.4.2. Parsers XML

La mayor parte de los *Parsers* HTML están construidos sobre *Parsers* para el lenguaje XML por lo que es necesario explicar en que se basan este tipo de programas. Los *Parsers* XML utilizan alguna de las siguientes tecnologías para hacer su labor:

4.4.3. Document Object Model

Document Object Model, conocido frecuentemente por su acrónimo DOM, es un modelo de objetos estándar libre de plataforma y lenguaje, para representar el lenguaje HTML o cualquier otro lenguaje basado en XML como objetos que tienen sus propios métodos y propiedades. El responsable del DOM es el *World Wide Web Consortium* (W3C).

DOM se puede mover en cualquier dirección dentro del documento XML. El funcionamiento de DOM es similar al de la figura expuesta abajo.

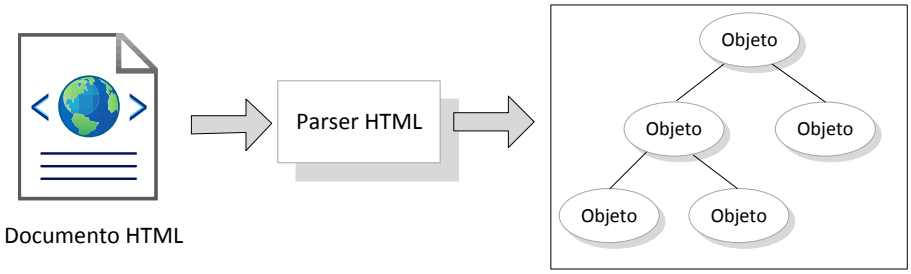


Imagen 6.4: Funcionamiento de un Parser DOM.
Fuente: Elaboración Propia.

4.4.3.1. Niveles del DOM

Las especificaciones del W3C-DOM se divide en varios niveles, cada uno de ellos contiene módulos requeridos y opcionales. Para que una aplicación soporte un determinado nivel, debe implementar todos los requisitos opcionales de ese nivel y los correspondientes de los niveles inferiores.

- Nivel 0: Las aplicaciones deben soportar un DOM intermedio, el cual existió antes de la creación del nivel 1 de DOM.
- Nivel 1: Navegación DOM del documento y manipulación del contenido. Los elementos específicos de HTML también se incluyen.
- Nivel 2: Soporte al espacio de nombres de XML, filtrando vistas y eventos.
- Nivel 3: Consiste en 6 especificaciones distintas:
    - XPath (ver Glosario) de DOM de nivel 3.
    - Núcleo de DOM de nivel 3.
    - Carga y almacenamiento de DOM de nivel 3.
    - Vistas y formato de DOM de nivel 3.
    - Requerimientos de DOM de nivel 3.
    - Validación de DOM de nivel 3.

4.4.4. Simple API for XML

Simple API for XML (SAX) es una interfaz de programación de aplicaciones o API (del inglés *Application Programming Interface*) para el acceso en serie a documentos XML. SAX provee un mecanismo para leer datos desde un documento XML, y se trata de la alternativa más popular a *Document Object Model* (DOM).

4.4.5. Procesamiento de XML con SAX

Un *Parser* basado en SAX funciona como un *Parser* de flujo con una API para el manejo de eventos. El usuario define los métodos que serán llamados cuando ocurra determinado evento durante el análisis del documento. Los eventos de SAX pueden ser:

- Nodos de texto.
- Nodos de elemento.
- Procesamiento de instrucciones.
- Comentarios.

Los eventos son lanzados cuando cada uno de estos elementos XML son encontrados, y también cuando es encontrado el final de cada uno de ellos. El escaneo de SAX es unidireccional, por lo que no puede ser releído un elemento sin tener que volver a comenzar el escaneo. En la siguiente imagen se observa el funcionamiento de un *Parser* SAX.

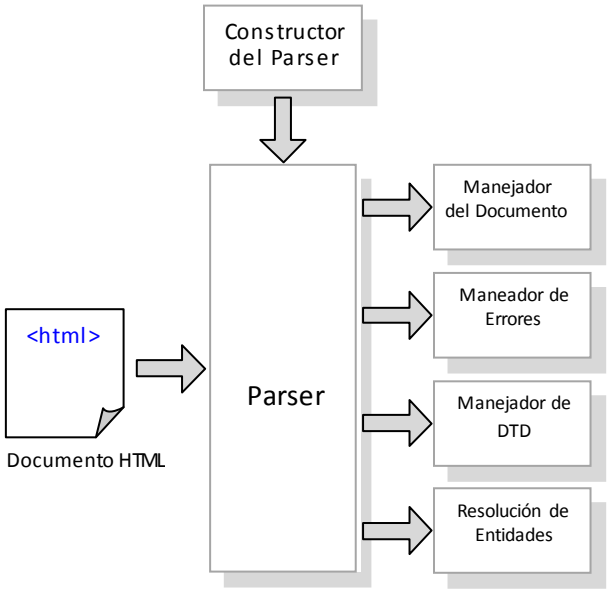


Imagen 6.5: Funcionamiento de un Parser SAX.
Fuente: Elaboración Propia.

4.4.6. Parsers HTML de Software Libre para Java

Existe gran cantidad de *Parsers* bajo la licencia de software libre para poder usarlos en Java. En esta sección se mostrarán los más importantes de los mismos, así como sus principales características y así poder escoger entro uno u otro para poder usarlo y así realizar la verificación de accesibilidad Web propuestas en la herramienta.

Se presenta a continuación una lista de *Parsers* que fueron evaluados dadas sus características, a fin de encontrar el más conveniente para ser utilizado durante el desarrollo de la herramienta.

4.4.6.1. JDOM

JDOM es una librería de código fuente para manipulaciones de datos XML optimizada para Java. A pesar de su similitud con DOM del consorcio *World Wide Web* (W3C), es una alternativa

como documento para modelado de objetos que no está incluido en DOM. La principal diferencia es que mientras que DOM fue creado para ser un lenguaje neutral e inicialmente usado para manipulación de páginas HTML con *JavaScript*, JDOM se creó específicamente para usarse con Java y por lo tanto beneficiarse de las características de Java, incluyendo sobrecarga de métodos, colecciones, etc. (Hunter, 2009).

Aunque JDOM parezca un acrónimo de *Java Document Object Model* (Java DOM), esto no es así, siendo desmentido por el propio proyecto de JDOM. JDOM se integra con *Document Object Model* (DOM) y también *Simple API for XML* (SAX), y soporta XPath y XSLT.

4.4.6.2. NekoHTML

NekoHTML es un escáner simple de código HTML y balanceador de etiquetas que permite analizar documentos HTML y acceder a la información usando interfaces de XML estándar. El programa puede explorar documentos HTML y obviar muchas de las equivocaciones que se cometen en la escritura de código HTML (Clark, 2007).

NekoHTML está escrito usando la Interfaz Nativa de Xerces (XNI) que está incluida en la implementación de Xerces2. Esto habilita a usar NekoHTML con cualquiera de las herramientas XNI existentes sin necesidad de variar su código.

Limitaciones

- Existe documentos que NekoHTML Parser no genera un flujo de eventos de XML bien formado, por ejemplo documentos con múltiples etiquetas `<html>`.
- El código añadido al núcleo DOM en Xerces-J versión 2.0.1 introduce un error en la implementación de DOM HTML basado en él. El problema afecta a los usuarios de NekoHTML que usen el *Parser* con Xerces-J versión 2.0.1.

4.4.6.3. HTML Parser

HTML Parser es una biblioteca de Java usada para analizar el código HTML de una manera lineal o jerarquizada. Su uso está optimizado sobre todo para la trasformación o extracción de código HTML y ofrece además una serie de filtros para poder filtrar la información y adición de nuevas etiquetas personalizadas. Se trata de un paquete rápido y robusto (Oswalt, 2006).

En cuanto a la extracción de datos del código HTML -que es lo que se necesita para este proyecto- es capaz de lo siguiente:

- Extracción de texto, por ejemplo para servir como entrada de motores de búsqueda de bases de datos.
- Extracción de enlaces, para poder acceder a las páginas Web que apuntan o conseguir los correos electrónicos que contiene.
- Extracción de recursos como imagen o sonido
- Chequeo de enlaces, averiguando que enlaces son válidos

4.4.6.4. Jericho HTML Parser

Jericho HTML Parser es una biblioteca Java simple y potente para analizar y modificar documentos HTML, así como ciertas etiquetas del lado del servidor. Es capaz de reproducir partes desconocidas o inválidas del código. Es una librería de código abierto bajo la licencia *Eclipse Public License* (EPL) y *GNU Lesser General Public License* (LGPL) (Jericho, 2009).

Esta librería se distingue de otras herramientas de análisis de código HTML por las siguientes cualidades:

- La presencia de HTML mal formado no interfiere en el análisis del resto del documento, lo que hace a esta librería ideal para el análisis en el mundo real, cosa que no cumplen otras herramientas.
- Etiquetas de servidor como ASP, JSP, PSP, PHP son reconocidas explícitamente por el *Parser*.
- No es un *Parser* basado en árbol (como DOM) o eventos (como SAX), pero utiliza una combinación de búsqueda simple de texto, reconocimiento eficiente de etiquetas y una caché de posición de etiquetas. El texto del documento fuente es cargado primero en memoria y entonces solamente se buscan los segmentos destacados por los caracteres relevantes para operación de búsqueda.
- Comparado con un programa de análisis basado en árbol tal como DOM, el uso de memoria puede ser radicalmente menor si lo que se necesita analizar son secciones muy pequeñas de código.
- Comparado con un *Parser* basado en eventos como SAX, su interfaz es más intuitiva y de más alto nivel.
- Las posiciones de comienzo y fin en el documento fuente de todas las partes analizadas son accesibles.
- El número de fila y columna de cada posición en el documento fuente es fácilmente accesible.

- Tipo personalizados de etiquetas pueden ser fácilmente definidas y registradas para poder ser reconocidas por el *Parser*.
- Funcionalidad para extraer todo el texto del código HTML para poder alimentar motores de búsqueda por ejemplo.

4.4.6.5. JTidy

JTidy es una interfaz de Java para *HTML Tidy* y un verificador de sintaxis de HTML. JTidy puede ser usado como una herramienta para limpiar HTML defectuoso y para formatear el código HTML. También provee de una interfaz DOM al documento que está siendo procesado. Sus características son (Sandor, 2006):

- Capacidad de eliminar errores en el código HTML tales como:
  - Ausencia de etiqueta de cierre de un elemento.
  - Etiquetas de cierre escritas en el orden equivocado.
  - Recuperación de un mezclado erróneo de etiquetas.
  - Adición del caracter "/" perdidas en las etiquetas de cierre.
  - Perfeccionamiento de listas añadiendo etiquetas olvidadas.
- Indenta (ver Glosario) el código HTML correctamente.
- Posibilidad de leer varios juegos de caracteres internacionales como US ASCII, ISO Latin-1, UTF-8 y la familia ISO-2022.
- Limpieza del marcado del código HTML.
- Posibilidad de añadir nuevas etiquetas.
- Soporte limitado a ASP, PHP y JSTE.
- Soporte limitado a XML.

4.4.6.6. TagSoup

TagSoup es un *Parser* basado en SAX y escrito en Java y es capaz de analizar HTML mal formado. Proporciona una interfaz de SAX que permite a las herramientas para XML estándar sean capaces de leer HTML mal formado. Sus características son las siguientes (Dodds, 2007):

- TagSoup está pensado como un *Parser* HTML, no como un programa capaz de limpiar permanentemente HTML mal formado, si no que escanea este código mal formado durante la ejecución.
- No es capaz de convertir la presentación en HTML a CSS.
- No depende de ninguna otra infraestructura aparte de SAX.
- TagSoup no funciona correctamente con las versiones 5 y 6 de Java.

4.4.6.7. HotSAX

HotSAX es un *Parser* basado en SAX versión 2 para el análisis de XML, HTML y XHTML. Puede ser usado por agentes Web simples o *Spiders* (ver Glosario). Es similar al *Parser* para XML de Apache, Xerces, pero con eventos para código HTML mal formado (Howland, 2006).

Esta herramienta está diseñada para ayudar a construir otras herramientas útiles como *Spiders*, analizadores de páginas y conversores de HTML a otros formatos. Se puede insertar HotSAX en grandes aplicaciones como sistemas de gestión.

4.4.6.8. HtmlCleaner

HtmlCleaner es un *Parser* HTML de código abierto escrito en Java. Dado que los documentos HTML usualmente están mal formados debido a errores de programación, existen herramientas que no pueden leerlo a no ser que se arreglen estos fallos. Para ello existe esta herramienta que es capaz de leer el documento y arreglar los fallos existentes produciendo XML estándar (Nikic, 2006).

HtmlCleaner implementa el conjunto de etiquetas estándar de HTML y reglas para realizar un balanceado correcto.

4.4.6.9. Java Mozilla Html Parser

Mozilla Html Parser es un *Parser* escrito en Java y basado en el *Parser* de Mozilla. Actúa como puente desde clases de Java a clases de Mozilla y obtiene como salida un DOM desde una entrada de código HTML sucio (Serfaty, 2007).

La mayor limitación conocida de *Mozilla HTML Parser* está relacionada con su funcionamiento, ya que el programa de análisis serializa las peticiones. En el momento que el programa de análisis está funcionando y recibe una petición, la analiza y la pone en una cola la respuesta al solicitante.