UNIVERSIDAD POLITECNICA DE VALENCIA

UNIVERSIDAD POLITÉCNICA DE VALENCIA

DESARROLLO DE UN COMPLEMENTO PARA FIREFOX DE FILTRADO Y RESALTADO DE PÁGINAS WEBS

AUTORA: TATIANA TOMÁS BALSEBRE

DIRECTOR: JOSEP FRANCESC SILVA GALIANA

VALENCIA, OCTUBRE 2008

1. PRESENTACIÓN

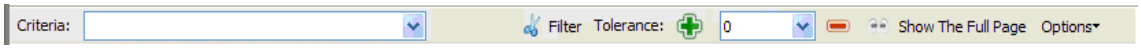
1.1. OBJETIVOS

Internet ha causado un gran impacto en el trabajo, el ocio y el conocimiento a nivel mundial. La importancia que ha adquirido la red de redes en nuestras vidas es enorme, debido en gran medida a la cantidad de información de todo tipo y acerca de todos los temas que se encuentra en la misma. Gracias a Internet, millones de personas tienen acceso fácil e inmediato a dicha información. Internet ha permitido una descentralización repentina y extrema de la información y de los datos.

En los primeros años de vida de Internet, encontrar la información que uno estaba buscando era una tarea relativamente rápida, ya que, el volumen de información que Internet contenía era relativamente pequeño. Actualmente, sin la existencia de motores de búsqueda, que busquen y localicen la información que nos interesa dentro de Internet, esa tarea sería inabordable. Sin embargo, no solo está creciendo la cantidad de información disponible dentro de Internet, sino, que cada más, existen páginas Web cuya información es extensa, y la tarea de encontrar la información que buscamos dentro de ellas se está convirtiendo en una tarea tediosa. Debido a esto, es necesario completar el uso de algún motor de búsqueda, tipo Google, con otro tipo de herramientas que ayuden a localizar y filtrar información útil dentro de una determinada página Web.

Este proyecto parte de uno existente realizado durante el curso 2006/2007. En aquel proyecto, se desarrolló un primer prototipo que no llegó a alcanzar el grado de madurez suficiente como para ser distribuido públicamente. Los nuevos objetivos de este proyecto consisten básicamente en la ampliación de su funcionalidad, la corrección de pequeños errores detectados, la mejora de su interfaz y su eficiencia, y la migración del prototipo a la versión 3 del navegador Firefox, ya que, la versión original estaba implementada para las versiones 2.0.0.* de ese navegador. Cada una de estas características, tanto su implementación como las decisiones de diseño tomadas serán explicadas detalladamente en esta memoria.

La funcionalidad y el aspecto actual de la barra antes de abordar este proyecto es la siguiente:



- *Botón 'Filter'*: es el botón principal de la aplicación, encargado de realizar el filtrado de la página, dejando visible únicamente aquella información que nos interesa, es decir, aquella que cumple con el criterio de búsqueda establecido.

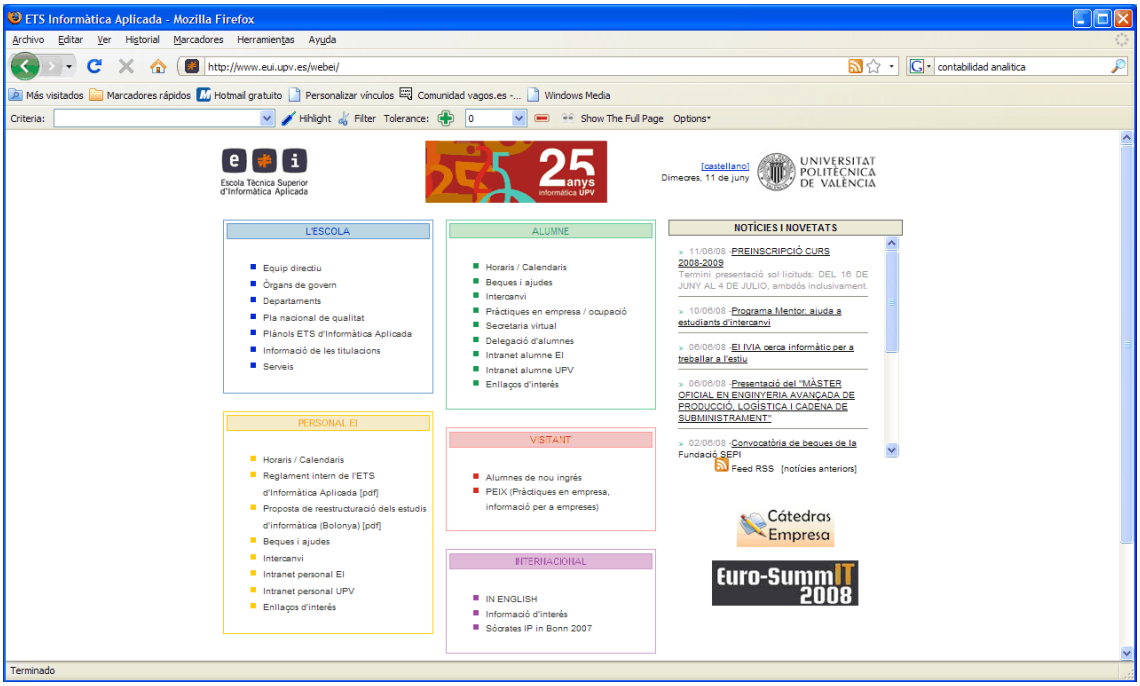


Figura 1. Página original

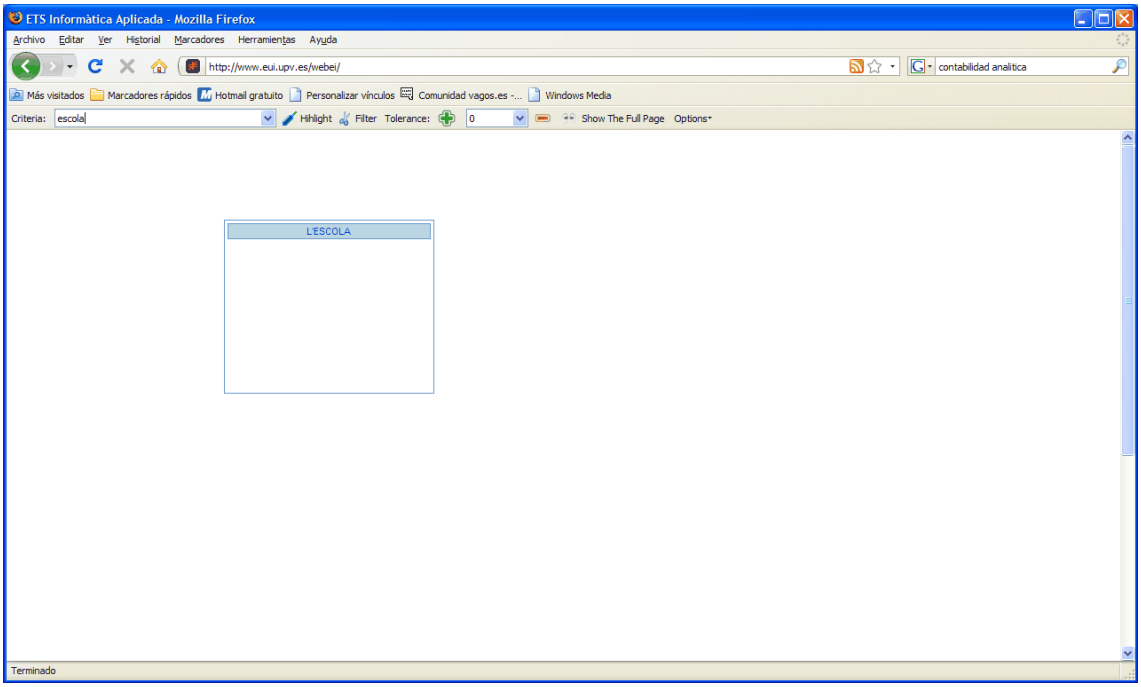


Figura 2. Página filtrada según el criterio de búsqueda 'escola' con nivel de tolerancia 0

- *Botón 'Tolerance'*: la tolerancia sirve para decidir que cantidad de información desea ver el usuario cuando realiza el filtrado de la página. A mayor nivel de tolerancia mayor será la cantidad de información que se visualice, teniendo en cuenta siempre que la máxima tolerancia posible es visualizar la página entera, y a menor nivel de tolerancia, como mínimo siempre será 0, menor cantidad de información.

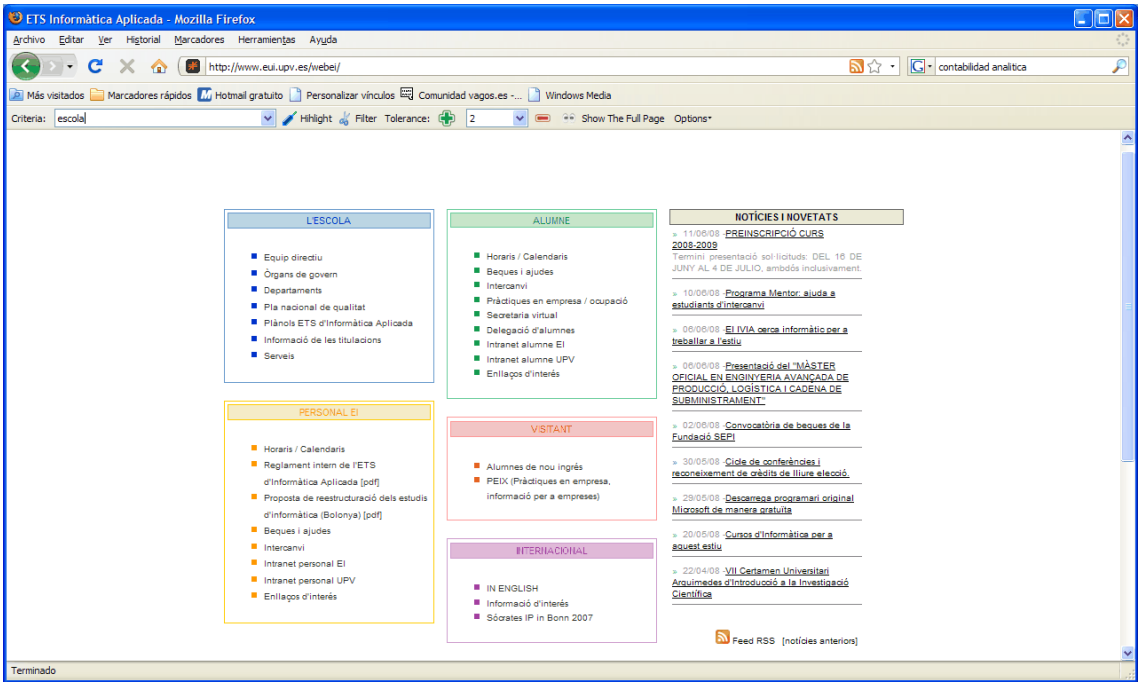


Figura 3. Mismo criterio de búsqueda pero con nivel de tolerancia 2

- Botón 'Show the full Page': se encarga de volver a mostrar la página original completa, eliminando el criterio de búsqueda aplicado anteriormente.

- Botón 'Options': este botón de la barra de herramientas se encarga de desplegar un menú con tres opciones: opciones, ayuda y acerca de. El usuario al seleccionar una de ellas se le abrirá el menú correspondiente.

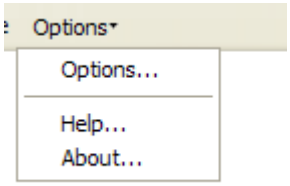


Figura 4. Menú que se despliega al pulsar sobre opciones

➢ La opción de menú 'Options' abre una ventana, en la que el usuario podrá definir algunas preferencias, las cuales se aplicarán posteriormente al filtrado. Dichas preferencias y que definen se explican en el punto de 'Especificación de Requisitos'.

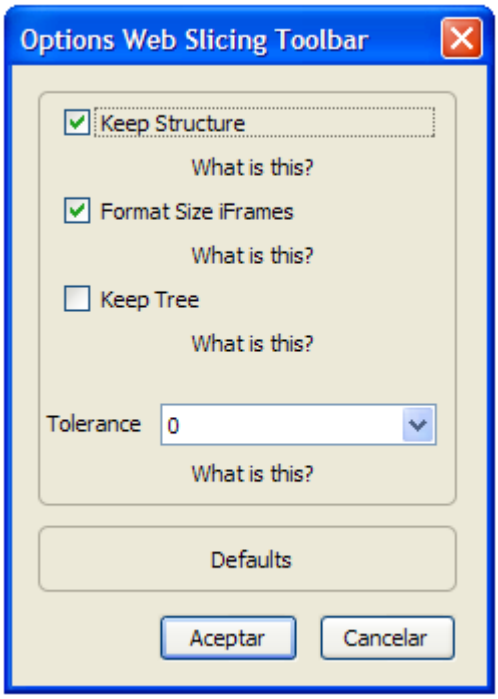


Figura 5. Ventana 'Options'

- La opción de menú 'Help' abre una ventana donde se muestra un manual de uso de la barra de búsqueda.

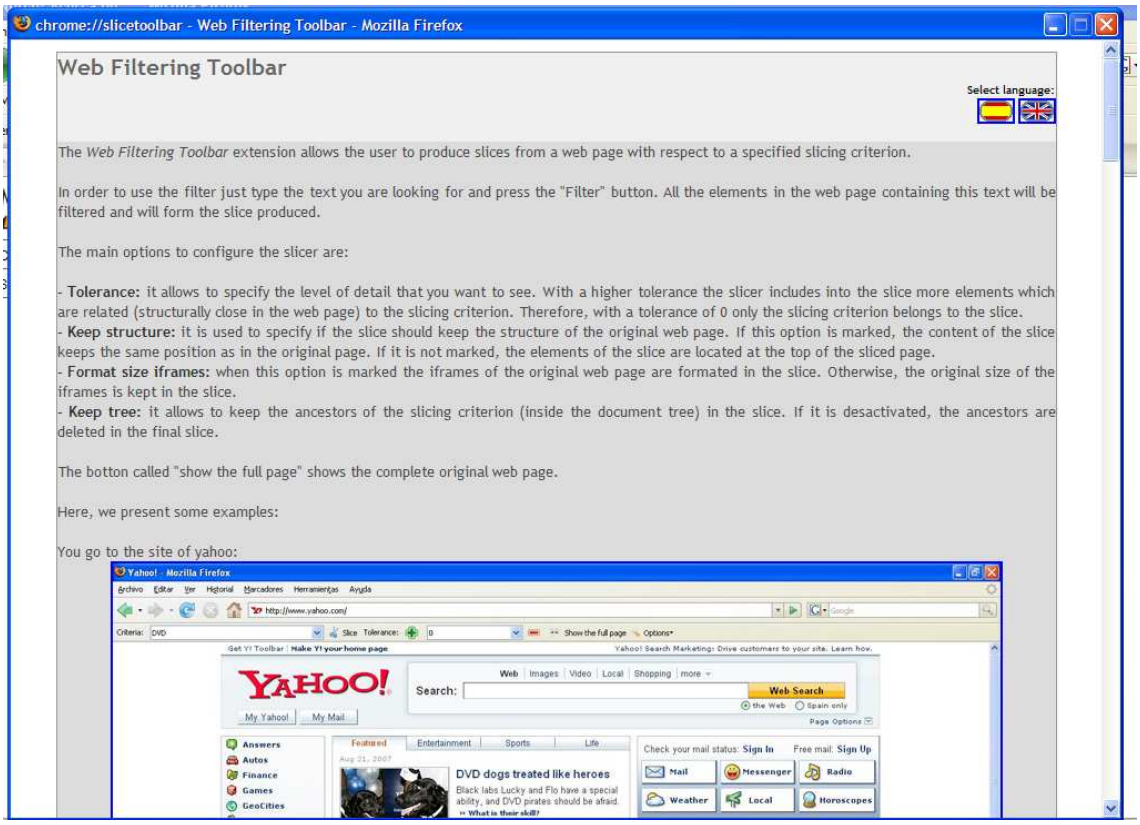
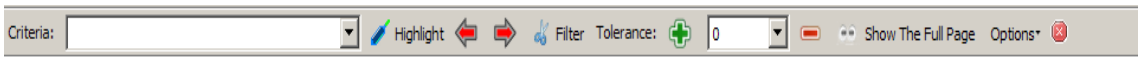


Figura 6. Ventana con el manual de ayuda

- La opción de menú 'About' abre una ventana con información de la versión de la barra y de los autores de la misma.

Figura 7. Ventana con información de la barra de búsqueda

La funcionalidad y el aspecto actual de la barra después de finalizar este proyecto es el siguiente:



- *Botón 'Resaltar':* botón encargado de resaltar en rojo todas las coincidencias que aparezcan de la palabra dentro de la página Web.

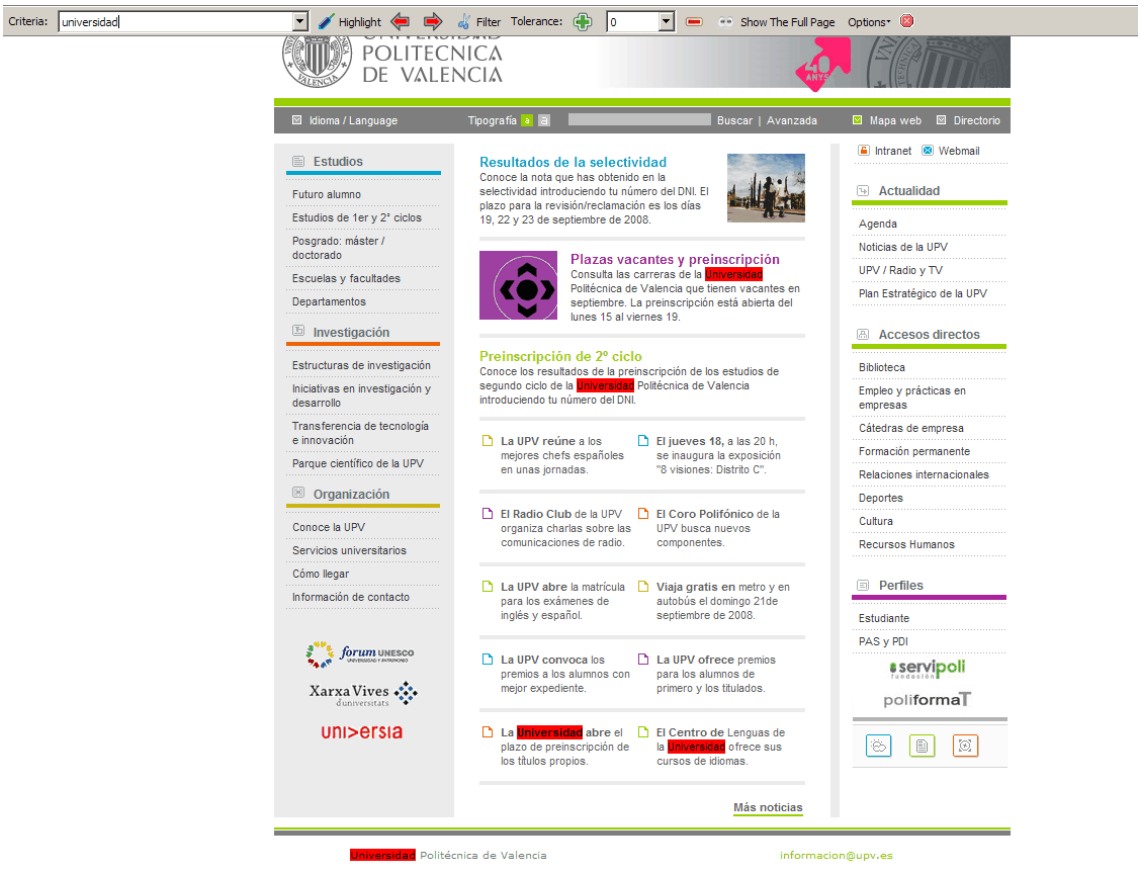


Figura 8. Página original con todas las coincidencias de la palabra 'Universidad' resaltadas

- *Botón 'Anterior':* botón encargado de ir resaltando en amarillo, de una en una, todas las coincidencias de la palabra, empezando por el final de la página y finalizando por el principio.

Figura 9. Se ha pulsado anterior una vez

Figura 10. Se ha pulsado anterior dos veces

- *Botón 'Siguiente':* botón encargado de ir resaltando en amarillo, de una en una, todas las coincidencias de la palabra, empezando por el principio de la página y finalizando por el final.

Figura 11. Se ha pulsado siguiente una sola vez

Figura 12. Se ha pulsado siguiente dos veces

- *Botón 'Filter':* misma funcionalidad que en la versión anterior.
- *Botón 'Tolerance':* misma funcionalidad que en la versión anterior.
- *Botón 'Show the full page':* misma funcionalidad que la versión anterior.
- *Botón 'Options':* misma funcionalidad que en la versión anterior.
- Resaltar y filtrar a la vez. Es decir, una vez aplicado un filtrado sobre la página Web, es posible resaltar todas las apariciones del mismo criterio de búsqueda o de uno diferente y viceversa (es decir primero resaltar y luego filtrar).

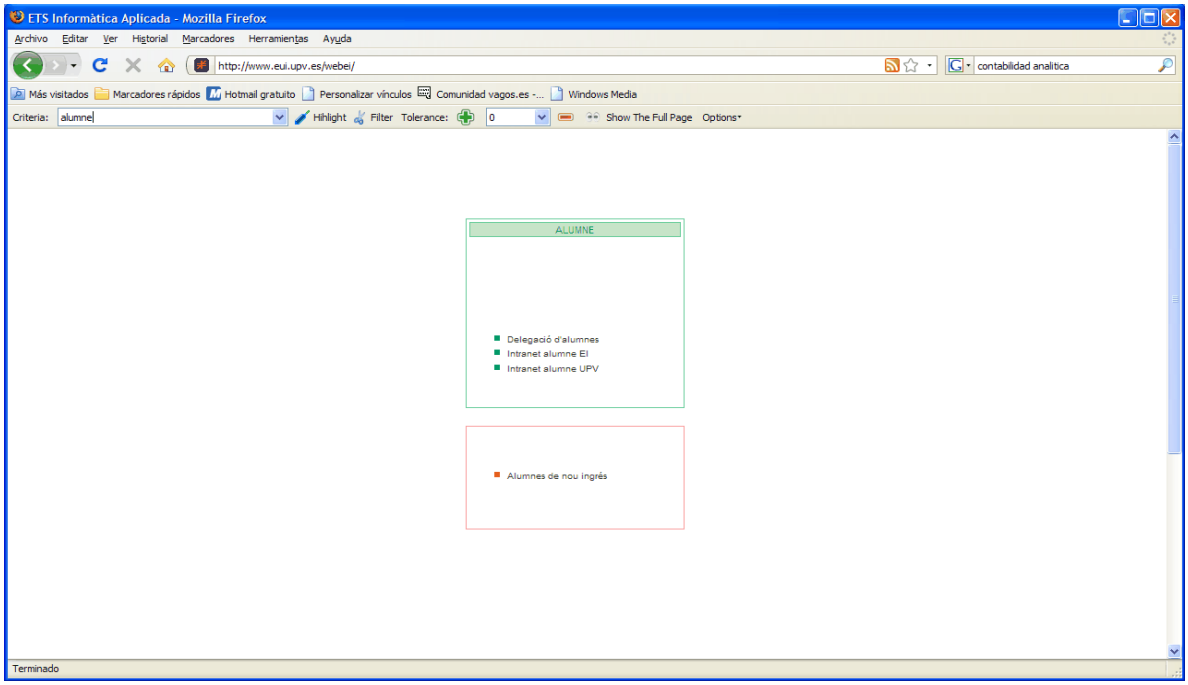


Figura 13. Se ha filtrado por la palabra alumno

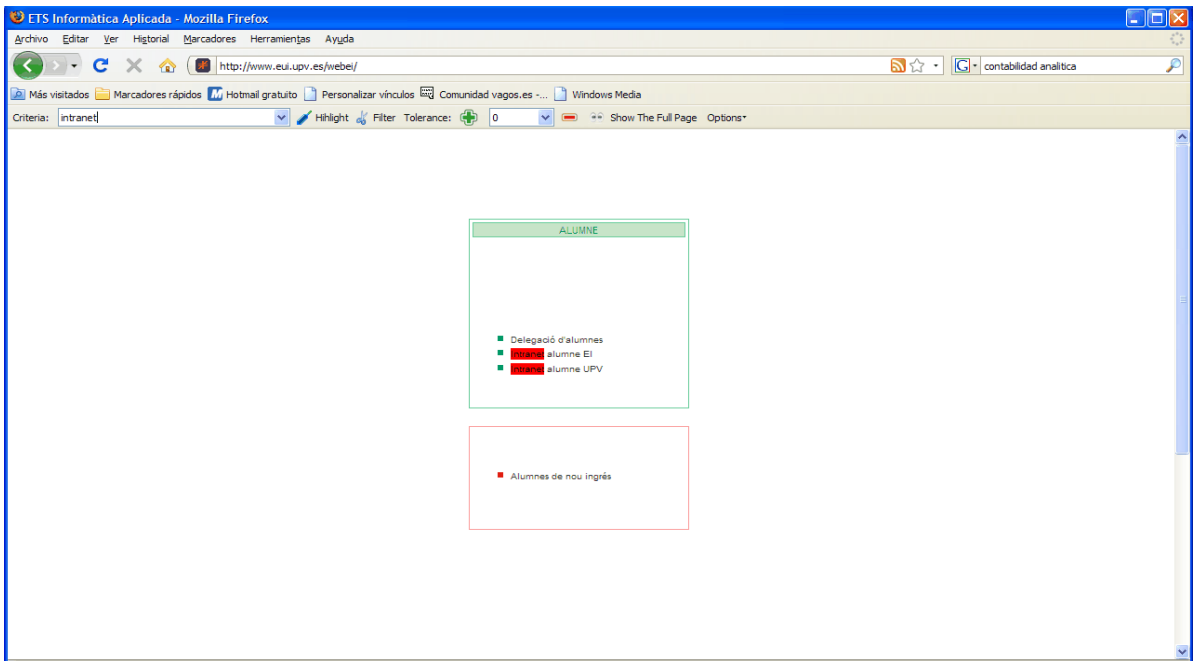


Figura 14. Se han resaltado todas las coincidencias de la palabra Intranet manteniendo el filtrado anterior

- *Botón 'Cerrar':* botón encargado de ocultar la barra. Para volver a mostrala basta con pulsar la combinación de teclas 'Ctrl+Y'.

1.2. ESTRUCTURA DEL DOCUMENTO

Para cubrir los diferentes puntos del proyecto, este documento se ha estructurado en cinco capítulos y tres anexos.

Un primer capítulo, el cual contiene, además del apartado anterior y el actual, información sobre la historia y evolución de los navegadores, concentrándose en otro apartado en la historia del navegador Firefox. Posteriormente se hace una pequeña presentación de la organización W3C y de los lenguajes utilizados para el desarrollo de este proyecto. Este capítulo finaliza explicando los dos estándares DOM y CSS que ha tenido que respetar este proyecto.

En el segundo capítulo se recoge la especificación de requisitos del proyecto siguiendo en todo momento el estándar IEEE 830.

En el tercer capítulo se detalla todo el análisis y diseño del proyecto, tanto el análisis anterior, del que parte este proyecto, como el nuevo realizado para la funcionalidad nueva de la aplicación.

En el cuarto capítulo se recoge toda la implementación del proyecto, desglosada por funciones, y las herramientas utilizadas para el desarrollo del mismo.

En el capítulo cinco se recogen todas las referencias a documentos, libros y páginas Web utilizadas para desarrollar el proyecto y documentar la memoria.

En el primero de los anexos se recoge el código fuente de las funciones, que por similitud a otras, no se han explicado en el punto cuatro.

En el segundo anexo se adjunta el código de todos los ficheros XUL de la aplicación.

En el tercero se explica que es la herramienta WebDeveloper y el código que ha sido necesario utilizar para el correcto funcionamiento del proyecto.

1.3. HISTORIA DE LOS NAVEGADORES

Antes de recapitular un poco en la historia de los navegadores, vamos a proporcionar una pequeña definición de lo que es un navegador Web.

Un navegador Web es una aplicación software que permite al usuario recuperar y visualizar documentos desde servidores Web de todo el mundo a través de Internet. Muchas de esas páginas Web poseen hipervínculos que enlazan texto o imágenes a otro documento. El seguimiento de una página a otra se llama navegación, y de ahí se origina el nombre de navegador. [Wikipedia, 2008]

La historia de los navegadores comienza en Suiza a principios de la década de los noventa. Por aquél entonces, en los laboratorios CERN (Centro Europeo para la Investigación Nuclear) de Ginebra, se estaba inventando la World Wide Web, como sistema para compartir documentación a lo largo del mundo de manera rápida y asequible, es decir, se le estaba dando forma a la estructura actual de Internet. Al mismo tiempo que inventaban la W3, los científicos del CERN aprovecharon para desarrollar un lenguaje que pudiera ser comprensible tanto para los humanos como para las máquinas, dicho lenguaje fue HTML. Una vez consiguieron crear un espacio digital y un lenguaje con el que dialogar con él, les faltaba crear alguna herramienta que les permitiera visualizar los documentos ubicados en distintos lugares del espacio digital. Dicha herramienta, sería el navegador.

El primer navegador que se creó y que estuvo en condiciones de ser utilizado fue Mosaic, lanzado en Abril de 1993 por Marc Andreessen y Eric Bina en el NCSA (National Center for Supercomputing Applications) de Illinois (EE.UU), donde se prepararon versiones para Windows y Macintosh. [Historia]

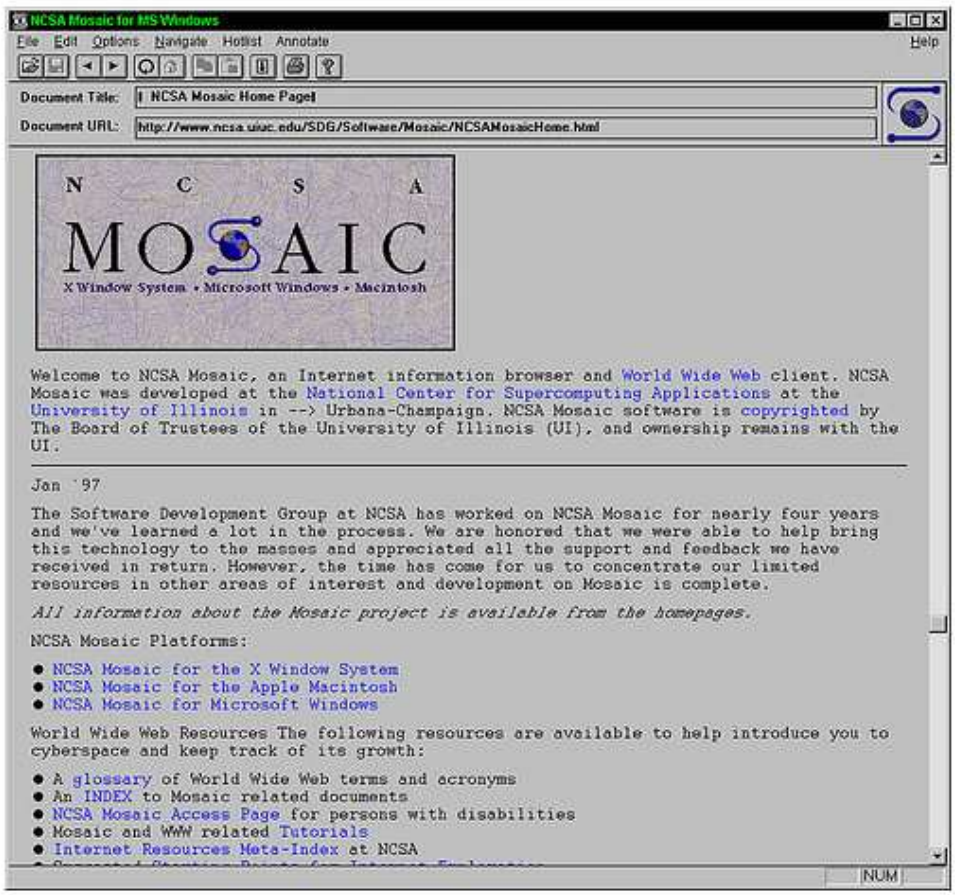


Figura 15. Aspecto del navegador Mosaic

Al poco de su lanzamiento, Marc Andressen se fue del NCSA y fundó Mosaic Communications Corporation con Jim Clark. El nombre de esta empresa duró muy poco tiempo, cambiándolo por el de Netscape Communications Corporation. Bajo este nombre crearon el navegador Netscape Navigator. Netscape fue el primer navegador comercial y el que consiguió sacar la W3 de los entornos universitarios y tecnológicos a la calle, convirtiendo Internet en un fenómeno de masas. Esta revolución se debe a que Netscape introdujo novedades. La primera de ellas fue la habilidad de mostrar información de manera inmediata, basándose en el sistema gradual de primero el texto y después las imágenes. La segunda de ellas fue permitir que el usuario se descargase una versión de prueba gratuita, probarla y si le interesaba pagar por ella. Netscape consiguió convertirse en el estándar para el W3C y la ECMA (European Computer Manufacturers Association).

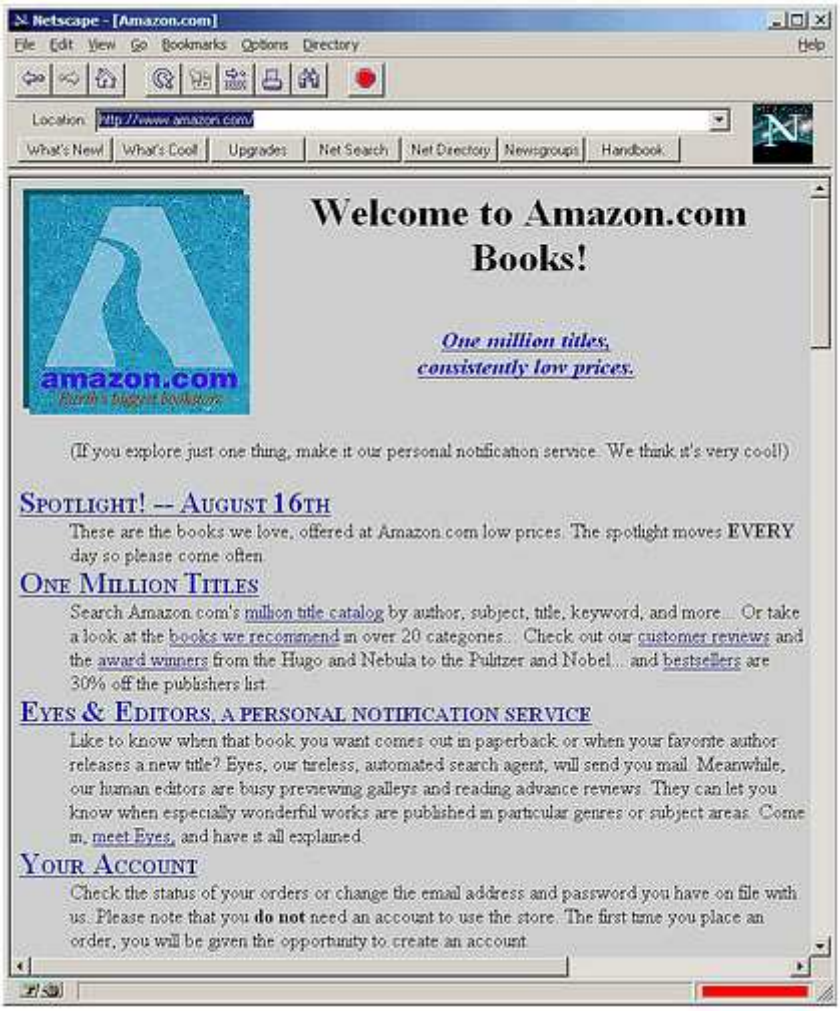


Figura 16. Aspecto del navegador Netscape Navigator versión 1.0

En 1995, cuando Microsoft compró Spyglass Mosaic y lanzó la primera versión de su navegador Internet Explorer, empezó lo que se conoció posteriormente como la 'Guerra de los Navegadores'. Microsoft consiguió 'ganar' dicha guerra rescribiendo la tercera versión de su navegador desde cero e integrando la cuarta en el sistema operativo Windows 98. Desde entonces IE ha alcanzado el liderazgo en el mercado de los navegadores, debido en gran medida, a que viene preinstalado con el sistema operativo Windows y no puede desinstalarse debido al navegador de archivos de Windows.

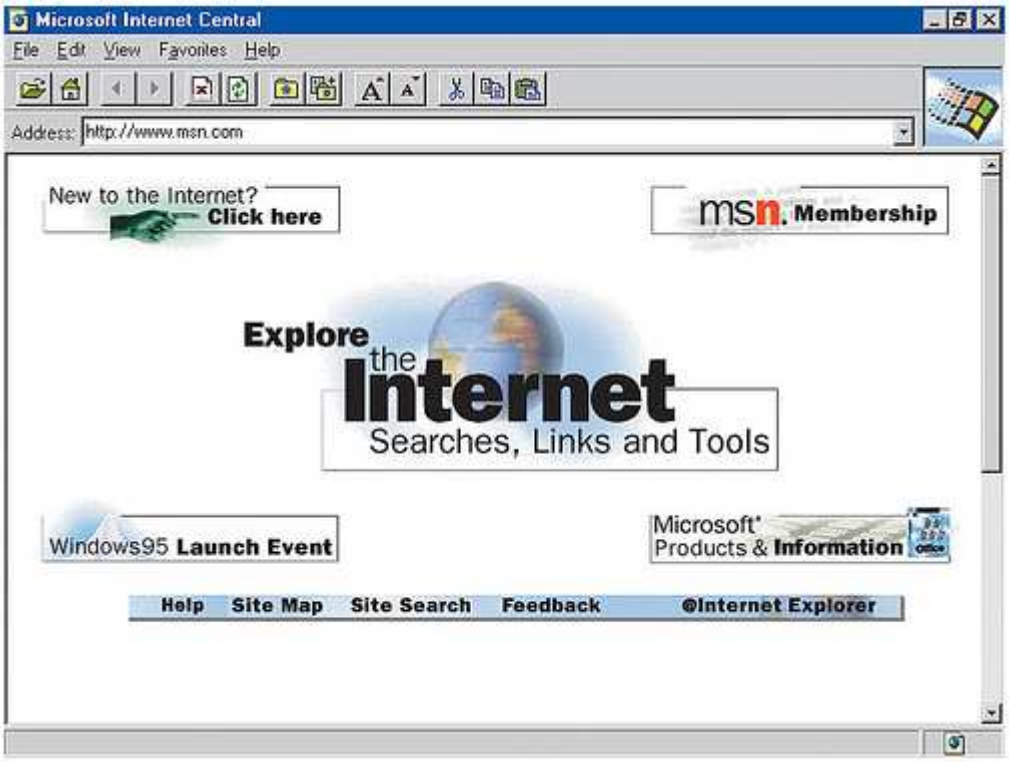


Figura 17. Aspecto de la primera versión de Internet Explorer

Actualmente el navegador más utilizado en el mundo es Internet Explorer en todas sus versiones. Por detrás de este se encuentra Mozilla Firefox, que desde su creación en 2004, se esta popularizando cada vez más y es un duro competidor de IE. También existen otros navegadores minoritarios como son Safari, Netscape y Opera, los cuales no superan el 2% de uso en el mercado. [Historia Navegadores].

Figura 18. Aspecto de la primera versión de Firefox

1.4. NAVEGADOR FIREFOX

Mozilla Firefox es un navegador con interfaz gráfica de usuario desarrollado por la Corporación Mozilla. En la primavera de 2002 aparece una rama experimental del proyecto Mozilla que condujeron David Hyatt y Blake Ross, que pretendía mejorar su código e interfaz. El nombre original del proyecto fue Mozilla Browser.
Comenzó como una parte del navegador de la Mozilla Application Suite y se ha convertido en el principal foco de desarrollo de la Fundación Mozilla. Antes de la publicación de la versión 1.0 en Noviembre de 2004, ya había llamado la atención de ciertos medios de comunicación. Con más de 25 millones de descargas de esta versión, propiciaron el desarrollo y publicación en Octubre de 2005 de la versión 1.5, la cual incorporaba numerosas ventajas como la compatibilidad con los estándares.

Dicho navegador ha sufrido muchos cambios de nombre a lo largo de su corta vida. Así, en Septiembre de 2002, cuando estaba lo bastante desarrollado como para poder ser utilizado se publicó bajo en nombre de "Phoenix". Pero este nombre solo duro hasta Abril de 2003, cuando por razones legales tubo que ser cambiado, el nombre escogido en esta ocasión fue "Firebird".
Este nombre era polémico porque coincidía con el nombre de otro proyecto de código abierto, el de la base de datos Firebird, por lo que, a finales de Abril de 2003 se publica con el nombre de Mozilla Firebird, para evitar confusiones.
Sin embargo este nombre no llego a cuajar en la comunidad de internautas, y finalmente, en febrero de 2004 la Fundación Mozilla decide rebautizarlo, y esta vez definitivamente, como Mozilla Firefox, patentando dicho nombre para evitar problemas posteriores.

El éxito y el incremento del uso de Firefox, esta originando la publicación de nuevas versiones, con más funcionalidades y más potentes. Así, en Octubre de 2006 se publicó la versión 2.0, conocida, en los entornos de desarrollo como 'Bon Echo'.

La versión 3.0 del navegador se publico el día 17 de junio de 2008. Desde la Web oficial de la organización Mozilla [Mozilla] se hizo un llamamiento a toda la comunidad de Internet invitando a que todos los usuarios interesados en este navegador se descargaran e instalaran esta versión ese día, ya que, la organización se había fijado como objetivo entrar en el libro Guinness de los Records consiguiendo el mayor número de descargas en un mismo día de la historia. El día 17 de Junio de 2008 fue bautizado por la Fundación Mozilla como 'Download Day', 'El día de las descargas' en castellano. Cabe destacar que lo consiguieron con más de 8 millones de descargas, la cifra oficial a la hora del cierre del susodicho día fue de 8,1 millones de descargas. España fue el cuarto país en número de descargas, alrededor de unas 300.000. Lideran la lista Estados Unidos con 2,5 millones, Alemania con 600.000 y Japón con 400.000.
Esta versión viene con un mejorado rendimiento, mejoras en la seguridad, protegiendo contra el 'phising' y el software malicioso, en la productividad y una mayor potencia de personalización.
Para finalizar comentar que la última versión del navegador Firefox fue conocida en los entornos de desarrollo como 'Grand Paradiso'. Este proyecto se tuvo que desarrollar sobre la versión de pruebas beta 5 de dicha versión, por no disponer de la versión final, pero se garantiza la total compatibilidad con la versión finalmente publicada.

Dejar constancia de que no funcionará en Windows 98, Windows Me y Windows NT 4.9, ya que, Microsoft finalizará, en breve, la asistencia técnica para dichas plataformas de su sistema operativo.

1.5. W3C

El W3C (Consorcio World Wide Web) es un consorcio internacional donde las organizaciones miembro trabajan para desarrollar estándares Web. La misión de este consorcio es orientar la Web hacia su máximo potencial a través del desarrollo de protocolos que aseguren el crecimiento futuro de la misma. Esto se consigue a través de la creación de Estándares Web abiertos, los cuales aseguran la interoperabilidad Web entre los navegadores de los diferentes fabricantes. Otro factor que ayuda a que la Web alcance su máximo potencial es que las tecnologías Web más importantes deben ser compatibles entre sí y permitir que cualquier hardware y software, utilizado para acceder a la Web, funcione conjuntamente.
Pero la W3C no solo se dedica a la creación de estándares, sino que está involucrada en tareas de educación y difusión con el objetivo de conseguir que la Web esté al alcance de todos, independientemente del hardware, software, infraestructura de red utilizados e independientemente del idioma, cultura, localización geográfica o habilidad física o mental de la persona.

La W3C se fundó en Octubre de 1994 por Tim Berners-Lee en el Laboratorio de Ciencias Informáticas del Instituto de Tecnología de Massachussets, en colaboración con el CERN, del DARPA y de la Comisión Europea. Para que el lector se haga una idea de la importancia de este consorcio, se van a detallar los principales hitos conseguidos por él mismo.

- **Octubre 1996:** se publicó la primera recomendación del W3C, "Gráficos de Red Portátiles (PNG) 1.0". Dicha recomendación se desarrolló para proporcionar una alternativa multiplataforma a los formatos de gráficos, los cuales habían causado problemas con las patentes.
- **Diciembre 1996:** se separo el contenido de la estructura de una página Web y se publica el CSS de Nivel 1. El CSS de Nivel 2 se publicó en 1998 e incluía más funciones. Para más información vaya al punto donde se habla de CSS [CSS].
- **Febrero 1997:** se lanza la "Iniciativa de accesibilidad Web". Está trata de alcanzar la accesibilidad de la Web a través de cuatro áreas de trabajo: tecnología, herramientas, educación y difusión, e investigación y desarrollo.
- **Diciembre 1997:** el HTML 4.0 añade tablas, scripting, CSS, internacionalización y accesibilidad a la publicación en la Web.
- **Febrero 1998:** apareció el XML 1.0 que promueve la interoperabilidad y el etiquetado específico de dominio. Pronto se convertiría en la base de la Web y serviría como base para el desarrollo de diferentes estándares.
- **Agosto 2000:** aparecen los "Gráficos Vectoriales Escalables (SVG) 1.0", los cuales enriquecieron los gráficos Web. Ha sido la base para las aplicaciones de móviles de nueva generación.
- **Mayo 2001:** el Esquema XML se convierte en la pieza esencial para que XML alcance su máximo potencial. Este esquema proporciona un estándar para crear vocabularios XML que permiten construir aplicaciones más potentes y versátiles.
- **Enero 2002:** se lanza la "Actividad de Servicios Web". Los servicios Web proporcionan estándares para interoperar entre diferentes aplicaciones de software, funcionando en diferentes plataformas.
- **Mayo 2003:** el W3C adopta de manera definitiva y oficial su Política de Patentes libre de derechos de autor. Ésta gestiona las patentes en el proceso de producción de estándares Web, promoviendo siempre el desarrollo de estándares abiertos.

Estos son solo algunos de los principales hitos conseguidos por el W3C. Si el lector desea obtener más información del resto de hitos así como del W3C, se le remite a su página Web. [W3C, 2008]. Añadir pagina a bibliografia

Desde que fundó hasta el día de hoy, Tim Berners-Lee ocupa el puesto de director del consorcio.

1.6. LOS LENGUAJES XUL Y JAVASCRIPT

A continuación se comentan brevemente los dos lenguajes utilizados para realizar este proyecto.

1.6.1. XUL

XUL (acrónimo de XML-based User-interface Language), es un lenguaje basado en XML utilizado para describir y crear interfaces de usuario, para el navegador Firefox. Se creó para que las interfaces fueran portables, por lo que permite desarrollar aplicaciones multi-plataforma sofisticadas o complejas sin necesidad de herramientas especiales.
Obviamente, al ser un lenguaje basado en XML contiene todas las características disponibles del mismo y otras propias que aportan ventajas a los desarrolladores.

El uso del lenguaje XUL para implementar interfaces para el navegador Firefox, aporta muchas ventajas y facilidades a los desarrolladores, ya que, como ya he indicado anteriormente, XUL brinda portabilidad.

La mayoría de las aplicaciones se deben desarrollar utilizando las características propias de una plataforma específica, haciendo que el desarrollo de software multi-plataforma sea costoso y consuma tiempo. Es por esto que en el pasado se han desarrollado soluciones multi-plataforma que brinden dicha portabilidad (como por ejemplo Java), y que este aspecto sea su característica más fuerte. Con XUL, una interfaz puede ser implementada y modificada fácil y rápidamente.
XUL puede ser utilizado en lugar de HTML, cuando se requiera desarrollar una interfaz de usuario portable y compleja. A diferencia de HTML, XUL provee un gran conjunto herramientas para crear menús, paneles, barras de herramientas, entre otras. Gracias a esto, no será necesario utilizar un lenguaje de programación propietario o incluir un gran código JavaScript para manejar el comportamiento de la interfaz de usuario. [Mozilla Developer Center]

1.6.2. JAVASCRIPT

JavaScript, en adelante JS, es un lenguaje de programación interpretado, es decir, un lenguaje que no necesita ser compilado para poder ser ejecutado.
El lenguaje JS fue inventado y desarrollado por la empresa NetScape Communications, apareciendo la primero versión de este lenguaje en el navegador NetScape Navigator 2.0.
JS tiene una característica especial que lo diferencia de los demás lenguajes interpretados y que lo hace especialmente idóneo para trabajar con la Web, y es que es interpretado (ejecutado) por los navegadores, con lo que, se consigue enviar documentos a través de la Web que lleven incorporados códigos fuentes de programas escritos con este lenguaje, convirtiéndose en documentos dinámicos.
La ventaja de escribir programas en JS es que se ejecutan en el navegador del cliente Web, sin necesidad de que intervenga el servidor, simplificándose a una

única transacción la carga de la página Web que contiene tanto el formulario como el programa en JS que proporciona los resultados.
Las dos principales características de JS son, por un lado es un lenguaje basado en objetos, es decir, se basa en la programación orientada a objetos pero con menos restricciones. Por otro lado es un lenguaje orientado a eventos, con funciones que responden a movimientos del ratón, pulsación de teclas, apertura o cerrado de ventanas, son algunos de los eventos que controla JS.

Es importante resaltar que existen dos tipos de JS, por un lado está el JavaScript que se ejecuta en el cliente Web, este es el JS propiamente dicho, aunque su nombre técnico es Navigator JavaScript. Por otro lado existe un JavaScript que se ejecuta en el servidor y se conoce con el nombre técnico de LiveWire JavaScript. Para este proyecto se ha utilizado el primero de los dos.
Todos los navegadores modernos interpretan el código JavaScript integrado dentro de páginas Web. Para que el navegador pueda interactuar con una página Web es necesario que el programa implementado en JavaScrit siga una implementación del DOM [Punto siguiente].
Para que todos todas las páginas Web pudieran ser interpretadas por los navegadores, independientemente de que navegador se tratase, los autores de JavaScript lo propusieron como estándar a la ECMA, el cual lo aceptó. Poco después también se convirtió en un estándar ISO.

Sin embrago, Microsoft ha desarrollado su propia versión de JavaScript, conocida como JScript, es muy similar a la versión del estándar pero con pequeñas diferencias en el modelo de objetos del navegador que hacen que ambas versiones sean frecuentemente incompatibles.

Sin embargo, JavaScript no permite un control sobre los recursos del ordenador, cada programa en JavaScript tiene acceso únicamente al documento HTML en el que está incluido, y si acaso, a la ventanas en las que se ejecuta el navegador dentro del cual se esta ejecutando el programa JS. Hay cosas que no se pueden hacer con JS, especialmente las relacionadas con el acceso a ficheros.

Diferencias entre JS y Java

Aunque tengan casi el mismo nombre se tratan de dos lenguajes de programación completamente diferentes. A continuación se detallan algunas de las diferencias más importantes entre ambos lenguajes:

- Java es un lenguaje de propósito general, es decir, no está especialmente orientado a la Web, puede utilizarse para programar cualquier tipo de aplicación aunque no sea para la Web.
  JS es un lenguaje específico para la Web, es decir, solamente se puede utilizar para programar aplicaciones que interactúen con la Web.
- JS es un lenguaje de programación interpretado y ejecutado por el cliente y Java es un lenguaje compilado por el servidor y que puede ser ejecutado en el cliente.
- JS esta basado en objetos, utilizando instancias de los mismos y no existiendo el concepto de clase ni todas las propiedades de la misma (herencia, encapsulación, etc.). Java es un lenguaje orientado a objetos.
- El código del programa en JS va inscrito en la página HTML en la que se ejecuta.
- JS es un lenguaje débilmente tipificado, es decir, no es necesario declarar el tipo de los datos antes de usarlos, mientras que Java es un lenguaje fuertemente tipificado.

Para este proyecto se ha usado la versión 1.8 de JS, que es la adecuada para la versión 3.0 del navegador, ya que esta versión aporta nuevas funciones que ayudan a explotar toda la potencia del navegador.

1.7. DOM

El Modelo de Objetos del Documento (Document Object Model ,DOM) es una interfaz de programación de aplicaciones (API) para documentos HTML y XML. Dicho modelo define la estructura lógica de los documentos y el modo en que se accede y manipula un documento. [DOM, 2008]
El DOM guarda una gran similitud con la estructura del documento al que esta modelizando. Veámoslo con el siguiente ejemplo; considérese la siguiente tabla tomada de un documento HTML:

```
<TABLA>
<CUERPO>
<TR>
<TD>Shady Grove</TD>
<TD>Aeolian</TD>
</TR>
<TR>
<TD>¿Cómo estás?</TD>
<TD>María</TD>
</TR>
</CUERPO>
</TABLA>
```

Figura 19. Tabla de un documento HTML

El DOM la representa del siguiente modo:

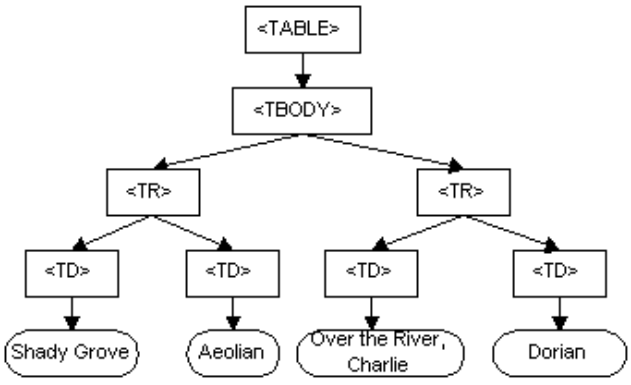


Figura 20. Árbol que representa la tabla de la figura anterior

Como podemos observar en la representación gráfica, los documentos en el DOM tienen una estructura lógica parecida a un árbol, pero también pueden ser bosques, ya que, puede contener más de un árbol. Sin embargo, el DOM no especifica que los documentos deban ser representados como un árbol, ni tampoco especifica cómo deben implementarse las relaciones entre objetos, en definitiva, el DOM es un modelo lógico que puede implementarse de cualquier manera que sea conveniente.

Una propiedad importante de los modelos de estructura del DOM es su isomorfismo estructural, es decir, si dos implementaciones cualesquiera del DOM se usan para

crear una representación del mismo documento, ambas crearán el mismo modelo de estructura, con exactamente los mismos objetos y relaciones.

Como hemos podido apreciar hasta el momento, el DOM es un "modelo de objetos" en el sentido tradicional del diseño orientado a objetos, ya que, los documentos se modelan usando objetos, y el modelo comprende no solamente la estructura de un documento, sino también el comportamiento de un documento y de los objetos de los cuales se compone. Resumiendo, los nodos del diagrama anterior no representan una estructura de datos, sino que representan objetos, los cuales pueden tener funciones e identidad.

Como modelo de objetos, lo que sí identifica el DOM es lo siguiente:

- Las interfaces y objetos usados para representar y manipular un documento.
- La semántica de estas interfaces y objetos, incluyendo comportamiento y atributos.
- Las relaciones y colaboraciones entre estas interfaces y objetos.

Actualmente, la jerarquía del DOM para la representación de documentos es la que se muestra en la siguiente figura:

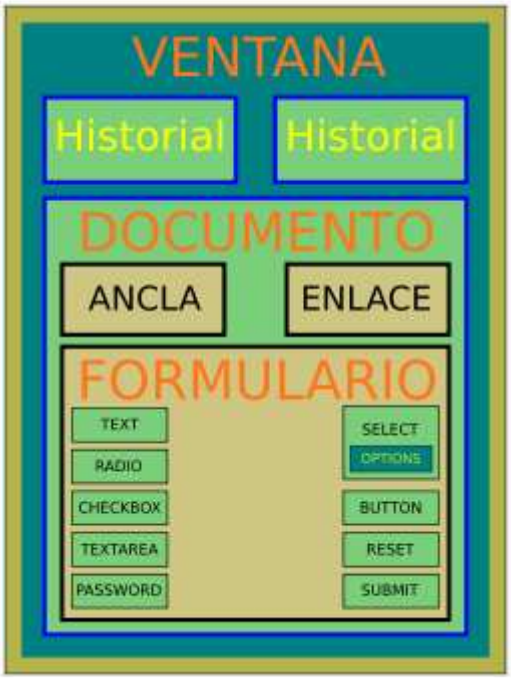


Figura 21. Jerarquía del DOM para la representación de documentos

1.8. CSS

Las hojas de estilo en cascada (Cascading Style Sheets ,CSS) son un mecanismo simple creado para controlar el aspecto o presentación de los documentos electrónicos escritos en HTML o XML (y por extensión en XHTML) por pantalla.
CSS es la mejor forma de separar la definición del contenido de la definición de su presentación y su uso es necesario para crear páginas Web complejas. La separación de ambas definiciones obliga a crear documentos HTML y XML bien definidos y con significado. Como la gran mayoría de estándares de Internet, es el consorcio W3C el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los navegadores.

El separar ambas definiciones, la del contenido de la de la presentación, mejora la accesibilidad del documento, reduce la complejidad del mantenimiento y permite visualizar el mismo documento en diferentes dispositivos.

Una vez definido y creado el contenido de una página Web, se pasa a dar formato a dicho contenido, el color, el tamaño la fuente, la ubicación de los diferentes elementos en la página, etc. Para dar formato a esa definición de aspecto se utiliza el lenguaje CSS.

CSS proporciona tres caminos diferentes para aplicar las reglas de estilo a una página Web:

Una hoja de estilo externa: es una hoja de estilo que está almacenada en un archivo diferente al archivo donde se almacena el código XUL de la extensión. Esta es la manera más potente de programar porque separa completamente las reglas de formateo para la extensión del código fuente. En el proyecto se ha seguido esta pauta para definir el aspecto de la aplicación.

Una hoja de estilo interna: es una hoja de estilo que está incrustada dentro de un documento HTML. De esta manera se consigue separar la información del estilo del código HTML, ya que, se ubica dentro del elemento <HEAD>.

Un estilo en línea: es un método para insertar el estilo de la página directamente dentro de una etiqueta HTML.

Una vez explicado que son las CSS y para que se utilizan, vamos a ver cuando y porqué se crean las CSS, así como quién las creó.

Los motivos que propiciaron al organismo W3C a proponer un estándar para la definición de estilos fueron, por un lado, se detectó que el lenguaje de etiquetas SGML no permitía aplicar consistentemente diferentes estilos a documentos electrónicos, y por otro lado, la guerra de los navegadores, ya que, ante la falta de dicho estándar los documentos no se visualizaban igual en un navegador y en otro.
Nueve fueron las propuestas presentadas y las dos ganadoras fueron la CHSS (Cascading HTML Style Sheets) y la SSP (Stream-based Style Sheet Proposal). Cada una de las propuestas ganadoras fue desarrollada por equipos diferentes, pero a mediados de los noventa (hacia 1994) ambos equipos de desarrollo se unieron creando un nuevo lenguaje de estilos que unía lo mejor de ambas propuestas, ese lenguaje fue bautizado con el nombre de CSS. A finales de 1996 el W3C publica la primera recomendación conocida como 'CSS Nivel 1'.
Debido a la aceptación que tuvo esta publicación, el W3C decide crear un grupo de trabajo que únicamente se dedicará al CSS, hasta ese momento, el grupo de trabajo dedicado al CSS formaba parte del grupo de trabajo encargado de desarrollar HTML. Como resultado del trabajo realizado por el grupo de trabajo CSS, en Mayo de 1998 se hace publica la segunda recomendación, conocida como 'CSS Nivel 2'.
Desde 1998 y hasta hoy está en desarrollo la tercera recomendación, conocida por sus borradores publicados, como 'CSS Nivel 3'. [Eguíluz].

2. ESPECIFICACIÓN DE REQUISITOS

A continuación se va a realizar la especificación de requisitos según el estándar IEEE 830.

Los principales objetivos de una especificación de requisitos son:

- Ayudar a los usuarios a describir claramente lo que se desean obtener mediante un producto software determinado.
- Ayudar a los desarrolladores a entender qué quiere exactamente el usuario.
- Servir de base para desarrollos de estándares de ERS particulares para cada organización, ya que, cada uno puede desarrollar sus propios estándares para definir sus necesidades.

En el presente apartado se va seguir el esquema para la Especificación de Requisitos Software (ERS) definido en el IEEE 830-1998. A continuación se presenta una figura del esquema, para que el lector vea los puntos que se van a tratar en este apartado y la estructura de los mismos.

1 Introducción
1.1 Propósito
1.2 Ámbito del sistema
1.3 Definiciones, acrónimos y abreviaturas
1.4 Referencias
1.5 Visión general del documento
2 Descripción general
2.1 Perspectiva del producto
2.2 Funciones del producto
2.3 Características de los usuarios
2.4 Restricciones
2.5 Suposiciones y dependencias
2.6 Requisitos futuros
3 Requisitos específicos
3.1 Interfaces externas
3.2 Funciones
3.3 Requisitos de rendimiento
3.4 Restricciones de diseño
3.5 Atributos del sistema
3.6 Otros requisitos
4 Apéndices
5 Índice

2.1. INTRODUCCIÓN

En los siguientes puntos se va a realizar la especificación de requisitos de un producto software que formará parte del navegador Firefox como una extensión. Por tanto, los usuarios de la misma serán todos aquéllos que trabajen con dicho navegador y decidan descargarse este producto.

2.1.1. PROPÓSITO

El propósito de esta ERS es detallar todos los requisitos que debe satisfacer este proyecto para que sea aceptado por la comunidad Mozilla. En dicha ERS se recogen todos los requisitos que cumplía la versión 1.0 de la extensión y los nuevos requisitos que debe satisfacer.

El objetivo es aportar una visión precisa y clara de la funcionalidad de la extensión y va dirigida a cualquier usuario que utilice el navegador Firefox.

2.1.2. ÁMBITO DE LA EXTENSIÓN FIREFOX

La extensión recibió como nombre 'Web Filtering Toolbar' en la versión 1.0 y para este proyecto se ha decido mantener el nombre. Ésta consiste en una barra de búsqueda rápida para el navegador Firefox. Esta compuesta por un cuadro de texto y un panel de botones.
La funcionalidad que realizaba en la versión 1.0 era buscar la información que el usuario necesitaba dentro de una página Web, con distintos niveles de visualización y con la capacidad de mantener o reconfigurar la estructura de la página. La nueva funcionalidad consiste en resaltar todas las coincidencias de la palabra buscada dentro de la página, respetando el filtrado aplicado anteriormente si fuera el caso, filtrar páginas implementadas con marcos y resaltarlas del mismo modo que las páginas normales, resaltar una a una todas las coincidencias de la palabra dentro de la página y ocultar o mostrar la barra cada vez que el usuario quiera.

2.1.3. DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS

En este punto se va a detallar todos los términos y abreviaturas utilizadas en el desarrollo de la especificación de requisitos.

ERS: Especificación de Requisitos Software. Se puede definir como el proceso de estudio de las necesidades de los usuarios para llegar a una definición de los requisitos del sistema, hardware o software, así como el proceso de estudio y refinamiento de dichos requisitos.

Requisito: se define como una condición o capacidad que necesita el usuario para resolver un problema o conseguir un objetivo determinado. Esta definición extiende y se aplica a las condiciones que debe cumplir o poseer un sistema para satisfacer un contrato, una norma o una especificación.

IEEE: Instituto de Ingenieros Eléctricos y Electrónicos. Es una asociación técnico-profesional mundial dedicada a la estandarización. Su trabajo es promover la creatividad, el desarrollo y la integración, compartir y aplicar los avances en las tecnologías de la información, electrónica y ciencias en general para el beneficio de los profesionales.

PC: Personal Computer, en castellano Ordenador Personal.

Add-on: extensión en castellano, es una aplicación que depende de un sistema mayor y le añade una funcionalidad extra que no posee. Esta funcionalidad debe ser útil para el usuario del sistema mayor.

2.1.4. REFERENCIAS

Para la elaboración de este documento se ha seguido el estándar IEEE 830-1998, especificado en el siguiente documento:
[IEEE 1998] IEEE Recommended practice for software requeriments specification.

2.1.5. VISIÓN GENERAL DEL PRODUCTO

En los puntos siguientes se va a explicar con mas detalle toda la funcionalidad de la extensión, tanto la que ya tenía como la nueva, los requisitos funcionales, los requisitos específicos, así como todas las restricciones de los lenguajes de programación utilizados.

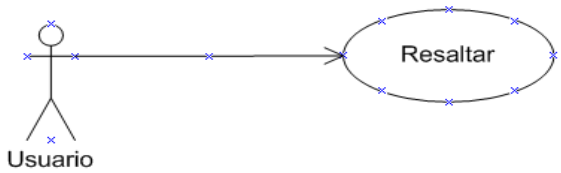
2.2. DESCRIPCIÓN GENERAL

2.2.1. PERSPECTIVA DEL PRODUCTO

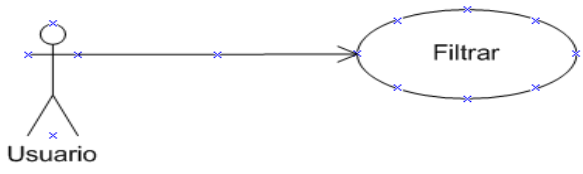
Este producto forma parte del navegador Firefox como un add-on, extensión en castellano, por lo tanto, está enteramente relacionado con él mismo, de hecho, únicamente funciona si el usuario tiene dicho navegador instalado.
La versión anterior de este producto (Web Filtering Toolbar 1.0) funcionaba con el navegador Firefox en su segunda versión, por lo tanto, el primer objetivo de este proyecto fue hacerlo compatible con la versión 3 de dicho navegador.
Este producto conserva la funcionalidad básica de su primera versión, y presenta funcionalidad nueva en su nueva versión. A grandes rasgos, este producto sirve para facilitar la búsqueda de información dentro de una página web, mostrando o resaltando solamente aquello que interesa al usuario.

2.2.2. FUNCIONES DEL PRODUCTO

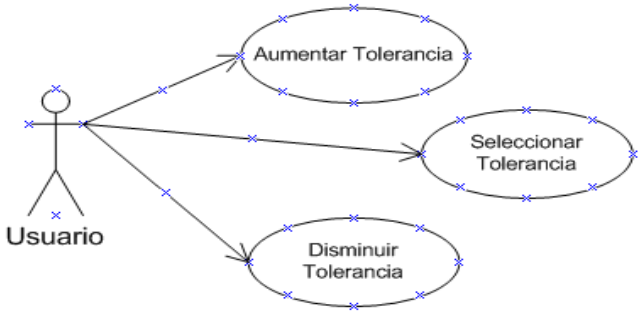
Resaltar: dada la palabra que el usuario desea buscar dentro de una página Web determinada, esta función se encarga de resaltar en rojo todas las coincidencias de la palabra dentro de la página Web, consiguiendo que el usuario localice la información que desea de manera rápida y visual.



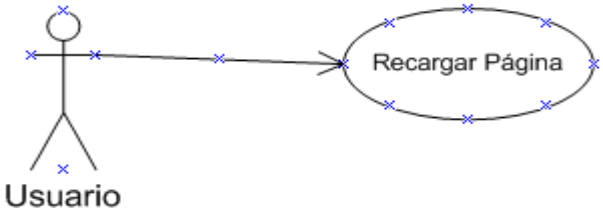
Filtrado: dada la palabra que el usuario desea buscar dentro de una página Web determinada, esta función se encarga de dejar visible únicamente aquella información que contiene la palabra buscada, ocultando el resto de la información de la página. Con esta función, se consigue que el usuario vea solo aquello que quiere ver.



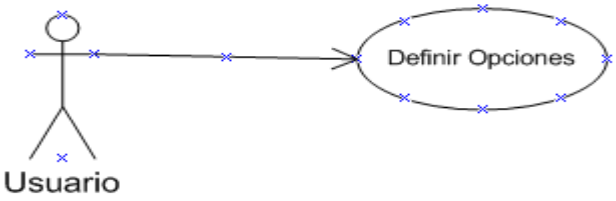
Tolerancia: con esta función el usuario puede decidir aumentar o disminuir la cantidad de información que desea ver, una vez aplicado el filtrado, es decir, si solamente desea ver la frase que contiene la palabra, el párrafo, etc. También se puede definir el nivel de tolerancia antes de filtrar, de este modo ya se consigue ver la información de la página con el nivel deseado.



Recargar página: una vez aplicada la función resaltar o la función filtrar, la visualización de la página Web queda modificada por la acción de dichas funciones. Con esta función, el usuario siempre puede volver a recargar la página web original, es decir, con la visualización y la estructura que tiene normalmente.



Opciones: esta función se encarga de abrir un menú con las diferentes preferencias que puede escoger aplicar el usuario sobre la extensión. Dichas preferencias y para qué sirven se explican una a una a continuación:



- **Mantener la estructura (*Keep Structure*):** si el usuario escoge mantener la estructura de la página Web, cuando éste filtre la página, la información se mostrará en la posición que ocupa dentro de la página Web original. Si por el contrario, el usuario decide no mantener la estructura, cuando aplique un filtrado, se mostrará toda la información agrupada al inicio de la página, quedando una página más corta, y en muchos casos desapareciendo la barra de desplazamiento vertical.

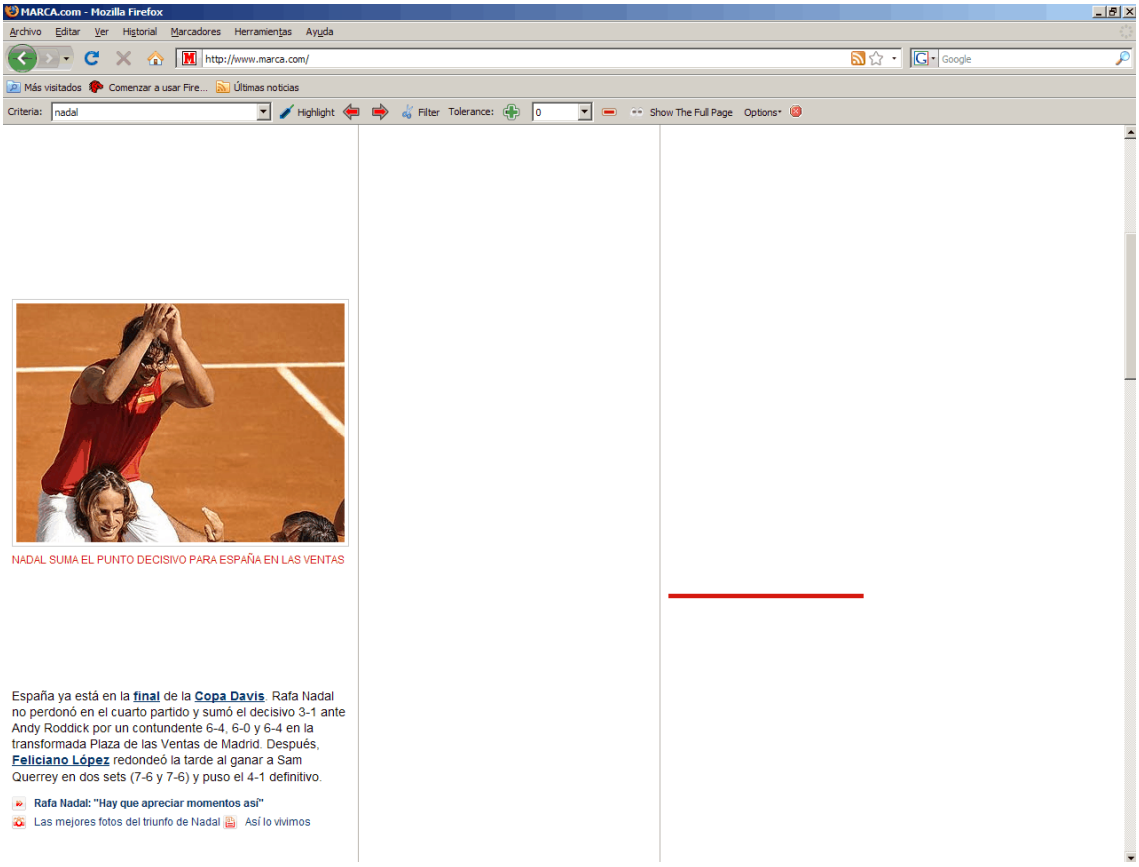


Figura 22. Filtrado con la opción 'Keep Structure' seleccionada.

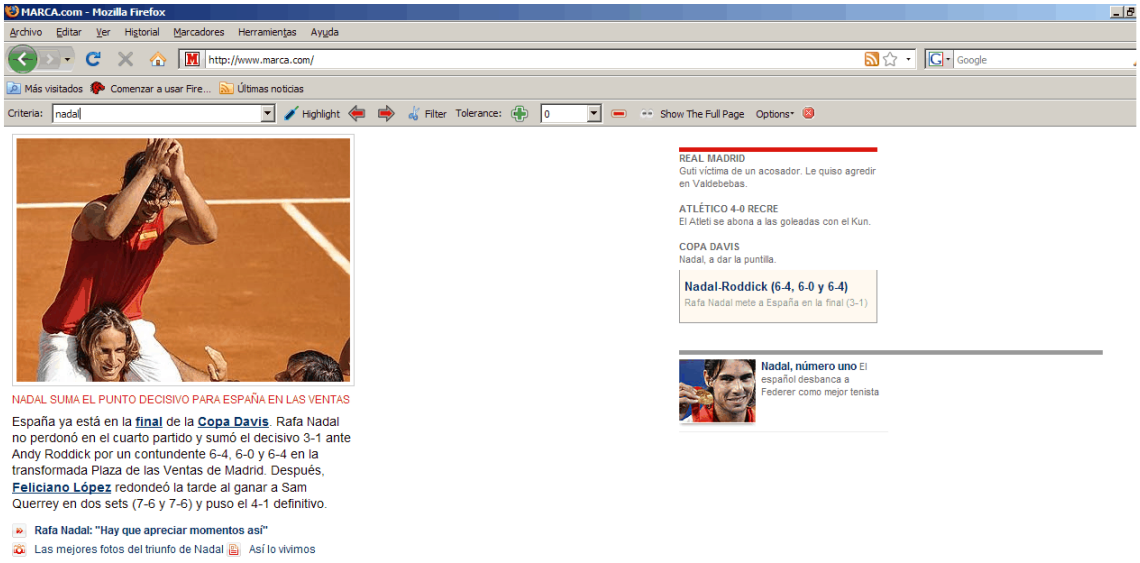


Figura 23. Filtrado sin la opción 'Keep Structure'.

- **Formato de los iframes (*Format Size Iframes*):** si el usuario decide mantener el formato de los iframes que aparecen en la página Web, cuando éste aplique un filtro, si la palabra que busca se encuentra dentro de un iframe, el tamaño del mismo se mantendrá tal cual aparece en la página original, manteniendo, si el iframe la posee y si es necesario, la barra de desplazamiento vertical. Si por el contrario el usuario decide no mantener el tamaño de los iframes, cuando la palabra que busca se encuentre dentro del mismo, desaparecerá la barra de desplazamiento vertical, visualizando toda la información de un solo vistazo.

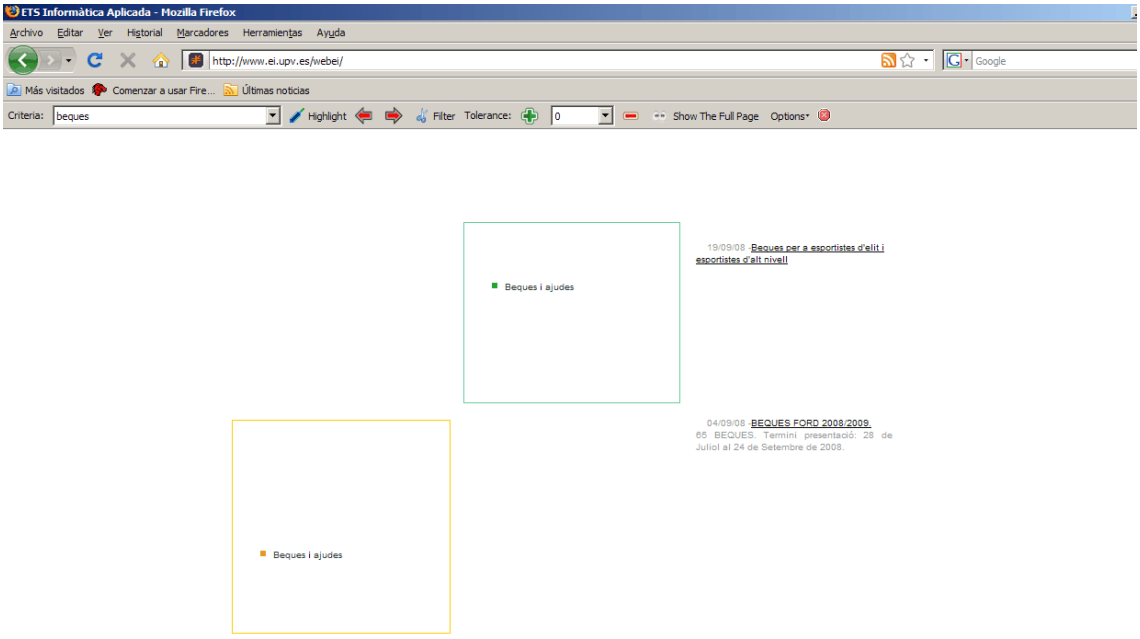


Figura 24. Filtrado sin mantener el formato de los iframes. Criterio de búsqueda utilizado 'beques'

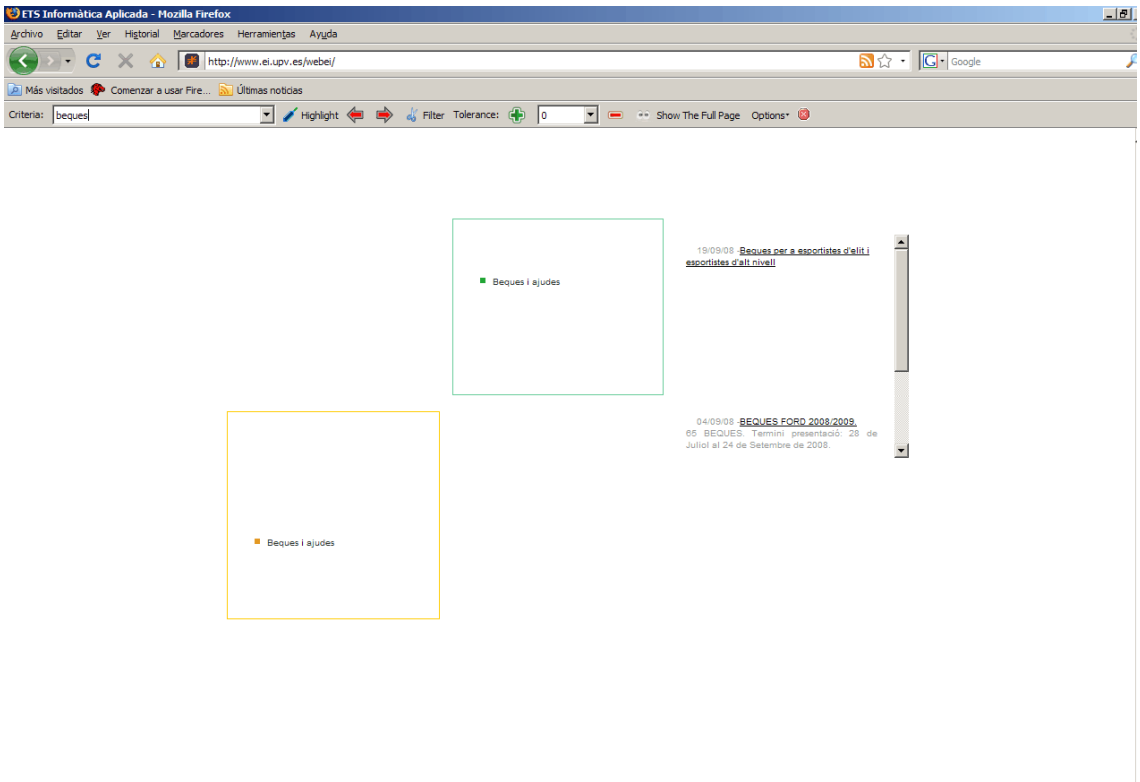


Figura 25. Filtrado manteniendo el formato de los iframes. Criterio de búsqueda utilizado 'beques'

- **Mantener el árbol *(Keep Tree):*** cuando un nodo en el árbol representa un documento que debe incluirse en el resultado final del filtrado, es posible incluir también todos los nodos del camino desde ese nodo a la raíz, es decir, los antecesores del nodo en cuestión. Para conseguir esto, la opción keep tree debe estar seleccionada, por el contrario, si esta opción no está activada, únicamente se mostrará el nodo que forme parte del filtrado.

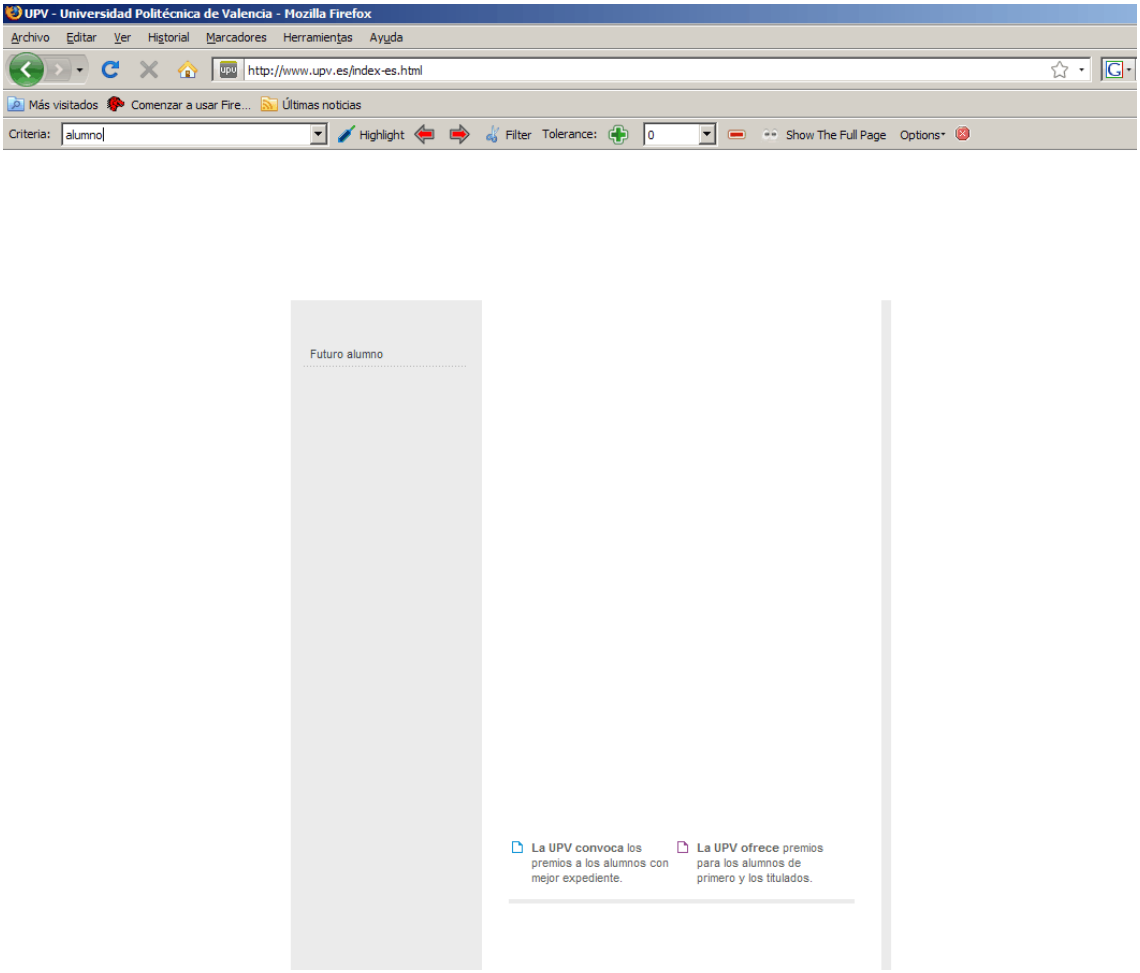


Figura 26. Filtrado con la opción 'Keep Tree' seleccionada.

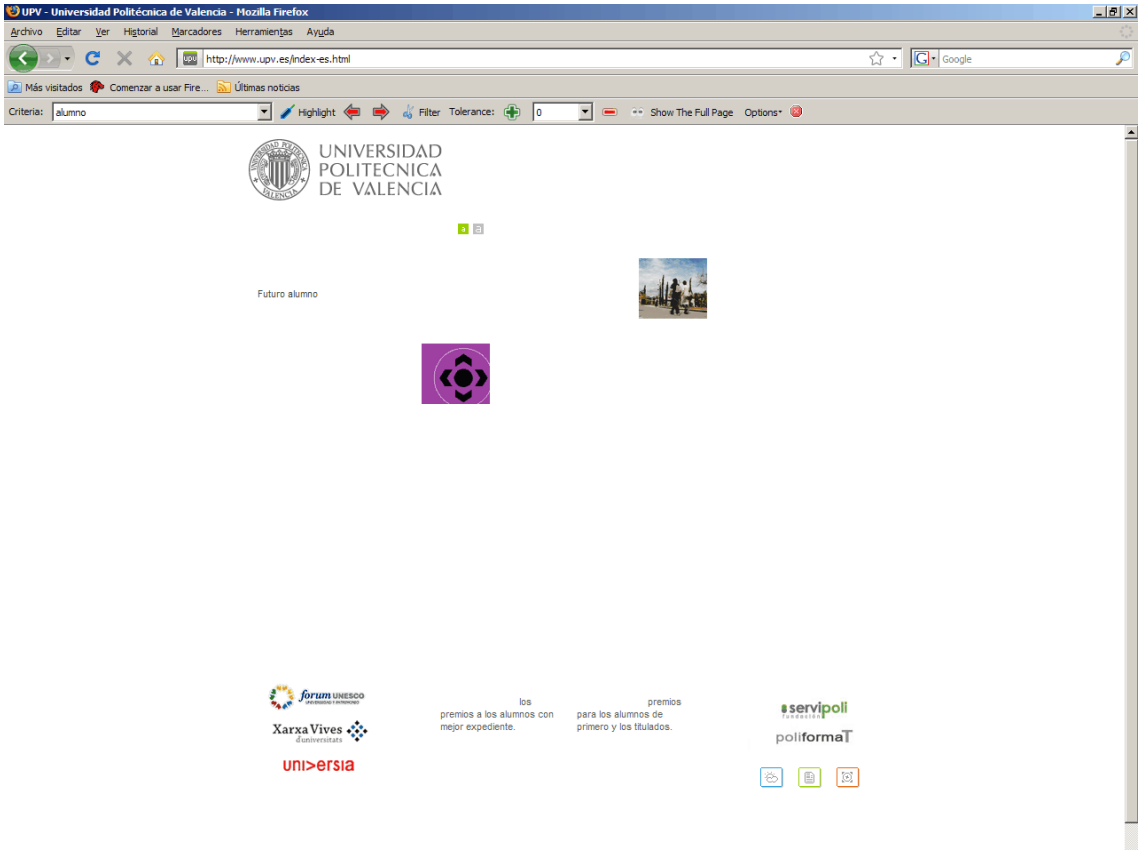


Figura 27. Filtrado sin la opción 'Keep Tree' y manteniendo la estructura de la página

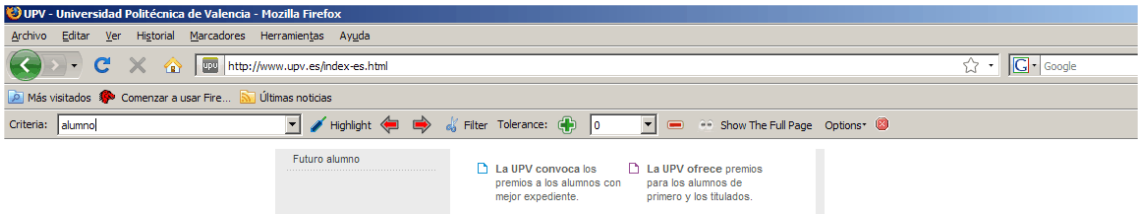


Figura 28. Filtrado con la opción 'Keep Tree' seleccionada y sin mantener la estructura de la página

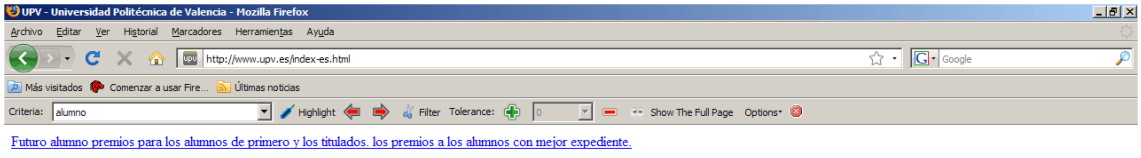


Figura 29. Filtrado sin la opción 'Keep Tree' y sin mantener la estructura de la página

2.2.3. CARACTERÍSTICAS DE LOS USUARIOS

Esta aplicación va dirigida a cualquier tipo de usuario que utilice el navegador Firefox. Debe tener un conocimiento básico de que es una extensión, para qué sirven, cómo se consiguen y cómo se instalan.
El uso de la aplicación es sencillo e intuitivo, con lo que, un usuario con conocimientos básicos de Internet puede utilizarla sin ningún tipo de problema.

2.2.4. RESTRICCIONES

El producto debe cumplir una serie de restricciones, las cuales se detallan a continuación.

- Es imprescindible que cumpla todos los requerimientos especificados por los desarrolladores de Firefox para la implementación de extensiones. Entre ellas destaca que el nombre de toda variable global y toda función debe ir precedido con el nombre de la extensión, para evitar posibles confusiones con otras extensiones (que el usuario pueda tener instaladas) con los mismos nombres de variables. En este caso los nombres van precedidos de la cadena 'webfiltering_'.
- El usuario debe tener instalado la versión 3 del navegador Firefox.
- El usuario debe tener acceso a Internet.

2.2.5. SUPOSICIONES Y DEPENDENCIAS

La extensión propuesta depende enteramente del navegador Firefox 3. Las futuras versiones de dicho navegador sólo podrán utilizar la extensión que aquí se describe si son compatibles con el modelo DOM utilizado en la versión 3 de Firefox.

2.2.6. REQUISITOS FUTUROS

Como funcionalidades futuras se proponen las recogidas a continuación:

- Se propone como mejora futura, el poder filtrar la estructura de documentos escritos en XML.
- Añadir agentes de búsqueda.
- Permitir definir perfiles de usuario.
- Integrar un lexicón al producto.
- Que se permita resaltar las palabras una a una, con los botones anterior y siguiente, en páginas implementadas con marcos.

2.3. REQUISITOS ESPECIFICOS

Esta sección contiene todos los requerimientos hasta un nivel de detalle suficiente para permitir a los diseñadores diseñar un sistema que satisfaga los requisitos que se van a especificar, y que permita diseñar las pruebas que ratifiquen que la extensión cumple con las necesidades requeridas.

2.3.1. INTERFACES EXTERNAS

No hay requisitos que afecten a la interfaz de usuario, puesto que no ha habido una captura de requisitos convencional, la interfaz fue diseñada para la versión 1.0 en el año 2007 y ésta ha sido mantenida. Las únicas restricciones que se encuentran,

son las propias del lenguaje XUL. Dichas restricciones se comentaran en el apéndice.
La interacción con el usuario se lleva a cabo a través del ratón y del teclado del ordenador.

2.3.2. REQUISITOS FUNCIONALES

2.3.2.1. Funcionalidad '*Filter*'

2.3.2.1.1. Descripción

Está función se encarga de ocultar o borrar los componentes de la página Web que no son relevantes, por no coincidir con el criterio de búsqueda introducido en el cuadro de texto etiquetado como '*Criteria*'.
Una página Web se puede ver como un árbol de nodos de etiquetados. Este árbol se representa internamente con el estándar DOM [DOM].

2.3.2.1.2. Requisitos funcionales

Precondiciones necesarias:

- Debe de introducirse un criterio de filtrado en el cuadro de texto.
  - Posibles errores:
    1. No se ha introducido una palabra en el cuadro de texto.
       En este caso, la aplicación muestra un mensaje por pantalla avisando al usuario que el criterio de búsqueda es nulo.
    2. La palabra introducida como criterio de búsqueda no se encuentra en la página.
       Este caso puede darse por dos motivos. Uno de ellos es que la palabra se haya escrito erróneamente en el cuadro de texto, y el otro es que la palabra esté escrita correctamente pero no se encuentre en la página. En ambos casos, la aplicación mostrará el mismo mensaje de error por pantalla, avisando al usuario de ello.

- Definir unas preferencias de cómo desea el usuario que se muestre la página resultante.
  - En el caso que el usuario no las defina, la aplicación debe utilizar unos valores por defecto para cada una de dichas preferencias.
  - Dichas preferencias pueden cambiarse siempre que el usuario lo desee. Una vez cambiadas, la siguiente vez que el usuario pulse el botón '*Filter*', la página resultante se filtrará utilizando las nuevas preferencias.

Preferencias para el usuario

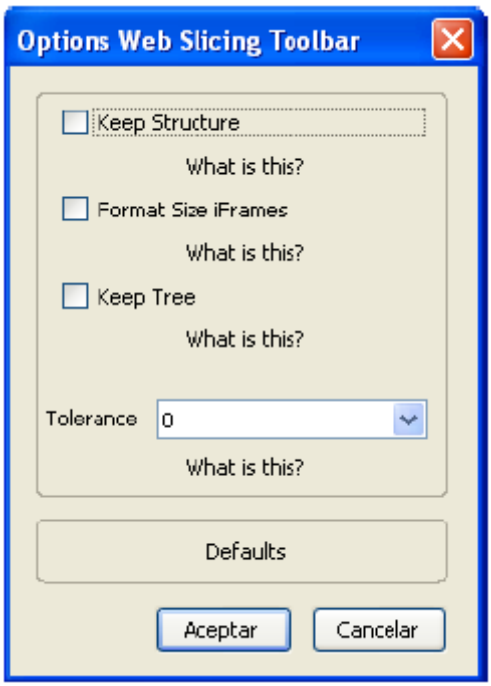


Figura 30. Menú de opciones

- *Keep Structure:* si el usuario selecciona esta opción en el menú de opciones, cuando pulse el botón 'Filter', los componentes se mostrarán en la misma posición en la que aparecían en la página original. La página resultante de aplicar esta función con está opción mostrará huecos en blanco ocultando aquella información que no sea relevante (ver figura 22). Si está opción no está seleccionada, la información en la página resultante se reorganizará de modo que no existan espacios en blanco (ver figura 23).

- *Format Size iFrames:* los iFrames permiten incrustar una página Web dentro de otra página Web. Si está activa la opción 'Format Size iFrames', el tamaño original de los iFrames se adaptan al tamaño del fragmento de página resultante de realizar 'filter'. Por el contrario, si no está activada, se mantendrá el tamaño original de los iFrames. Normalmente la página Web de un iFrame es más grande que el área reservada para el iFrame, dentro de la página Web principal, de ahí, que los iFrames usen normalmente barras de desplazamiento, tanto verticales como horizontales. A veces, el fragmento extraído del iFrames es pequeño, con lo que, ajustando el tamaño del iFrame, es innecesario tener áreas vacías producidas por el desplazador. (Veánse las figuras 24 y 25).

- *Keep Tree:* cuando un nodo del árbol representa un documento que se encuentra dentro del fragmento de la página, obtenido después de aplicar el filtrado, también es posible incluir todos los nodos antecesores, que son aquéllos que van desde el nodo raíz o root hasta el nodo en cuestión. Para poder obtener los antecesores del nodo, debe estar activada está opción. (Veánse las figuras de la 26 a la 29).

- *Tolerance:* el valor por defecto de está opción es cero. Con tolerancia cero, sólo se mostrarán en la página resultante tras realizar un filtrado los nodos relevantes y sus descendientes. Si el usuario

aumenta el valor de la tolerancia, los nodos que estén directamente relacionados con los nodos relevantes, pasan a ser relevantes también. Por ejemplo, si la tolerancia aumenta en uno, los padres (y sus descendientes) de los nodos relevantes pasan a ser relevantes también.

Postcondiciones:

Tras realizar el filtrado, la página original, que se recibe como entrada, es transformada. Como resultado final, se obtiene una nueva página (copia de la original) pero con la información o nodos que no son relevantes ocultos mediante espacios en blanco (véase figuras 1 y 2).

2.3.2.2. Funcionalidad de resaltado de coincidencias

2.3.2.2.1. Descripción

Está función se encarga de resaltar todas las coincidencias que aparezcan en la página que cumplan el criterio de búsqueda.

2.3.2.2.2. Requisitos funcionales

Precondiciones necesarias:

- Debe de introducirse un criterio de búsqueda en el cuadro de texto.
    - Posibles errores:
        1. No se ha introducido una palabra en el cuadro de texto.
           En este caso, la aplicación muestra un mensaje por pantalla avisando al usuario que el criterio de búsqueda es nulo.
        2. La palabra introducida como criterio de búsqueda no se encuentra en la página.
           Este caso puede darse por dos motivos. Uno de ellos es que la palabra se haya escrito erróneamente en el cuadro de texto, y el otro es que la palabra esté escrita correctamente pero no se encuentre en la página. En ambos casos, la aplicación mostrará el mismo mensaje de error por pantalla, avisando al usuario de ello.

Esta función mantiene los filtrados anteriores; es decir, si primero se ha aplicado un filtrado sobre la página original, esta función, solamente resaltará las coincidencias que haya dentro del/los fragmento/s obtenido/s.

Poscondiciones:

Como salida se obtiene la página Web original pero con todas las coincidencias resaltadas. (Veáse figura 8)

2.3.2.3. Funcionalidad resaltar siguiente

2.3.2.3.1. Descripción

Esta función se encarga de ir resaltando una a una, comenzando por el principio de la página Web y hasta el final de la misma, todas las coincidencias de la palabra buscada dentro de la página.

2.3.2.3.2. Requisitos funcionales

Precondiciones necesarias:

- Debe de introducirse un criterio de búsqueda en el cuadro de texto.
    - Posibles errores:
        1. No se ha introducido una palabra en el cuadro de texto.
           En este caso la aplicación muestra un mensaje por pantalla avisando al usuario que el criterio de búsqueda es nulo.
        2. La palabra introducida como criterio de búsqueda no se encuentra en la página.
           Este caso puede darse por dos motivos. Uno de ellos es que la palabra se haya escrito erróneamente en el cuadro de texto, y el otro es que la palabra esté escrita correctamente pero no se encuentre en la página. En ambos casos, la aplicación mostrará el mismo mensaje de error por pantalla, avisando al usuario de ello.

Esta función mantiene los filtrados anteriores; es decir, si primero se ha aplicado un filtrado sobre la página original, esta función, solamente resaltará una a una todas las coincidencias que haya dentro del/los fragmento/s obtenido/s.

Poscondiciones:

Como resultado se obtiene la página Web original pero con la coincidencia correspondiente en número de la palabra resaltada. (Veánse figuras 11 y 12).

2.3.2.4. Funcionalidad resaltar anterior

2.3.2.4.1. Descripción

Esta función se encarga de ir resaltando una a una, comenzando por el final de la página Web y hasta el principio de la misma, todas las coincidencias de la palabra buscada dentro de la página.

2.3.2.4.2. Requisitos funcionales

Precondiciones necesarias:

- Debe de introducirse un criterio de búsqueda en el cuadro de texto.
    - Posibles errores:
        1. No se ha introducido una palabra en el cuadro de texto.
           En este caso la aplicación muestra un mensaje por pantalla avisando al usuario que el criterio de búsqueda es nulo.
        2. La palabra introducida como criterio de búsqueda no se encuentra en la página.
           Este caso puede darse por dos motivos. Uno de ellos es que la palabra se haya escrito erróneamente en el cuadro de texto, y el otro es que la palabra esté escrita correctamente pero no se encuentre en la página. En ambos casos, la aplicación mostrará el mismo mensaje de error por pantalla, avisando al usuario de ello.

Esta función mantiene los filtrados anteriores; es decir, si primero se ha aplicado un filtrado sobre la página original, esta función, solamente resaltará una a una todas las coincidencias que haya dentro del/los fragmento/s obtenido/s.

Poscondiciones:

Como resultado se obtiene la página Web original pero con la coincidencia correspondiente en número de la palabra resaltada. (Veánse figuras 9 y 10).

2.3.2.5. Funcionalidad mostrar la página entera

2.3.2.5.1. Descripción

Esta función se encarga de recargar la página Web original con la que ha estado trabajando el usuario, de manera que si se había aplicado un filtrado o un resaltado en la página original, estos se perderán, obteniendo la página Web tal cual era originalmente.

2.3.2.5.2. Requisitos funcionales

Precondiciones necesarias:

Esta función no necesita precondiciones para poder ejecutarse correctamente.

Poscondiciones:

Como resultado de esta función se obtiene la página Web original.

2.3.3. REQUISITOS DE RENDIMIENTO

Se especifica como requisito de rendimiento que ninguna funcionalidad del producto debe tardar más de 10 segundos en ejecutarse.

No se han definido requisitos de carga del sistema, ya que, la extensión estará instalada en el PC del usuario que haya decidido descargársela y por lo que no hay usuarios concurrentes, ni almacenamiento en bases de datos, puesto que, este proyecto no utiliza ninguna base de datos.

2.3.4. RESTRICCIONES DE DISEÑO

Dado que la barra de herramientas es una extensión para Firefox, la implementación de la misma está sujeta a las restricciones de Firefox para el desarrollo de extensiones.

Por esta razón, la interfaz gráfica deberá ser realizada utilizando el lenguaje XUL; y el comportamiento se especificará utilizando el lenguaje Javascript.

2.3.5. ATRIBUTOS DEL SISTEMA

Este proyecto no tiene ninguna política de acceso, ni ningún tipo de control de acceso, puesto que es una extensión de acceso libre, que se integra con el navegador Firefox.
Al tratarse de una versión final, este proyecto no necesita ningún tipo de mantenimiento, pero si se considera necesario se le puede añadir funcionalidad nueva.

Si se desea mantener en uso la barra de herramientas desarrollada en este proyecto, se requerirá hacer las modificaciones necesarias para hacerlo compatible con las futuras versiones del navegador Firefox.

2.4. APÉNDICES

Las restricciones impuestas por este proyecto, vienen determinadas por los diferentes lenguajes de programación utilizados para el desarrollo del mismo, y por los estándares seguidos. [DOM].
Actualmente el estándar DOM para la versión 3 del navegador Firefox no contempla funciones para el tratamiento de ficheros XML, por lo que no se puede manipular ningún fichero XML.

2.4.1. RESTRICCIONES DE LAS EXTENSIONES

En este punto se detallan todas las restricciones que deben cumplir las extensiones para que sean compatibles y funcionen correctamente con Firefox.

1. Las extensiones usan una estructura de directorio que contiene el chrome, los componentes y la interfaz.
2. La interfaz de usuario se debe implementar en XUL [XUL] y el comportamiento que tendrá la misma ante las acciones del usuario en JavaScript [JS].
3. Las extensiones deben distribuirse en paquetes comprimidos en formato zip o en paquetes con formato xpi.
4. Cada extensión debe tener un archivo install.rdf, el cuál contiene metadatos acerca de la extensión, así como un identificador único, la versión de la misma, el autor o autores e información de compatibilidad con las diferentes versiones de Firefox.
5. Toda extensión debe contener un fichero 'manifiesto' (chrome.manifest) donde se indica la ubicación de los paquetes que utilizará la extensión.
6. El nombre de todas las variables globales y el nombre de todas las funciones definidas en el archivo javascript deben de ir precedidas del nombre de la extensión. Esto debe ser así, para evitar confusión de nombres, en el caso de que dos o más extensiones utilicen nombres de variables o funciones iguales, de entre las distintas que pueda tener instaladas el usuario. En este proyecto, todos los nombres antes citados van precedidos por la cadena 'webfiltering_'.

2.4.2. RESTRICCIONES DEL LENGUAJE XUL

A continuación se detallan las restricciones que impone el lenguaje XUL para desarrollar las extensiones de Firefox.

1. Todos los elementos definidos y sus atributos deben escribirse en minúscula, puesto que XML (lenguaje del que se deriva XUL) hace distinción entre mayúsculas y minúsculas.
2. Todos los valores en XUL deben introducirse entre comillas, aunque estos sean números.
3. Toda aplicación escrita en este lenguaje se estructura en paquetes. La organización de los mismos es la siguiente:
    a. Contenido (content): esta carpeta almacena las declaraciones de las componentes y los elementos que formaran parte de la interfaz. Estas declaraciones se recogen en un archivo con extensión xul. Esta

carpeta puede contener muchos archivos xul, pero el que define la ventana o componente principal debe tener el mismo nombre que el paquete. En este proyecto, el paquete se llama slicetoolbar, por lo tanto, el archivo xul principal se llama slicetoolbar.xul.

b. Aspecto (skin): esta carpeta almacena las hojas de estilo (css) y las imágenes. Las hojas de estilo se almacenan separadamente de los archivos xul para facilitar la modificación de las primeras.
c. Configuración regional (locale): esta carpeta almacena los archivos de los diferentes idiomas en los que se podrá instalar la aplicación, además de otros archivos que recojan configuraciones locales que se defina el usuario.

3. ANÁLISIS Y DISEÑO

3.1. INTERFAZ DE USUARIO

Figura 31. Detalle de la barra completa

La aplicación se muestra en forma de barra de herramientas, que se sitúa justo debajo de la barra actual que tiene el navegador Firefox. Dicha barra de herramientas, se compone de una serie de botones que hacen más fácil, intuitivo y amigable el uso de la misma para el usuario.

La primera componente que aparece en la barra de herramientas es un cuadro de texto desplegable y editable, es decir, un cuadro de texto donde se escribe la palabra a buscar dentro de la página. También almacena todas las palabras buscadas con anterioridad, de manera que el usuario, pueda seleccionar alguna de la lista para volver a buscarla.

A continuación aparece el botón 'HighLight' que, junto con la componente anterior, resalta todas las coincidencias de la palabra introducida en la página.

A continuación aparecen dos botones, la flecha hacia la derecha (siguiente) y la flecha hacia la izquierda (anterior). Dichos botones se encargan de ir resaltando cada una de las coincidencias que tenga la palabra en la página, bien la siguiente o bien la anterior.

Al lado de este botón aparece el botón 'Filter', que es el encargado de ocultar aquella información que no es útil de la página.

A continuación aparecen los botones para aumentar, disminuir o escribir el nivel de tolerancia deseado.

Seguidamente aparecen dos botones adicionales, uno que sirve para mostrar la página original, sin ningún filtro ni ningún resaltado aplicado y el botón de opciones. Este último botón despliega un menú con tres opciones:

- Si el usuario selecciona la opción de menú 'Options', se abrirá una nueva ventana donde podrá seleccionar las preferencias para el filtrado.
- La segunda opción muestra la ayuda de cómo usar la aplicación.
- La tercera opción abre otra ventana con información de los autores de la aplicación y la versión de la misma.

Por último, aparece el botón de cerrar, el cuál se encarga de ocultar la barra. Para volver a mostrarla basta con pulsar la combinación de teclas 'Ctrl + Y'.

Los iconos que aparecen en la barra de herramientas han sido extraídos de la distribución Ubuntu del sistema operativo Linux de software libre y la página Web www.famfamfam.com.

3.2. ERRORES CORREGIDOS DE LA VERSIÓN ANTERIOR

En la versión anterior del producto se detectaron una serie de errores, los cuales han sido corregidos para esta versión.

Todos los errores detectados se detallan a continuación:

1. Solamente se filtraban imágenes comparando su atributo 'alt' con la palabra introducida.

   En esta versión se filtran las imágenes comparando los siguientes atributos con la palabra introducida por el usuario: name, alt, longDesc y src. De manera que si alguno de ellos contiene la palabra la imagen se mostrará.

2. La instrucción window.addEventListener(), no tenía escrito correctamente el nombre de la función inicializar.

   Lo único que le faltaba para que su nombre fuera correcto era la cadena de texto 'webfiltering_'. En esta versión se ha añadido.

3. El cuadro de texto guardaba todas las palabras introducidas, estuvieran o no almacenadas. De manera que, si un usuario buscaba varias veces la misma palabra, esta aparecía repetida en el desplegable, puesto que se guardaba cada vez que se buscaba.

   En esta versión la palabra solamente se guarda una vez, evitándose así que la misma palabra aparezca varias veces cuando se despliega el cuadro de texto.

4. Cuando una palabra no se encontraba en la página, ésta se quedaba completamente en blanco.

   En esta versión, si una palabra no se encuentra en la página, además de mostrar un mensaje al usuario se recarga la página original.

5. El formato de los iframes no se mantenía nunca.

   Aunque el usuario seleccionaba en el desplegable de opciones, mantener el formato de los iframes, estos no se mantenían. En esta versión se ha arreglado para que tenga en cuenta la preferencia del usuario, es decir, si decide no mantener el formato de los iframes, este no se mantenga y si decide mantenerlo que éste se mantenga.

3.3. MIGRACIÓN DE LA SEGUNDA VERSIÓN DEL NAVEGADOR FIREFOX A LA TERCERA

La versión 1.0 de la aplicación Web Filtering Toolbar estaba desarrollada para la segunda versión del navegador. Al publicarse la tercera versión del mismo se comprobó que dicha versión de la aplicación no funcionaba.
Se consultó la página Web que habilitó la comunidad Mozilla con todos los cambios introducidos en Firefox 3 y que pasos se debían seguir para poder hacer compatible las extensiones desarrolladas para Firefox 2.

Los cambios que se tuvieron que realizar en la versión 1.0 de la aplicación fueron los siguientes:

1. En el archivo install.rdf se tuvo cambiar la línea donde se define las compatibilidades:
   <em:maxVersion>2.0.0.*</em:maxVersion> por
   <em:maxVersion>3.0*</em:maxVersion>

2. En el archivo slicetoolbar.xul, los números del desplegable del cuadro de la tolerancia no se mostraban, ya que el elemento con el que se definían 'menuitem-iconic', ha desaparecido en Firefox 3. El equivalente a dicho elemento en el XUL para Firefox 3, es el elemento 'menuitem', por lo que se tuvo que sustituir todos los elementos definidos como 'menuitem-iconic' por 'menuitem'.

3.4. NUEVA FUNCIONALIDAD

Para este proyecto, además de la migración a la nueva versión del navegador Firefox y de corregir los errores de la versión 1.0, se ha desarrollado nueva funcionalidad útil para el usuario de la misma. Dicha funcionalidad se detalla a continuación:

1. Resaltar en rojo todas las apariciones de la palabra buscada dentro de la página Web.

2. Cuando la palabra introducida por el usuario no se encuentra en la página se muestra un mensaje informando al usuario de ello. Dicho mensaje, se muestra tanto si el usuario decide resaltar o filtrar.

3. Mantener el resaltado mientras se filtra y viceversa. Es decir, si previamente se ha resaltado una palabra y posteriormente el usuario decide filtrar por esa u otra palabra, el resaltado anterior se mantiene; y si primero se ha filtrado por una palabra y posteriormente el usuario decide resaltar la misma u otra palabra, el filtrado anterior se mantiene. (Veánse las figuras 13 y 14).

4. La aplicación es capaz de resaltar o filtrar páginas Webs que contengan frames.

5. Resaltar una a una todas las apariciones de la palabra, o bien hacía adelante (con el botón siguiente) o bien hacía atrás (con el botón anterior).

6. Mostrar u ocultar la barra a gusto del usuario. Pulsando la combinación de teclas 'Ctrl+Y' se muestra la barra, y pulsando sobre el botón 'Cerrar' se oculta.

7. No mantener el árbol de nodos del documento en páginas con frames.

8. Desmontar la estructura del cuerpo de cada frame.

4. IMPLEMENTACIÓN

En este punto se va a explicar con todo detalle las funciones implementadas, tanto las desarrolladas en la versión 1.0 de la Web Filtering Toolbar, como las desarrollas para el presente proyecto.
La explicación se va a desglosar función por función, adjuntando para cada una de ellas, su código fuente y una explicación detallada del mismo.

Para finalizar con este apartado, se explicarán las todas las herramientas utilizadas para llevar a cabo este proyecto.

4.1. EXPLICACIÓN DE LAS FUNCIONES PRINCIPALES

Se pasa a detallar toda la implementación de la aplicación, desglosada por funciones para facilitar su comprensión. Para cada función se adjunta su código fuente, una explicación de lo que hace, porque se ha optado por esa solución y, siempre que sea posible, el botón de la interfaz que la desencadena.

El orden que se va a seguir es el que se sigue en la interfaz, para que se sepa en todo momento de que funcionalidad se esta hablando.

4.1.1. VARIABLES GLOBALES

A continuación se detallan todas las variables globales usadas en la aplicación, así de para que se han utilizado.

```
var webfiltering_arrayocultos= new Array (); //vector que almacena todos los elementos que se ocultarán al aplicar el filtrado
var webfiltering_arraymuestra = new Array (); //vector que almacena todos los elementos que quedarán visibles al aplicar filtrado
var webfiltering_iglobal=0; //cuenta los elementos de arraymuestra
var webfiltering_jglobal=0; //cuenta los elementos de arrayocultos

//variables utilizas para las opciones de usuario
var webfiltering_tolerancia=0; //valor inicial de la tolerancia
var webfiltering_mantenerestructura=1; // 1 = true   0 = false
var webfiltering_formateartamanyoiframes=0; //vble que indica si se mantiene o no el formato de los iframes
var webfiltering_mantenerarbol = 1; //vble que indica si se mantiene el árbol o no
var webfiltering_mantenerbarravisible=1; //por defecto siempre estará visible

//vbles de uso general
var webfiltering_numPalabrasFiltradas=0;
var webfiltering_numPalabrasResaltadas=-1;
var webfiltering_palabrasBuscadas=new Array(); //vector que almacenara todas las palabras buscadas
var webfiltering_palabrasResaltadas=new Array(); //vector que almacena todas las palabras resaltadas (siguiente y anterior)
var webfiltering_sliceAplicado=0; //0: no se aplica slice, 1: sí se ha aplicado slice
//esta vble servirá para mantener el slice mientras se resaltan las coincidencias
var webfiltering_texto; //texto que hay en el cuadro
//var webfiltering_criteriobusqueda=null;
var webfiltering_palabraanterior=null; //vble usada para controlar en las funciones siguiente y anterior si se ha cambiado de palabra o no

var webfiltering_paginaOriginal=0;
```

```
var webfiltering_URLActual=0;
var webfiltering_frameOrig=0;
var webfiltering_vectorFrames=new Array();
var webfiltering_numFramesSinPalabra=0; //número de frames que no contienen la palabra buscada

var webfiltering_contador=0;
var webfiltering_contadorSig=0;
var webfiltering_contadorAnt=0;
var webfiltering_AplicadoSiguiente=0; //no se ha aplicado siguiente
var webfiltering_AplicadoAnterior=0; //no se ha aplicado siguiente

var webfiltering_AplicadoResaltar=0; //no se ha aplicado resaltar
var webfiltering_palabraEncontradaIframe=0;
var webfiltering_primerapasadaframes=0; //sirve para controlar la primera vez que se resalta los frames, para luego poder recuperar el original
var webfiltering_primerapasadaiframes=0; //sirve para controlar la primera vez que se resalta los iframes, para luego poder recuperar el original
var webfiltering_iframeOriginal=new Array(); //vector donde se almacenan todos los iframes originales. Sirve para recuperar el cuerpo del iframe original una vez se ha aplicado el resaltado
```

4.1.2. FUNCIÓN RESALTAR *(MATCH)*

Esta función se encarga de resaltar en rojo todas las apariciones que existan de la palabra introducida por el usuario.

4.1.2.1. EVENTO

Está función se dispara pulsando el botón 'HighLight' 

de la interfaz. Se encarga de llamar a la función 'webfiltering_match()', la cual, tras una serie de comprobaciones, se encarga de llamar a la función encargada de resaltar todas las coincidencias que aparecen de la palabra dentro de la página Web y devolverla al usuario.

4.1.2.2. FUNCIONAMIENTO

A continuación se muestra el código de la función:

```
function webfiltering_match(){ //funcion Resaltar coincidencias
    webfiltering_AplicadoResaltar=1;

    var vectorFrames =new Array(); //vector que almacenara el número de Frames
    var vectorIFrames =new Array(); //vector que almacenara el número de iFrames

    if(webfiltering_sliceAplicado==1){ //se ha aplicado slice
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        webfiltering_sliceAplicado=0;
    }
    else{
        if(webfiltering_paginaOriginal==0){ //guardo la página
                webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
                webfiltering_URLActual=webfiltering_guardarURL(content.document);
        }
    }

    if(webfiltering_mantenerestructura == 0){
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
    }
```

```
var urlActual=content.document.URL;
if(urlActual != webfiltering_URLActual){
     webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
     webfiltering_URLActual=webfiltering_guardarURL(content.document);
}

webfiltering_texto = document.getElementById("cuadroblanco").value
if (webfiltering_texto==""){
     alert("El criterio de busqueda es nulo, debe introducir al menos, una palabra");
     return;
}

var pag=webfiltering_paginaOriginal;
webfiltering_numPalabrasFiltradas++;
webfiltering_formateartamanyoiframes =
                        webfiltering_getIntegerPreference("slice.incrustacion",true);

vectorFrames=content.document.getElementsByTagName("FRAME");
if(vectorFrames.length > 0){
     var frame;
     var bodyFrame;
     var framesinresaltado;
     if( webfiltering_primerapasadaframes == 0 )
               for(var j=0; j < vectorFrames.length ; j++)
                        webfiltering_vectorFrames[j]=
                            webfiltering_guardarFrameOrig(vectorFrames[j]);
     webfiltering_primerapasadaframes=1;
     for(var i=0;i<vectorFrames.length;i++){
               frameOriginal= webfiltering_vectorFrames[i];
               frame=frameOriginal;
               bodyFrame=frame;
               bodyFrame=webfiltering_resaltar(bodyFrame,webfiltering_texto,null,null);
               vectorFrames[i].contentDocument.body.innerHTML=bodyFrame;
     }
}
else{
     var bodyText;
     vectorIFrames=content.document.getElementsByTagName("IFRAME");
     if(vectorIFrames.length > 0){
               webfiltering_resaltariframes(content.document,vectorIFrames);
               bodyText=content.document.body.innerHTML;
               bodyText=webfiltering_resaltar(bodyText,webfiltering_texto,null,null);
               if((bodyText == pag) && (webfiltering_palabraEncontradaIframe==0))
                        alert("La palabra buscada no se encuentra en la
                                 página. Revise la ortografia");
               content.document.body.innerHTML = bodyText; //reemplazamos  el
                                 body de la pagina, por el nuevo con las palabras resaltadas
      }
      else{
               bodyText=pag; // nuevo body que reemplaza al original
               bodyText=webfiltering_resaltar(bodyText,webfiltering_texto,null,null);
               if(bodyText == pag)
                        alert("La palabra buscada no se encuentra en la pagina.
                                 Revise la ortografia");
               content.document.body.innerHTML = bodyText; //reemplazamos  el
                        body de la pagina, por el nuevo con las palabras resaltadas
      }
}

//guarda un histórico de lo que se ha buscado
webfiltering_criteriobusqueda=document.getElementById("cuadroblanco");
if(webfiltering_numPalabrasFiltradas == 1){
```

```
        webfiltering_criteriobusqueda.appendItem(webfiltering_texto,webfiltering_texto);

        webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                                webfiltering_texto
    }
    else{
        var word=webfiltering_texto;
        var existe=false;//la palabra existe
        var i=1;
        while((i <= webfiltering_numPalabrasFiltradas) && (existe == false)){
                if(word == webfiltering_palabrasBuscadas[i] ) existe=true;
                i++;
        }
        if(existe == false){
                if(webfiltering_numPalabrasFiltradas > =10) {
                        webfiltering_criteriobusqueda.removeItemAt(0);
                }
                webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                                webfiltering_texto;
                webfiltering_criteriobusqueda.appendItem(webfiltering_texto,
                                                                webfiltering_texto);
        }
    }
}
```

Esta función realiza las siguientes tareas:

1. Pone la variable global 'AplicadoResaltar' a uno, para indicar que se ha pulsado el botón 'HighLight'.
2. Crea dos vectores locales que almacerán todos los frames e iframes que pueda tener la página Web para posteriormente poder procesarlos.
3. Realiza toda una serie de comprobaciones necesarias para el correcto funcionamiento de la aplicación:
    a. Comprueba que si se ha aplicado algún filtrado o no con anterioridad. Si sí se ha aplicado, la página a guardar será la resultante del filtrado previo, si no se ha aplicado se guarda la página Web original y la URL.
    b. Comprueba que si el usuario ha decidido no mantener la estructura de la página, ya que, en ese caso la página a guardar será la resultante del filtrado sin mantener la estructura.
    c. Comprueba, utilizando las URL's, si el usuario ha cambiado de página Web, ya que, en ese caso, habrá que guardar la nueva página.
    d. Comprueba si el usuario ha introducido una palabra en el cuadro de texto, avisando en caso contrario de ello.

```
        webfiltering_AplicadoResaltar=1;
        var vectorFrames =new Array(); //vector que almacenara el número de Frames
        var vectorIFrames =new Array(); //vector que almacenara el número de iFrames

        if(webfiltering_sliceAplicado==1){ //se ha aplicado slice
             webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
             webfiltering_sliceAplicado=0;
        }
        else{
             if(webfiltering_paginaOriginal==0){ //guardo la página
               webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
               webfiltering_URLActual=webfiltering_guardarURL(content.document);
             }
        }

        if(webfiltering_mantenerestructura == 0){
```

```
            webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        }

        var urlActual=content.document.URL;
        if(urlActual != webfiltering_URLActual){
            webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
            webfiltering_URLActual=webfiltering_guardarURL(content.document);
        }

        webfiltering_texto = document.getElementById("cuadroblanco").value
        if (webfiltering_texto==""){
           alert("El criterio de busqueda es nulo, debe introducir al menos, una palabra");
           return;
        }
```

4. Obtiene la preferencia del usuario para el formato de los iframes, mediante el método 'webfiltering_getIntegerPreference' y aumenta en uno el número de palabras filtradas (contador global del que hacen uso prácticamente todas las funciones), que sirve para llevar un control de las palabras buscadas.

```
        var pag=webfiltering_paginaOriginal;
        webfiltering_numPalabrasFiltradas++;
        webfiltering_formateartamanyoiframes =
                        webfiltering_getIntegerPreference("slice.incrustacion",true);
```

5. Mediante al método getElementsByTagName("FRAME") del estándar del DOM obtenemos todos los frames de la página Web.

   - Si la página tiene frames entonces se procede a procesar cada uno por separado, por tratarse cada uno de ellos de páginas diferentes dentro de la página Web, con su propio cuerpo. Para procesar los frames se hace uso de una variable global (webfiltering_primerapasadaframes) que indica si es la primera que vez que se van a resaltar los frames, ya que, en ese caso, se almacena en el vector (webfiltering_vectorFrames) el cuerpo original de cada uno de ellos, para poder recuperarlo posteriormente y así aplicar el resaltado siempre sobre el original.
     Una vez hecho esto se pasa a resaltar cada coincidencia de la palabra dentro de cada uno de los frames.
   - Si la página no tiene frames se pasa a comprobar si tiene iframes, del mismo modo como se ha hecho para los frames.
     - Si la página tiene iframes hay que resaltar las coincidencias, primero en el iframe y posteriormente en el resto de la página. Para ello se llama a la función 'webfiltering_resaltariframes' que será la encargada de devolver el iframe resaltado. Una vez se tiene el iframe resaltado se pasa a resaltar el resto de la página Web, haciendo uso de la función 'webfiltering_resaltar'.
       Si la palabra no se encuentra en la página se avisa al usuario de ello.
     - Si la página no tiene iframe solamente hay que resaltar las posibles coincidencias de la palabra dentro de la página Web, para ello se hace uso de la función, nombrada anteriormente, 'webfiltering_resaltar'.

```
        vectorFrames=content.document.getElementsByTagName("FRAME");
        if(vectorFrames.length > 0){
            var frame;
            var bodyFrame;
            var framesinresaltado;
            if( webfiltering_primerapasadaframes == 0 )
                for(var j=0; j < vectorFrames.length ; j++)
                  webfiltering_vectorFrames[j]=
```

```
                    webfiltering_guardarFrameOrig(vectorFrames[j]);
            webfiltering_primerapasadaframes=1;
            for(var i=0;i<vectorFrames.length;i++){
                frameOriginal= webfiltering_vectorFrames[i];
                frame=frameOriginal;
                bodyFrame=frame;
                bodyFrame=
                        webfiltering_resaltar(bodyFrame,webfiltering_texto,null,null);
                vectorFrames[i].contentDocument.body.innerHTML=bodyFrame;
            }
        }
        else{ //la página no tiene frames
             var bodyText;
             vectorIFrames=content.document.getElementsByTagName("IFRAME");
             if(vectorIFrames.length > 0){
                webfiltering_resaltariframes(content.document,vectorIFrames);
                bodyText=content.document.body.innerHTML;
                bodyText=webfiltering_resaltar(bodyText,webfiltering_texto,null,null);
                if((bodyText == pag) &&
                  (webfiltering_palabraEncontradaIframe==0))
                        alert("La palabra buscada no se encuentra en la
                                página. Revise la ortografia");
                        content.document.body.innerHTML = bodyText; //reemplazamos  el
                                body de la pagina, por el nuevo con las palabras resaltadas
            }
            else{
                bodyText=pag; //creamos un nuevo body para que reemplace al original
                bodyText=webfiltering_resaltar(bodyText,webfiltering_texto,null,null);
                if(bodyText == pag)
                        alert("La palabra buscada no se encuentra en la pagina.
                        Revise la ortografia");
                content.document.body.innerHTML = bodyText; //reemplazamos  el body de
                            la pagina, por el nuevo con las palabras resaltadas
            }
        }
```

6. Por último, guarda en la pestaña del cuadro de texto la palabra que se acaba de resaltar. Si la palabra ya está en la pestaña no se inserta de nuevo. Como máximo existirán 10 palabras insertadas en todo momento.

```
//guarda un histórico de lo que se ha buscado
webfiltering_criteriobusqueda=document.getElementById("cuadroblanco");
if(webfiltering_numPalabrasFiltradas == 1){
    webfiltering_criteriobusqueda.appendItem(webfiltering_texto,webfiltering_texto);

    webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                            webfiltering_texto
}
else{
    var word=webfiltering_texto;
    var existe=false;//la palabra existe
    var i=1;
    while((i <= webfiltering_numPalabrasFiltradas) && (existe == false)){
            if(word == webfiltering_palabrasBuscadas[i] ) existe=true;
            i++;
    }
    if(existe == false){
            if(webfiltering_numPalabrasFiltradas > =10) {
                    webfiltering_criteriobusqueda.removeItemAt(0);
            }
            webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                                webfiltering_texto;
```

```
                webfiltering_criteriobusqueda.appendItem(webfiltering_texto,
                                                                webfiltering_texto);
        }
    }
```

4.1.3. FUNCIONES *RESALTAR* Y *RESALTARIFRAMES*

En este punto se van a explicar las dos funciones nombradas en el punto anterior usadas para resaltar los iframes y la página normal.
Estas funciones se encargan de resaltar todas las coincidencias que aparezcan dentro de la página o dentro del iframe de la palabra buscada.

4.1.3.1. EVENTO

Estas funciones son llamadas por 'webfiltering_match()'. Ambas reciben el texto original, una el de la página Web y la otra del iframe, y devuelven otro con las coincidencias resaltadas.

4.1.3.2. FUNCIONAMIENTO

A continuación se adjunta el código de la función 'webfiltering_resaltar', que es la encargada de resaltar las coincidencias de la página Web original:

```
function webfiltering_resaltar(bodyText,searchTerm,highlightStartTag,highlightEndTag){
//bodyText es una copia de página original

    if ((!highlightStartTag) || (!highlightEndTag)) {
        highlightStartTag = "<font style='color:black; background-color:red;'>";
        highlightEndTag = "</font>";
    }

    var newText = ""; //vble que almacenara el nuevo body con las palabras resaltadas
    var i = -1;
    var textoMinus = searchTerm.toLowerCase();
    var bodyMinus = bodyText.toLowerCase();
    while (bodyText.length > 0) { //recorremos toda la pagina buscando coincidencias
        i = bodyMinus.indexOf(textoMinus, i+1);//almacenamos en i si hay coincidencia o no
        if (i < 0) { //no hay coincidencia porque el índice devuelto es menor que 0
                newText += bodyText;  //añadimos el texto original al nuevo texto
                bodyText = "";
        } else { //si ha habido coincidencia, en este caso tendremos que resaltarla
                if (bodyText.lastIndexOf(">", i) >= bodyText.lastIndexOf("<", i)) {
                        newText += bodyText.substring(0, i) + highlightStartTag +
                                bodyText.substr(i, searchTerm.length) + highlightEndTag;
                        bodyText = bodyText.substr(i + searchTerm.length);
                        bodyMinus = bodyText.toLowerCase();
                        i = -1; //igualamos a  -1 para continuar buscando coincidencias
                }
        }
     }
     return newText;
}
```

Recibe como parámetros de entrada una copia de la página original, la palabra a buscar y dos etiquetas (la de inicio de la palabra y la de final de palabra), aquellas que encerraran la coincidencia y se encargan de darle color de fondo.

1. Si las etiquetas de inicio y fin no tienen valor, lo primero que hace es inicializar poniendo como color de letra el negro y como fondo de la palabra

el rojo. Es decir, cada coincidencia aparecerá en un recuadro rojo y con las letras en negro.

```
if ((!highlightStartTag) || (!highlightEndTag)) {
    highlightStartTag = "<font style='color:black; background-color:red;'>";
    highlightEndTag = "</font>";
}
```

2. A continuación, crea una serie de variables locales.

```
var newText = "";  //vble que almacenará el nuevo body con las palabras resaltadas
var i = -1; //variable que almacenará la posición de la palabra buscada dentro del body
var textoMinus = searchTerm.toLowerCase(); //copia, en minúsculas, de la palabra buscada
var bodyMinus = bodyText.toLowerCase(); //copia, en minúsculas, del cuerpo de la página recibida como parámetro
```

3. Una vez tiene todas las variables creadas, recorre toda la página buscando coincidencias. Cada vez que encuentra una, añade al inicio y al final de la palabra las etiquetas, iguala la posición a menos uno (la variable i) para continuar con la búsqueda.
   Una vez recorrida toda la página, devuelve la nueva página con todas las etiquetas puestas.

```
while (bodyText.length > 0) { //recorremos toda la página buscando coincidencias
    i = bodyMinus.indexOf(textoMinus, i+1);//almacenamos en i si hay coincidencia o no
    if (i < 0) { //no hay coincidencia porque el índice devuelto es menor que 0
            newText += bodyText;  //añadimos el texto original al nuevo texto
            bodyText = "";
    } else { //si ha habido coincidencia, en este caso tendremos que resaltarla
            if (bodyText.lastIndexOf(">", i) >= bodyText.lastIndexOf("<", i)) {
                    newText += bodyText.substring(0, i) + highlightStartTag +
                            bodyText.substr(i, searchTerm.length) + highlightEndTag;
                    bodyText = bodyText.substr(i + searchTerm.length);
                    bodyMinus = bodyText.toLowerCase();
                    i = -1; //igualamos a  -1 para continuar buscando coincidencias
            }
    }
 }
 return newText; //devolvemos el texto con todas las etiquetas añadidas
```

A continuación se adjunta el código de la función 'webfiltering_resaltariframes', que es la encargada de resaltar las coincidencias dentro del iframe.

```
function webfiltering_resaltariframes(pagina,vector){
    var bodyIframe=null;
    var iframeOrig=null;
    var iframe=null;
    var altura,anchura,padreiframe,divNuevo;

    if( webfiltering_primerapasadaiframes == 0 ){
        for(var k=0; k < vector.length; k++) {
                webfiltering_iframeOriginal[k]=webfiltering_guardarIframeOrig(vector[k]);
        }
    }
    webfiltering_primerapasadaiframes=1;

    for(var i=0;i<vector.length;i++){
        if(webfiltering_formateartamanyoiframes== 1){
```

```
                altura=vector[i].height;
                anchura=vector[i].width;
                padreiframe = vector[i].parentNode;

                iframeOrig=webfiltering_iframeOriginal[i];
                                                //en iframeOrig tengo el iframe original
                iframe=iframeOrig;
                bodyIframe=iframe;
                bodyIframe=webfiltering_resaltar(bodyIframe,webfiltering_texto,null,null);
                if(bodyIframe != iframeOrig) webfiltering_palabraEncontradaIframe=1;
                vector[i].contentDocument.body.innerHTML=bodyIframe; //sustituyo el
                        cuerpo del iframe, por el nuevo resaltado

                divNuevo = pagina.createElement('div');
                divNuevo.id='nuevo';
                divNuevo.innerHTML =
                  vector[i].contentDocument.getElementsByTagName("body")[0].innerHTML;
                        //guardo en div el cuerpo resaltado del iframe
                divNuevo.height = altura;
                divNuevo.width = anchura;
                divNuevo.overflow="auto";

                padreiframe.replaceChild(divNuevo,vector[i]);
        }
        else{
                padreiframe = vector[i].parentNode;
                iframeOrig = webfiltering_iframeOriginal[i];
                iframe=iframeOrig;
                bodyIframe=iframe;
                bodyIframe=webfiltering_resaltar(bodyIframe,webfiltering_texto,null,null);
                if(bodyIframe != iframeOrig) webfiltering_palabraEncontradaIframe=1;
                vector[i].contentDocument.body.innerHTML=bodyIframe;
                padreiframe.replaceChild
                                (vector[i].contentDocument.body.innerHTML,vector[i]);
        }
    }
}
```

Recibe como parámetro la página original y el vector con los iframes, obtenido en la función 'webfiltering_match()'.

1. Esta función hace uso de la variable global 'webfiltering_primerapasadaiframes' que controla si el primera vez que se resalta el iframe, ya que, en ese caso se guarda en un vector el iframe original.
2. Si el usuario activa la opción de no mantener el formato de los iframes, para cada iframe se guarda su altura, anchura y su padre. Estas variables servirán para sustituir el iframe original de la página por la capa div cuyo contenido es el contenido del iframe resaltado. Si por el contrario, el usuario decide mantener el formato de los iframes, únicamente habrá que reemplazar el iframe original por el resaltdo. En este caso no es necesario recuperar su altura y su anchura, puesto que no se crea una capa div.
3. Para cada iframe se llama a la función 'webfiltering_resaltar', la cual, en este caso, recibe el cuerpo del iframe.
4. Por último, sustituye el iframe resaltado o la capa div por el original de la página.

4.1.4. FUNCIONES *ANTERIOR* Y *SIGUIENTE*

Por ser muy parecidas en comportamiento y en código solamente se va a explicar la función siguiente, pero siempre teniendo en cuenta, que la función anterior hace exactamente lo mismo pero partiendo desde la última palabra de la página y acabando en la primera.
La función siguiente se encarga de buscar, primeramente, todas las coincidencias de la palabra y posteriormente ir resaltándolas una a una, desde el inicio de la página al final.

4.1.4.1. EVENTO

El botón de la flecha hacia la izquierda  llama a la función 'webfiltering_siguiente' la cuál, tras una serie de comprobaciones, llama a las funciones 'webfiltering_buscarpalabra' y 'webfiltering_resaltarpalabrasiguiente'.

El botón de la flecha hacia la derecha  llama a la función 'webfiltering_anterior' la cuál, tras realizar las mismas comprobaciones que la función siguiente, llama a las funciones 'webfiltering_buscarpalabra' y 'webfiltering_resaltarpalabraanterior'.

4.1.4.2. FUNCIONAMIENTO

A continuación se muestra el código de la función siguiente (el de la función anterior se muestra en el anexo A).

```
function webfiltering_siguiente(){

    var filtrado=0; //variable que indica si se aplica filtrado o no
    webfiltering_AplicadoSiguiente=1; //se ha aplicado siguiente

    if(webfiltering_sliceAplicado==1){ //se ha aplicado slice
        filtrado=1; //indicamos que se ha aplicado un filtrado
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
                        //almacenamos la pagina filtrada anteriormente
        webfiltering_sliceAplicado=0;
    }
    else{
        if(webfiltering_paginaOriginal==0){ //me guardo  la pagina
                webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
                webfiltering_URLActual=webfiltering_guardarURL(content.document);
        }
    }
    var urlActual=content.document.URL;
    if(urlActual != webfiltering_URLActual){ //si son distintas URL es que se ha cambiado de
                                             página y la tengo que volver a guardar
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        webfiltering_URLActual=webfiltering_guardarURL(content.document);
    }

    webfiltering_texto = document.getElementById("cuadroblanco").value;
    if (webfiltering_texto==""){
        alert("El criterio de busqueda es nulo, debe introducir al menos, una palabra");
        return;
    }

    var pag=webfiltering_paginaOriginal; //en pag tengo una copia de la página original
    webfiltering_formateartamanyoiframes=
                                webfiltering_getIntegerPreference("slice.incrustacion",true);
```

```
    webfiltering_formateartamiframes(content.document);

    var bodyText;

    if( (webfiltering_texto != webfiltering_palabraanterior) ||
       (urlActual != webfiltering_URLActual) || (filtrado == 1)){
                webfiltering_contador=0;
                bodyText=pag; //creamos un nuevo body para que reemplace al original
                bodyText=webfiltering_buscarpalabra(bodyText,webfiltering_texto);
    }
    else {
         if(webfiltering_AplicadoAnterior == 1){
                webfiltering_contador=webfiltering_contadorAnt+1;
                webfiltering_AplicadoAnterior=0; //reemplazamos  el body de la
                                                 página, por el nuevo con las anclas
         }
         bodyText= content.document.body.innerHTML; //tengo la misma palabra, por lo
                                                    tanto, guardo el texto con las anclas
    }

    if(webfiltering_palabrasResaltadas.length==0 ||
      webfiltering_numPalabrasResaltadas==0)
         alert("La palabra buscada no se encuentra en la pagina");

    content.document.body.innerHTML = bodyText;
    webfiltering_palabraanterior=document.getElementById("cuadroblanco").value;
    webfiltering_numPalabrasFiltradas++;

    //guarda un histórico de lo que se ha buscado
    webfiltering_criteriobusqueda=document.getElementById("cuadroblanco");
    if(webfiltering_numPalabrasFiltradas == 1){
         webfiltering_criteriobusqueda.appendItem(webfiltering_texto,webfiltering_texto);
         webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                              webfiltering_texto; //almacena la palabra buscada en el vector

    }
    else{
         var word=webfiltering_texto;
         var existe=false; //la palabra existe
         var i=1;
         while((i <= webfiltering_numPalabrasFiltradas) && (existe == false)){
                if(word == webfiltering_palabrasBuscadas[i] ) existe=true;
                i++;
         }
         if(existe == false){
                if(webfiltering_numPalabrasFiltradas >= 10) {
                       webfiltering_criteriobusqueda.removeItemAt(0);
                }
                 webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                                  webfiltering_texto;
                 webfiltering_criteriobusqueda.appendItem
                                                 (webfiltering_texto,webfiltering_texto);
         }
    }
    webfiltering_resaltarpalabrasiguiente(content.document,webfiltering_contador);
}
```

1. Esta función hace uso de una variable global que indica si se ha pulsado el botón siguiente o no. Se utiliza en la función 'webfiltering_resaltarpalabraanterior()' e indica a la función

'webfiltering_anterior()' que tiene que partir de la palabra donde se ha quedado siguiente e ir hacia el inicio de la página.

```
webfiltering_AplicadoSiguiente=1; //se ha aplicado siguiente
```

2. Comprobaciones que realiza:
    a. Comprueba si se ha aplicado un filtrado con anterioridad, ya que, si se ha aplicado habrá que guardar la página filtrada y poner la variable global que indica si se ha aplicado filtrado a cero, y si no se aplica, habrá que guardar la página Web original.
    b. Comprueba si el usuario cambia de página comparando las URL's, porque si cambia hay que volver a guardar la nueva página Web original.
    c. Comprueba que el usuario haya introducido una palabra en el cuadro de texto. Si no lo ha hecho se muestra un mensaje por pantalla avisándole de ello.

```
if(webfiltering_sliceAplicado==1){ //se ha aplicado slice
    filtrado=1;
    webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
                    //almacenamos la página filtrada anteriormente
    webfiltering_sliceAplicado=0;
}
else{
    if(webfiltering_paginaOriginal==0){ //me guardo la página original
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        webfiltering_URLActual=webfiltering_guardarURL(content.document);
    }
}
var urlActual=content.document.URL;
if(urlActual != webfiltering_URLActual){ //si son distintas URL es que se ha cambiado de
                                        página y la tengo que volver a guardar
    webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
    webfiltering_URLActual=webfiltering_guardarURL(content.document);
}

webfiltering_texto = document.getElementById("cuadroblanco").value;
if (webfiltering_texto==""){
    alert("El criterio de busqueda es nulo, debe introducir al menos, una palabra");
    return;
}
```

3. Obtiene la preferencia del usuario para el formato de los iframes, mediante el método 'webfiltering_getIntegerPreference'. Llama a la función 'webfiltering_formateartamiframes' que es la encargada de procesar los iframes según el valor de la preferencia del usuario.

```
var pag=webfiltering_paginaOriginal; //en pag tengo una copia de la página original
webfiltering_formateartamanyoiframes=
                        webfiltering_getIntegerPreference("slice.incrustacion",true);

webfiltering_formateartamiframes(content.document);
```

4. Para poder identificar si el usuario cambia de palabra o no, compara la palabra introducida actualmente (webfiltering_texto) con la introducida anteriormente (webfiltering_palabraanterior, inicialmente no tiene valor).
   Si ambas coinciden significa que el usuario desea avanzar por las diferentes coincidencias de la palabra.

En el caso de que el usuario decida moverse por las distintas coincidencias, se comprueba si ha pulsado previamente el botón anterior, ya que, en ese caso, el contador de siguiente tendrá el valor del contador de anterior más uno, para continuar a partir de la palabra resaltada con anterior hacia al final de la página.
Si el usuario cambia de palabra, se recupera la página Web original, se llama de nuevo a la función 'webfiltering_buscarpalabra' con la nueva palabra y se inicializa el contador a cero para empezar desde la primera hasta la última.

```
var bodyText;

if( (webfiltering_texto != webfiltering_palabraanterior) ||
   (urlActual != webfiltering_URLActual) || (filtrado == 1)){
            webfiltering_contador=0;
            bodyText=pag; //creamos un nuevo body para que reemplace al original
            bodyText=webfiltering_buscarpalabra(bodyText,webfiltering_texto);
}
else {
    if(webfiltering_AplicadoAnterior == 1){
            webfiltering_contador=webfiltering_contadorAnt+1;
            webfiltering_AplicadoAnterior=0; //reemplazamos  el body de la
                                              página, por el nuevo con las anclas
    }
    bodyText= content.document.body.innerHTML; //tengo la misma palabra, por lo
                                                 tanto, guardo el texto con las anclas
}
```

5. Si la palabra no se encuentra en la página avisa al usuario de ello.

```
if(webfiltering_palabrasResaltadas.length==0 ||
  webfiltering_numPalabrasResaltadas==0)
    alert("La palabra buscada no se encuentra en la pagina");
```

6. Sustituye la página Web original por la página devuelta por la función 'webfiltering_buscarpalabra' y aumenta el contador global de palabras buscadas por el usuario en uno.

```
content.document.body.innerHTML = bodyText;
webfiltering_palabraanterior=document.getElementById("cuadroblanco").value;
webfiltering_numPalabrasFiltradas++;
```

7. Si la palabra buscada no está guardada en el cuadro de texto la guarda, si ya existe no la almacena.

```
//guarda un histórico de lo que se ha buscado
webfiltering_criteriobusqueda=document.getElementById("cuadroblanco");
if(webfiltering_numPalabrasFiltradas == 1){
    webfiltering_criteriobusqueda.appendItem(webfiltering_texto,webfiltering_texto);

    webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                        webfiltering_texto
}
else{
    var word=webfiltering_texto;
    var existe=false;//la palabra existe
    var i=1;
    while((i <= webfiltering_numPalabrasFiltradas) && (existe == false)){
            if(word == webfiltering_palabrasBuscadas[i] ) existe=true;
            i++;
    }
```

```
        if(existe == false){
                if(webfiltering_numPalabrasFiltradas > =10) {
                        webfiltering_criteriobusqueda.removeItemAt(0);
                }
                webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                                webfiltering_texto;
                webfiltering_criteriobusqueda.appendItem(webfiltering_texto,
                                                                webfiltering_texto);
        }
    }
```

8. Por último, se encarga de llamar a la función que irá resaltando cada una de las coincidencias de la palabra.

```
webfiltering_resaltarpalabrasiguiente(content.document,webfiltering_contador);
```

4.1.5. FUNCIÓN *BUSCARPALABRA*

Esta función la utilizan tanto anterior como siguiente para buscar todas las coincidencias de la palabra buscada dentro de la página.
La función 'webfiltering_buscarpalabra' pone un ancla en cada una de las posiciones donde aparece la palabra buscada por el usuario. De este modo, cada vez que se pulse anterior o siguiente, simplemente bastará con saltar al ancla correspondiente.

4.1.5.1. EVENTO

Como se ya se ha comentado, esta función es llamada, o bien desde 'webfiltering_siguiente' o bien desde 'webfiltering_anterior'.

4.1.5.2. FUNCIONAMIENTO

A continuación se muestra el código de la misma:

```
function webfiltering_buscarpalabra(bodyText,searchTerm){

    var cierraAncla="</a>";

    var newText = ""; //vble que almacenara el nuevo body con las palabras resaltadas
    var i = -1;
    var textoMinus = searchTerm.toLowerCase();
    var bodyMinus = bodyText.toLowerCase();

    while(bodyText.length > 0){
        i = bodyMinus.indexOf(textoMinus, i+1);
        if(i < 0){
                newText += bodyText; //añadimos el texto original al nuevo texto
                bodyText = "";
        }
        else{
                if(bodyText.lastIndexOf(">", i) >= bodyText.lastIndexOf("<", i)){
                    webfiltering_numPalabrasResaltadas++;
                    newText += bodyText.substring(0, i) + '<a name="i">' +
                                        bodyText.substr(i, searchTerm.length) + cierraAncla;
                    webfiltering_palabrasResaltadas[webfiltering_numPalabrasResaltadas]=i;
                            //guardamos la posicion de la palabra buscada
                    bodyText = bodyText.substr(i + searchTerm.length);
                    bodyMinus = bodyText.toLowerCase();
                    i = -1; //volvemos a poner la i a -1 para continuar buscando coincidencias
```

```
                }
            }
        }
        return newText;
}
```

Por ser muy parecida a la función 'webfiltering_resaltar()' no se va a explicar su funcionamiento. La única diferencia es que ésta, en lugar de poner una etiqueta de color al inicio y al final de la palabra, coloca un ancla en ambas posiciones y almacena en un vector las posiciones donde se ha encontrado una coincidencia.

4.1.6. FUNCIÓN *RESALTARPALABRASIGUIENTE*

Esta función se encarga de ir resaltando la coincidencia correspondiente dentro de la página devuelta por la función 'webfiltering_buscarpalabra'.

4.1.6.1. EVENTO

Esta función es invocada desde la función 'webfiltering_siguiente()'.

4.1.6.2. FUNCIONAMIENTO

A continuación se muestra su código. La función 'webfiltering_resaltarpalabraanterior()' puede consultarse en el anexo A.

```
function webfiltering_resaltarpalabrasiguiente(bodyText,contador){

        var numeroAnclas = bodyText.anchors.length;

        if(numeroAnclas == 0) return;

        if(webfiltering_AplicadoAnterior == 1){
                bodyText.anchors[contador].style.background="none";
        }

        if(contador == numeroAnclas) {
                bodyText.anchors[contador-1].style.background="none";
                contador=0; //lo igualamos a 0 par volver a empezar
                webfiltering_contador=contador;
        }

        var devolver = webfiltering_palabrasResaltadas[contador];

        bodyText.anchors[contador].setAttribute('href','#devolver');

        if(contador > 0){
                bodyText.anchors[contador-1].style.background="none";
        }

        bodyText.anchors[contador].style.background="yellow";

        if(contador != numeroAnclas) {
                webfiltering_contadorSig=webfiltering_contador;
                webfiltering_contador++;
        }
        return bodytext;
}
```

Recibe como parámetros la página con las anclas colocadas en las posiciones donde ha habido una coincidencia, y el número de la palabra que toca resaltar.

1. Se debe controlar cuando se resalta la última coincidencia, ya que, cuando eso ocurra se debe volver a la primera de ellas para que el usuario pueda seguir moviéndose por las mismas.
   Para ello se obtiene el número total de anclas colocadas en la página, mediante el método 'anchors'.

```
var numeroAnclas = bodyText.anchors.length;
```

2. Si la palabra no se encontraba en la página, el número de anclas colocadas será cero. En ese caso, se volverá se opta por volver a la función 'webfiltering_siguiente()' para no bloquear la aplicación.

```
f(numeroAnclas == 0) return;
```

3. Si previamente el usuario ha pulsado anterior, la palabra resaltada debe volver a su estado original (sin el resaltado) y continuar con la siguiente.

```
if(webfiltering_AplicadoAnterior == 1){
        bodyText.anchors[contador].style.background="none";
}
```

4. Cuando el contador llegue al final, es decir, se haya resaltado la última coincidencia, hay que ponerlo a cero, otra vez, para volver a comenzar desde el principio de la página.

```
if(contador == numeroAnclas) {
        bodyText.anchors[contador-1].style.background="none";
        contador=0; //lo igualamos a 0 par volver a empezar
        webfiltering_contador=contador;
}
```

5. Se obtiene la posición de la palabra que toca resaltar a continuación, para poder saltar al ancla correspondiente.
   Cuando el contador es mayor que cero, significa que ya se ha resaltado una palabra y se tiene que saltar a la siguiente pero dejando la palabra anterior como estaba originalmente, para que en todo momento solamente haya una palabra resaltada.
   Una vez la palabra anterior ha vuelto a su estado original, se resalta la que toca en amarillo.

```
var devolver = webfiltering_palabrasResaltadas[contador];

bodyText.anchors[contador].setAttribute('href','#devolver');

if(contador > 0){
        bodyText.anchors[contador-1].style.background="none";
}

bodyText.anchors[contador].style.background="yellow";
```

6. Por último, mientras el contador sea distinto del número total de anclas colocadas, el contador que controla por que palabra se queda la función siguiente se iguala al contador global y posteriormente el contador global de posiciones ('webfiltering_contador') se aumenta en uno.

Como resultado devuelve la página únicamente con la coincidencia que corresponda resaltada en amarillo.

```
if(contador != numeroAnclas) {
            webfiltering_contadorSig=webfiltering_contador;
            webfiltering_contador++;
}
return bodytext;
```

4.1.7. FUNCIÓN PRINCIPAL DE FILTRADO *(FUNCSLICE)*

Esta función inicializa las opciones, llama a las funciones encargadas de procesarlas para obtener un filtrado según las mismas y por último guarda la palabra a filtrar.

4.1.7.1. EVENTO

El botón 'Filter'  llama a la función 'webfiltering_funcslice()' que se encarga de obtener las preferencias seleccionadas por el usuario, llamar a las distintas funciones encargadas de procesar dichas preferencias para obtener un filtrado en función de las mismas y por último guarda la palabra a filtrar en el cuadro de texto.

4.1.7.2. FUNCIONAMIENTO

A continuación se muestra el código de la misma:

```
function webfiltering_funcslice(){ //función principal, se activa en el botón filter
    webfiltering_sliceAplicado=1; //ya se ha aplicado un filtrado
    webfiltering_mostrarocultos();
    webfiltering_arrayocultos= new Array (); //inicializar arrayocultos para no mantener
filtrados anteriores
    webfiltering_arraymuestra = new Array (); //inicializar arraymuestra para no mantener
filtrados anteriores

    //este if solo se ejecutara cuando se trate de la primera página
    if(webfiltering_paginaOriginal==0){ //me guardo  la página
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        webfiltering_URLActual=webfiltering_guardarURL(content.document);
    }

    var urlActual=content.document.URL;
    if(urlActual != webfiltering_URLActual){
    //si son distintas URL es que se ha cambiado de pagina y la tengo que volver a guardar
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        webfiltering_URLActual=webfiltering_guardarURL(content.document);
    }

    //guarda en la variable webfiltering_texto el contenido del cuadro de texto
    webfiltering_texto = document.getElementById("cuadroblanco").value;
    if (webfiltering_texto=="") {
        alert("El criterio de busqueda es nulo, debe introducir al menos, una palabra");
        return;
    }

    //inicializa las opciones
    webfiltering_mantenerestructura =
                        webfiltering_getIntegerPreference("slice.mantenerestructura",true);
    webfiltering_tolerancia = webfiltering_getIntegerPreference("slice.tolerance",true);
```

```
    webfiltering_formateartamanyoiframes =
                        webfiltering_getIntegerPreference("slice.incrustacion",true);
     webfiltering_mantenerarbol = webfiltering_getIntegerPreference("slice.keeptree",true);

    webfiltering_numPalabrasFiltradas++;
    webfiltering_formateartamiframes(content.document);
    if (webfiltering_mantenerarbol==0)
        webfiltering_nokeeptree();
    else   webfiltering_ejecucionfunc(content.document);

   // mostramos los nodos del árbol que estén en el arraymuestra (teniendo en cuenta la
tolerancia)
    webfiltering_mostrarcontolerancia();

    //guarda un histórico de lo que se ha buscado
    webfiltering_criteriobusqueda=document.getElementById("cuadroblanco");
    if(webfiltering_numPalabrasFiltradas == 1){
        webfiltering_criteriobusqueda.appendItem(webfiltering_texto,webfiltering_texto);
        webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                            webfiltering_texto; //almacena la palabra buscada en el vector

    }
    else{
        var word=webfiltering_texto;
        var existe=false; //la palabra existe
        var i=1;
        while((i <= webfiltering_numPalabrasFiltradas) && (existe == false)){
                if(word == webfiltering_palabrasBuscadas[i] ) existe=true;
                i++;
        }
        if(existe == false){
                if(webfiltering_numPalabrasFiltradas >= 10) {
                        webfiltering_criteriobusqueda.removeItemAt(0);
                }
                webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                                webfiltering_texto;
                webfiltering_criteriobusqueda.appendItem
                                                (webfiltering_texto,webfiltering_texto);
        }
    }
}
```

1. Indica, mediante la variable global, que se va a aplicar un filtrado.
2. Muestra todos los elementos que puedan estar ocultos de algún filtrado anterior.
3. Inicializa los vectores que almacenarán los elementos a mostrar y a ocultar, para no tener acumulados los elementos de filtrados anteriores.

```
webfiltering_sliceAplicado=1; //ya se ha aplicado un filtrado
    webfiltering_mostrarocultos();
    webfiltering_arrayocultos= new Array (); //inicializar arrayocultos para no mantener
filtrados anteriores
    webfiltering_arraymuestra = new Array (); //inicializar arraymuestra para no mantener
filtrados anteriores
```

4. Realiza las siguientes comprobaciones:
    a. Comprueba si se trata de la primera página, ya que, en ese caso, siempre se guardará.
    b. Comprueba si el usuario cambia de página, comparando la URL actual con la nueva. Si se produce cambio se almacena la nueva URL y el contenido de la nueva página.

c. Comprueba si el usuario ha escrito alguna palabra en el cuadro de texto. Si no lo ha hecho se le informa de ello y se termina la función.

```
//este if solo se ejecutara cuando se trate de la primera página
    if(webfiltering_paginaOriginal==0){ //me guardo  la página
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        webfiltering_URLActual=webfiltering_guardarURL(content.document);
    }

    var urlActual=content.document.URL;
    if(urlActual != webfiltering_URLActual){
    //si son distintas URL es que se ha cambiado de pagina y la tengo que volver a guardar
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        webfiltering_URLActual=webfiltering_guardarURL(content.document);
    }

    //guarda en la variable webfiltering_texto el contenido del cuadro de texto
    webfiltering_texto = document.getElementById("cuadroblanco").value;
    if (webfiltering_texto=="") {
        alert("El criterio de busqueda es nulo, debe introducir al menos, una palabra");
        return;
    }
```

5. Inicializa las opciones con el método 'getIntergerPreference' del software libre Webdeveloper.

```
    //inicializa las opciones
    webfiltering_mantenerestructura =
                        webfiltering_getIntegerPreference("slice.mantenerestructura",true);
    webfiltering_tolerancia = webfiltering_getIntegerPreference("slice.tolerance",true);
    webfiltering_formateartamanyoiframes =
                        webfiltering_getIntegerPreference("slice.incrustacion",true);
     webfiltering_mantenerarbol = webfiltering_getIntegerPreference("slice.keeptree",true);
```

6. Se aumenta el número de palabras filtradas. Se llama a la función encargada de procesar los iframes y en función de si el usuario decide mantener árbol de elementos de la página o no, se llama a la función 'webfiltering_nokeeptree()' o a 'webfiltering_ejecucionfunc()'.

```
    webfiltering_numPalabrasFiltradas++;
    webfiltering_formateartamiframes(content.document);
    if (webfiltering_mantenerarbol==0)
        webfiltering_nokeeptree();
    else   webfiltering_ejecucionfunc(content.document);
```

7. Una vez se tiene la página filtrada, para poder mostrarla por pantalla al usuario hay que comprobar que nivel de tolerancia desea, ya que, la cantidad de información a mostrar varía en función de ese valor.

```
   // mostramos los nodos del árbol que estén en el arraymuestra (teniendo en cuenta la
tolerancia)
    webfiltering_mostrarcontolerancia();
```

8. Por ultimo, guarda la palabra introducida por el usuario en el cuadro de texto, siempre y cuando no existiera con anterioridad.

```
    //guarda un histórico de lo que se ha buscado
    webfiltering_criteriobusqueda=document.getElementById("cuadroblanco");
    if(webfiltering_numPalabrasFiltradas == 1){
        webfiltering_criteriobusqueda.appendItem(webfiltering_texto,webfiltering_texto);
```

```
        webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                webfiltering_texto; //almacena la palabra buscada en el vector

    }
    else{
        var word=webfiltering_texto;
        var existe=false; //la palabra existe
        var i=1;
        while((i <= webfiltering_numPalabrasFiltradas) && (existe == false)){
                if(word == webfiltering_palabrasBuscadas[i] ) existe=true;
                i++;
        }
        if(existe == false){
                if(webfiltering_numPalabrasFiltradas >= 10) {
                        webfiltering_criteriobusqueda.removeItemAt(0);
                }
                webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                                        webfiltering_texto;
                webfiltering_criteriobusqueda.appendItem
                                                (webfiltering_texto,webfiltering_texto);
        }
    }
```

4.1.8. FUNCIÓN *FORMATEARTAMIFRAMES*

Se encarga de procesar el tamaño de todos los iframes que pueda contener la página, según el valor de la preferencia introducida por el usuario.
Si el usuario decide mantener el tamaño original de los iframes, estos mantendrán, si la tienen y es necesario, las barras de desplazamiento.
Por el contrario si el usuario decide no mantener su tamaño, todos los iframes perderán las barras de desplazamiento.

4.1.8.1. EVENTO

Esta función se invoca desde 'webfiltering_funcslice()' y se encarga de modificar los iframes según la preferencia del usuario.

4.1.8.2. FUNCIONAMIENTO

A continuación se puede observar el código de la misma:

```
function webfiltering_formateartamiframes(pagina){
    var array=new Array();
    var altura,anchura;
    var padreiframe=0;
    var divNuevo;

    array=pagina.getElementsByTagName("IFRAME");
    for(var i=0;i<array.length;i++){
        if(webfiltering_formateartamanyoiframes == 0){
            var arbol = document.createTreeWalker(array[i].contentDocument.body,
                        NodeFilter.SHOW_ELEMENT, null, false);
            while(arbol.nextNode()) {
                if(arbol.currentNode.tagName == "IMG"){
                    if((arbol.currentNode.alt.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1) &&
                      (arbol.currentNode.name.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1) &&
```

```
                    (arbol.currentNode.longDesc.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1) &&
                    (arbol.currentNode.src.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1))
                        webfiltering_ocultarnodos(arbol.currentNode);
                    else{
                        webfiltering_arraymuestra[webfiltering_iglobal] = arbol.currentNode;
                        webfiltering_iglobal++;
                    }
                } //if de IMG
                else{
                    if(arbol.currentNode.innerHTML.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1 &&
                        arbol.currentNode.tagName != "IFRAME")
                        webfiltering_ocultarnodos(arbol.currentNode);
                    else{
                        webfiltering_arraymuestra[webfiltering_iglobal]= arbol.currentNode;
                        webfiltering_iglobal++;
                    }
                }
            } //cierra el while
        } //cierra if no mantener iframes
        if(webfiltering_formateartamanyoiframes == 1){
                altura = array[i].height; //obtenemos la altura del iframe
                anchura = array[i].width; //obtenemos la anchura la iframe
                padreiframe = array[i].parentNode; //guardamos el padre del iframe

                divNuevo = pagina.createElement('div'); //crea una capa div
                divNuevo.id='nuevo';
                divNuevo.innerHTML =
                    array[i].contentDocument.getElementsByTagName("body")[0].innerHTML;
                    //le añade al nuevo div el contenido del iframe
                divNuevo.height = altura;
                divNuevo.width = anchura;
                divNuevo.overflow="auto";

                padreiframe.replaceChild(divNuevo,array[i]); //reemplaza el iframe por el div
        } //cierra if de mantener iframes
    } //for
}
```

Recibe como parámetro la página Web.

1. Obtiene todos los iframes que pueda tener la página Web con el método getElementsByTagName del estándar DOM y los guarda en un vector.
2. Todos los iframes de la página se van a tratar de la misma manera, en función de la opción escogida por el usuario.
3. Si decide no mantener el tamaño de los iframes:
    a. Crea un treewalker con el contenido del iframe.
    b. Para cado nodo del treeWalker se pregunta si es una imagen o es texto.
    c. Si es una imagen se mira si en cualquiera de estos atributos: name, alt, longDesc o src aparece la palabra buscada, en ese caso se mostrará la imagen. Por el contrario, si no aparece en ninguno de ellos la imagen se oculta.
    d. Si es texto se mira si contiene la palabra buscada y que no sea un iframe. Si la contiene ese nodo se mostrará, sino la contiene no se mostrará.

```
        while(arbol.nextNode()) {
            if(arbol.currentNode.tagName == "IMG"){
```

```
                if((arbol.currentNode.alt.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1) &&
                   (arbol.currentNode.name.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1) &&
                   (arbol.currentNode.longDesc.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1) &&
                   (arbol.currentNode.src.toLowerCase().
                                 search(webfiltering_texto.toLowerCase()) == -1))
                     webfiltering_ocultarnodos(arbol.currentNode);
                else{
                     webfiltering_arraymuestra[webfiltering_iglobal] = arbol.currentNode;
                     webfiltering_iglobal++;
                 }
              } //if de IMG
             else{
                if(arbol.currentNode.innerHTML.toLowerCase().
                           search(webfiltering_texto.toLowerCase()) == -1 &&
                  arbol.currentNode.tagName != "IFRAME")
                     webfiltering_ocultarnodos(arbol.currentNode);
                else{
                     webfiltering_arraymuestra[webfiltering_iglobal]= arbol.currentNode;
                     webfiltering_iglobal++;
                }
             }
           } //cierra el while
        } //cierra if no mantener iframes
```

4. Si decide no mantener el tamaño:
    a. Se obtiene la altura y anchura originales del iframe, así como el nodo padre del mismo.
    b. Se crea una capa DIV (con su identificador), que servirá para sustituir el iframe en la página Web original.
    c. Como contenido de la capa DIV se copia el contenido del iframe.
    d. Como altura y anchura de la capa las mismas que tiene el iframe.
    e. La propiedad 'overflow' indica si se mostrará o no barra de desplazamiento. Los posibles valores que puede tener son: 'yes', se mostrará siempre, 'no', no se mostrará nunca y 'auto', se mostrará cuando sea necesario (es decir, cuando la información contenida en el iframe no quepa en la capa div creada).
    f. Por último sustituye el iframe por la nueva capa DIV creada.

```
        if(webfiltering_formateartamanyoiframes == 1){
              altura = array[i].height; //obtenemos la altura del iframe
              anchura = array[i].width; //obtenemos la anchura la iframe
              padreiframe = array[i].parentNode; //guardamos el padre del iframe

              divNuevo = pagina.createElement('div'); //crea una capa div
              divNuevo.id='nuevo';
              divNuevo.innerHTML =
                 array[i].contentDocument.getElementsByTagName("body")[0].innerHTML;
                 //le añade al nuevo div el contenido del iframe
              divNuevo.height = altura;
              divNuevo.width = anchura;
              divNuevo.overflow="auto";

              padreiframe.replaceChild(divNuevo,array[i]); //reemplaza el iframe por el div
        } //cierra if de mantener iframes
```

4.1.9. FUNCIÓN QUE APLICA EL FILTRADO *(EJECUCIONFUNC)*

Esta función es la encargada de realizar el filtrado. Oculta aquellos nodos que no son útiles para el usuario y muestra aquellos que si lo son.

4.1.9.1. EVENTO

Es invocada desde la función 'webfiltering_funcslice' cuando la opción de mantener árbol está seleccionada.

4.1.9.2. FUNCIONAMIENTO

A continuación se muestra el código de la misma:

```
function webfiltering_ejecucionfunc(pagina){ //funcion encargada de hacer el filtrado
    var treeWalker,arbolFrame;
    var cuerpoFrame;
    var nodo;

    //crea un treeWalker con el contenido de la pagina y va recorriendo los nodos
    treeWalker= document.createTreeWalker(pagina.documentElement,
                                          NodeFilter.SHOW_ELEMENT, null, false);
    while(nodo = treeWalker.nextNode()){
        if(nodo.tagName=="FRAME"){
                cuerpoFrame = nodo.contentDocument.body.innerHTML;
                if(cuerpoFrame.toLowerCase().search(webfiltering_texto.toLowerCase()) ==
                   -1){
                        webfiltering_ocultarnodos(nodo);
                }
                else{
                    webfiltering_arraymuestra[webfiltering_iglobal] = nodo;
                    webfiltering_iglobal++;
                    //crea un treeWalker con el contenido del frame y se va recorriendo
                    arbolFrame=document.createTreeWalker
                       (nodo.contentDocument.body, NodeFilter.SHOW_ELEMENT, null, false);

                    while(arbolFrame.nextNode()){
                        if(arbolFrame.currentNode.tagName == "IMG"){
                            if((arbolFrame.currentNode.alt.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1) &&
                               (arbolFrame.currentNode.name.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1)&&
                               (arbolFrame.currentNode.longDesc.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1) &&
                               (arbolFrame.currentNode.src.toLowerCase().
                                search(webfiltering_texto.toLowerCase()) == -1))
                                        webfiltering_ocultarnodos(arbolFrame.currentNode);
                            else{
                                        webfiltering_arraymuestra[webfiltering_iglobal] =
                                                arbolFrame.currentNode;
                                        webfiltering_iglobal++;
                            }
                        }//IMG
                        else{
                            if(arbolFrame.currentNode.innerHTML.toLowerCase().
                               search(webfiltering_texto.toLowerCase()) == -1 &&
                               arbolFrame.currentNode.tagName != "IFRAME")
                                        webfiltering_ocultarnodos(arbolFrame.currentNode);
                            else{
                                        webfiltering_arraymuestra[webfiltering_iglobal] =
                                                    arbolFrame.currentNode;
```

```
                                        webfiltering_iglobal++;
                                    }
                                } //no IMG
                            } //while del frame con la palabra
                        } //frame con la palabra
                } //if FRAME
                else{
                    if(nodo.tagName=="IMG"){
                      if((nodo.alt.toLowerCase().search(webfiltering_texto.toLowerCase())==-1)
                        &&
                       (nodo.name.toLowerCase().search(webfiltering_texto.toLowerCase())==-1)
                        &&
                       (nodo.longDesc.toLowerCase().search(webfiltering_texto.toLowerCase())==-
                        1) &&
                       (nodo.src.toLowerCase().search(webfiltering_texto.toLowerCase())==-1)){
                                webfiltering_ocultarnodos(nodo);
                     }
                      else{
                                //sino lo guarda en el arraymuestra
                                webfiltering_arraymuestra[webfiltering_iglobal] = nodo;
                                webfiltering_iglobal++;
                      }
                    }//IMG
                    else{
                        if(nodo.innerHTML.toLowerCase().search
                          (webfiltering_texto.toLowerCase())==-1  && nodo.tagName != "IFRAME"){
                                webfiltering_ocultarnodos(nodo);
                         }
                         else{
                                webfiltering_arraymuestra[webfiltering_iglobal] = nodo;
                                webfiltering_iglobal++;
                         }
                     } //no IMG
                } //no FRAME
        } //se cierra while
        if(webfiltering_arraymuestra.length == 0){
            alert("La palabra buscada no se encuentra en la pagina. Revise la ortografia");
            content.window.location.reload();
        }
}
```

1. Crea un treeWalker (estructura en forma de árbol que contiene todos los elementos (nodos) de una página de manera jerarquizada. Es muy útil esta estructura porque es sencillo recorrerla, tanto hacia las hojas como hacia la raíz), con el contenido de la página. De este modo, para ver si algún elemento contiene la palabra, bastará con buscarla dentro de cada nodo del árbol.

```
treeWalker= document.createTreeWalker(pagina.documentElement,
                                        NodeFilter.SHOW_ELEMENT, null, false);
```

2. Comprueba si el nodo es un frame, una imagen o texto, ya que, el procesamiento para cada tipo de nodo cambia:
    a. Si el nodo es un frame y no contiene la palabra:
        i. Oculta el frame entero llamando a la función 'webfiltering_ocultarnodos()' y pasándole como argumento el frame a ocultar.
    b. Si el nodo es un frame y contiene la palabra:
        i. Crea un treeWalker para recoger la información contenida dentro del frame.

ii. Para cada nodo comprueba si es una imagen o es texto. Si es una imagen comprueba si alguno de estos atributos: alt, name, longDesc y src aparece la palabra buscada. Si aparece guarda la imagen en el vector 'webfiltering_arraymuestra' y si no aparece llama a la función 'webfiltering_ocultarnodos()'.

iii. Si el nodo es texto y no es un iframe, busca en su contenido la palabra, si aparece guarda dicho nodo en el vector 'webfilteirng_arraymuestra' y si no aparece llama a la función 'webfiltering_ocultarnodos()'.

```
if(nodo.tagName=="FRAME"){
        cuerpoFrame = nodo.contentDocument.body.innerHTML;
        if(cuerpoFrame.toLowerCase().search(webfiltering_texto.toLowerCase()) ==
          -1){
              webfiltering_ocultarnodos(nodo);
        }
        else{
            webfiltering_arraymuestra[webfiltering_iglobal] = nodo;
            webfiltering_iglobal++;
            //crea un treeWalker con el contenido del frame y se va recorriendo
            arbolFrame=document.createTreeWalker
              (nodo.contentDocument.body, NodeFilter.SHOW_ELEMENT, null, false);

            while(arbolFrame.nextNode()){
                if(arbolFrame.currentNode.tagName == "IMG"){
                    if((arbolFrame.currentNode.alt.toLowerCase().
                        search(webfiltering_texto.toLowerCase()) == -1) &&
                       (arbolFrame.currentNode.name.toLowerCase().
                        search(webfiltering_texto.toLowerCase()) == -1)&&
                       (arbolFrame.currentNode.longDesc.toLowerCase().
                        search(webfiltering_texto.toLowerCase()) == -1) &&
                       (arbolFrame.currentNode.src.toLowerCase().
                        search(webfiltering_texto.toLowerCase()) == -1))
                                webfiltering_ocultarnodos(arbolFrame.currentNode);
                    else{
                                webfiltering_arraymuestra[webfiltering_iglobal] =
                                      arbolFrame.currentNode;
                                webfiltering_iglobal++;
                    }
                }//IMG
                else{
                    if(arbolFrame.currentNode.innerHTML.toLowerCase().
                       search(webfiltering_texto.toLowerCase()) == -1 &&
                       arbolFrame.currentNode.tagName != "IFRAME")
                                webfiltering_ocultarnodos(arbolFrame.currentNode);
                    else{
                                webfiltering_arraymuestra[webfiltering_iglobal] =
                                          arbolFrame.currentNode;
                                webfiltering_iglobal++;
                    }
                } //no IMG
            } //while del frame con la palabra
        } //frame con la palabra
} //if FRAME
```

c. Si el nodo es una imagen busca en los atributos: alt, name, longDesc y src si aparece la palabra buscada. Si sí lo hace, guarda el nodo en 'webfiltering_arraymuestra' y si no lo hace llama a la función 'webfiltering_ocultarnodos()'.

```
if(nodo.tagName=="IMG"){
```

```
        if( (nodo.alt.toLowerCase().search(webfiltering_texto.toLowerCase())==-1)  &&
           (nodo.name.toLowerCase().search(webfiltering_texto.toLowerCase())==-1)  &&
           (nodo.longDesc.toLowerCase().search(webfiltering_texto.toLowerCase())==-1) &&
           (nodo.src.toLowerCase().search(webfiltering_texto.toLowerCase())==-1)){
                webfiltering_ocultarnodos(nodo);
        }
        else{
                //sino lo guarda en el arraymuestra
                webfiltering_arraymuestra[webfiltering_iglobal] = nodo;
                webfiltering_iglobal++;
         }
    }//IMG
```

d. Si el nodo es texto y no es un iframe, entonces busca en su contenido la palabra y si aparece entonces lo guarda en el vector 'webfiltering_arraymuestra' y si no aparece llama a la función 'webfiltering_ocultarnodos()'.

```
        else{
                if(nodo.innerHTML.toLowerCase().search
                  (webfiltering_texto.toLowerCase())==-1  && nodo.tagName != "IFRAME"){
                        webfiltering_ocultarnodos(nodo);
                }
                else{
                        webfiltering_arraymuestra[webfiltering_iglobal] = nodo;
                        webfiltering_iglobal++;
                 }
           } //no IMG
```

3. Si el vector 'arraymuestra' está vacío quiere decir que no se ha guardado nada, y eso significa que la palabra buscada no se encuentra en la página, bien por estar mal escrita, bien por no encontrarse. En ambos casos se avisa al usuario de ello y se recarga la página Web original.

```
    if(webfiltering_arraymuestra.length == 0){
        alert("La palabra buscada no se encuentra en la pagina. Revise la ortografia");
        content.window.location.reload();
    }
```

4.1.10. FUNCIÓN *MOSTRARCONTOLERANCIA*

Esta función es la encargada de mostrar el resultado del filtrado con el nivel de tolerancia que el usuario haya indicado previamente.

4.1.10.1. EVENTO

Es invocada desde la función 'webfiltering_funcslice' y como se ha dicho anteriormente, muestra la página resultante del filtrado con la cantidad de información que se corresponda con el nivel de tolerancia que haya indicado el usuario.

4.1.10.2. FUNCIONAMIENTO

A continuación se adjunta su código fuente:

```
function webfiltering_mostrarcontolerancia(){
    //se guarda en la variable global tolerancia el contenido del cuadro de tolerancia
```

```
    webfiltering_tolerancia = document.getElementById("cuadrotolerancia").value;

    //recorre el arraymuestra
    for(var i=0;i<webfiltering_arraymuestra.length;i++){
        //si existe el nodo y es hoja entra en el if
        if (webfiltering_arraymuestra[i]!=null &&
           webfiltering_eshoja(webfiltering_arraymuestra[i])){
                var parametro;
                var anterior;
                var k;

                 //va cogiendo el nodo padre según el nivel de tolerancia
                parametro = webfiltering_arraymuestra[i];
                for(k=0; k<webfiltering_tolerancia;k++){
                        anterior = parametro
                        parametro = parametro.parentNode;
                        //si el nombre del nodo es nulo o igual al documento significa que no
                        existe y el bucle para
                        if(parametro.nodeName==null ||
                          parametro.nodeName == "#document"){
                                parametro = anterior;
                                break;
                        }
                }
                //llama al método mostrar con el nodo que tiene que mostrar
                webfiltering_mostrar(parametro);
        }
    }
}
```

1. Obtiene el nivel de tolerancia del recuadro  deseado por el usuario.

```
    //se guarda en la variable global tolerancia el contenido del cuadro de tolerancia
    webfiltering_tolerancia = document.getElementById("cuadrotolerancia").value;
```

2. Recorre el vector 'webfiltering_arraymuestra', en el cual se encuentran almacenados todos los elementos a mostrar. Si el nodo es una hoja, entonces se deben mostrar todos sus antecesores hasta el nivel de tolerancia indicado por el usuario.
3. Una vez se tiene el antecesor a mostrar, se llama a la función 'webfiltering_mostrar()' que recibe el nodo correspondiente.

```
    for(var i=0;i<webfiltering_arraymuestra.length;i++){
        //si existe el nodo y es hoja entra en el if
        if (webfiltering_arraymuestra[i]!=null &&
           webfiltering_eshoja(webfiltering_arraymuestra[i])){
                var parametro;
                var anterior;
                var k;

                 //va cogiendo el nodo padre según el nivel de tolerancia
                parametro = webfiltering_arraymuestra[i];
                for(k=0; k<webfiltering_tolerancia;k++){
                        anterior = parametro
                        parametro = parametro.parentNode;
                        //si el nombre del nodo es nulo o igual al documento significa que no
                        existe y el bucle para
                        if(parametro.nodeName==null ||
                          parametro.nodeName == "#document"){
                                parametro = anterior;
```

```
                                break;
                            }
                        }
                        //llama al método mostrar con el nodo que tiene que mostrar
                        webfiltering_mostrar(parametro);
                }
        }
```

4.1.11. FUNCIÓN *MOSTRAR*

Esta función se encarga de mostrar un nodo y todos sus hijos.

4.1.11.1. EVENTO

Se invoca desde la función 'webfiltering_mostrarcontolerancia()' y se encarga de hacer visible el nodo que recibe como parámetro y todos sus hijos hasta llegar a una hoja.

4.1.11.2. FUNCIONAMIENTO

A continuación se muestra su código:

```
//muestra el contenido del padre mostrando para ellos el contenido de todos sus hijos
function webfiltering_mostrar(n){
    n.style.visibility="visible";
    for (var j=0; j<n.childNodes.length; j++)
        if(n.childNodes[j].tagName!=null){
                webfiltering_mostrar(n.childNodes[j]);
    }
}
```

Recibe como parámetro el nodo a mostrar. Hace visible el nodo que recibe como parámetro y recursivamente también hace visibles todos sus hijos. Esta función terminará cuando se llegué a un nodo hoja, puesto que, estos no tienen hijos.

4.1.12. FUNCIÓN *ESHOJA*

Dado un nodo cualquiera, esta función se encarga de comprobar si es un nodo hoja o no lo es.

4.1.12.1. EVENTO

Se llama desde la función 'webfiltering_mostrarcontolerancia()', y se encarga de comprobar si el nodo que recibe como argumento es hoja o no lo es.

4.1.12.2. FUNCIONAMIENTO

A continuación se puede ver su código:

```
//devuelve true si el nodo es hoja o no, para saber si tiene hijos
function webfiltering_eshoja(N){
    if(N.tagName==null) return false; //si no tiene tagName es porque no existe
    var ii;
    //recorre todos los hijos del nodo
    for(ii=0;ii<N.childNodes.length;ii++)
        //si algún hijo tiene tagName es porque no es hoja
        if(N.childNodes[ii].tagName!=null)
```

```
                    //si algún hijo contiene la palabra buscada es que N no es hoja
                    if(N.childNodes[ii].innerHTML.toLowerCase().search
                            (webfiltering_texto.toLowerCase())!=-1)
                                return false;
    return true;
}
```

1. Comprueba que el nodo que recibe como parámetro existe.
2. Si el nodo existe va recorriendo todos sus hijos. Para cada hijo comprueba si tiene tagName y si contiene la palabra buscada, ya que, en ambos casos no es uno nodo hoja.
3. Si el nodo es hoja devuelve cierto, en caso contrario devuelve falso.

4.1.13. FUNCIÓN OCULTAR NODOS

Esta función se encarga de ocultar el nodo que recibe como parámetro.

4.1.13.1. EVENTO

Esta función se invoca desde prácticamente todas las funciones principales de la aplicación. Oculta el nodo que recibe como parámetro en función del valor de la preferencia que indica si el usuario decide mantener la estructura o no.

4.1.13.2. FUNCIONAMIENTO

El código de la función es el mostrado a continuación:

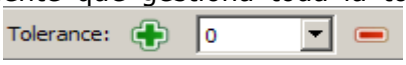
```
function webfiltering_ocultarnodos(n){
    // Los frames no respetan la propiedad "display".
    // Por tanto, siempre los ocultamos cambiando la propiedad "visibility"
    // independientemente de que se haya activado o no la opción "keep Structure"
    if (n.tagName!="FRAME"){
        //si esta activada la opcion mantener estructura esconde el nodo
        if(webfiltering_mantenerestructura==1) n.style.visibility="hidden";
            //sino lo quita
        else n.style.display="none";
    }
    else{
            n.style.visibility="hidden";
     }
     //mete el nodo en el array arrayocultos
     webfiltering_arrayocultos[webfiltering_jglobal] = n;
     webfiltering_jglobal++;
}
```

1. Si el nodo es un frame, independientemente de si se mantiene la estructura o no, siempre se modifica la propiedad 'visibility' a hidden, puesto que los frames no respetan la propiedad display.
2. Para el resto de nodos, si el usuario decide mantener la estructura, entonces la propiedad 'visibility' del nodo se pone a 'hidden'. Esta propiedad es la responsable de que se visualice ese elemento o no se visualice, poniéndola a 'hidden' indicamos que queremos ocultarlo para que no se muestre.
3. Por el contrario, si el usuario decide no mantener la estructura de la página, la propiedad 'display', indica si el nodo va a estar presente en la página o no, se pone a 'none', es decir, se elimina ese nodo de la página Web.
4. Independientemente del valor de la preferencia mantener estructura, el nodo a ocultar se guarda dentro del vector creado para ello 'webfiltering_arrayocultos'.

4.1.14. FUNCIONES DEL MANEJO DE LA TOLERANCIA

Las funciones que se explicarán a continuación son las encargadas de actualizar la tolerancia dentro del cuadro de texto, así como de guardarla para, posteriormente recuperarla con la función 'mostrarcontolerancia'.

4.1.14.1. EVENTO

La componente que gestiona toda la tolerancia se presenta de este modo en la aplicación: 

La función 'webfiltering_aumentartolerancia()' se invoca desde el botón  y aumenta en uno el valor que aparece en el cuadro de texto de la tolerancia.

La función 'webfiltering_disminuirtolerancia()' se invoca desde el botón  y disminuye en uno el valor que aparece en el cuadro de texto de la tolerancia.

La función 'webfiltering_actualizartolerancia()' se llama desde el cuadro de texto 

de la tolerancia cada vez que el usuario lo despliega y selecciona un valor.

4.1.14.2. FUNCIONAMIENTO

El código de la función 'webfiltering_aumentartolerancia' se muestra a continuación. El de la función 'webfiltering_disminuirtolerancia' no se muestra por ser su funcionamiento idéntico al de aumentar tolerancia, salvo que diminuye el valor de la tolerancia; su código puede verse en el anexo A.

```
function webfiltering_aumentartolerancia(){
    webfiltering_tolerancia++;
    webfiltering_setIntegerPreference("slice.tolerance",webfiltering_tolerancia);
    document.getElementById("cuadrotolerancia").value = webfiltering_tolerancia;
    webfiltering_funcslice();
}
```

1. Incrementa en uno la variable global encargada de controlar el valor de la tolerancia.
2. Modifica el valor de la variable global con el método setIntegerPreference del software webdeveloper.
3. Actualiza el valor del cuadro de texto con la nueva tolerancia.
4. Llama a la función 'webfiltering_funcslice' para volver a realizar el filtrado con el nuevo valor de la tolerancia. En este caso, como resultado del filtrado se visualizará más información que anteriormente.

El código de la función 'webfiltering_actualizartolerancia()' puede verse a continuación:

```
function webfiltering_actualizartolerancia(number){
        webfiltering_tolerancia = number;
        webfiltering_setIntegerPreference("slice.tolerance",webfiltering_tolerancia);
}
```

Recibe como parámetro el valor seleccionado por el usuario en el desplegable y se encarga de modificar la variable global de la tolerancia con el nuevo valor.

4.1.15. FUNCIÓN *VERPAGENTERA*

Esta función se encarga de recargar la página Web original cada vez que el usuario lo desee, perdiendo toda acción realizada sobre la misma, ya sea un filtrado o un resaltado.

4.1.15.1. EVENTO

Esta función se invoca desde el botón 

y desde otras funciones de la aplicación, encargándose de inicializar algunas variables globales y recargando la página Web original.

4.1.15.2. FUNCIONAMIENTO

El código de la misma se adjunta a continuación:

```
function webfiltering_verpagentera(){
        //inicializamos algunas variables globales
        webfiltering_sliceAplicado=0;
        webfiltering_paginaOriginal=0;
        webfiltering_URLActual=0;
        webfiltering_iglobal=0;
        webfiltering_jglobal=0;
        webfiltering_numPalabrasResaltadas=0;
        webfiltering_contador=0;
        webfiltering_contadorSig=0;
        webfiltering_contadorAnt=0;
        webfiltering_AplicadoAnterior=0;
        webfiltering_AplicadoSiguiente=0;
        webfiltering_AplicadoResaltar=0;
        webfiltering_primerapasadaframes=0;
        webfiltering_primerapasadaiframes=0;

        content.window.location.reload(); // Recargamos la página
}
```

1. La variable que indica si se ha aplicado algún filtrado la pone a cero, puesto que al recargarse la página todo lo hecho anteriormente se pierde.

```
        webfiltering_sliceAplicado=0;
```

2. Las variables que almacenan el contenido de la página original y su URL, respectivamente, las pone a cero para que no guardar ninguna acción anterior realizada sobre la página.

```
        webfiltering_paginaOriginal=0;
        webfiltering_URLActual=0;
```

3. Pone a cero el número de elementos mostrados y ocultados, para no tener ninguna información anterior.

```
        webfiltering_iglobal=0;
        webfiltering_jglobal=0;
```

4. Inicializa a cero la cuenta del número de palabras resaltadas, ya que se vuelve a partir de cero cuando se recarga la página.

```
webfiltering_numPalabrasResaltadas=0;
```

5. También inicializa todas las variables que controlan los eventos anterior, siguiente y resaltar.

```
webfiltering_contador=0;
webfiltering_contadorSig=0;
webfiltering_contadorAnt=0;
webfiltering_AplicadoAnterior=0;
webfiltering_AplicadoSiguiente=0;
webfiltering_AplicadoResaltar=0;
```

6. Por último, pone a cero las dos variables que controlan si es la primera vez que se van a procesar tanto los iframes como los frames, puesto que, al recargarse deben procesarse exactamente igual a como lo hicieron la primera vez.

```
webfiltering_primerapasadaframes=0;
webfiltering_primerapasadaiframes=0;
```

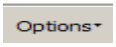
7. Se recarga la página desde el servidor Web donde está ubicada, haciendo uso de la siguiente instrucción.

```
content.window.location.reload(); // Recargamos la página
```

4.1.16. FUNCIÓN MOSTRAR MENÚ OPCIONES *(SLICE_OPTIONS)*

Esta es la función encargada de abrir el menú donde el usuario puede seleccionar sus preferencias de filtrado, mostrar la ayuda o ver la información acerca de la versión de la aplicación y de los autores de la misma.

4.1.16.1. EVENTO

Esta función se invoca cuando se pulsa sobre el botón 

y se encarga de mostrar un menú con tres opciones: opciones, ayuda o acerca de. El aspecto que tiene el menú que se muestra es el siguiente:

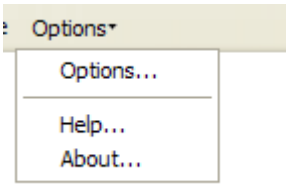


Figura 32. Menú que se muestra cuando se pulsa sobre el botón 'Options'

4.1.16.2. FUNCIONAMIENTO

El código fuente de la misma puede verse a continuación:

```
function webfiltering_slice_options(){
    window.openDialog("chrome://SliceToolbar/content/options/options.xul", "slice-options-
    dialog", "centerscreen,chrome,modal,resizable");

    //se actualiza la variable global tolerancia y el cuadro de tolerancia
    webfiltering_tolerancia =webfiltering_getIntegerPreference("slice.tolerance", true);

   if (document.getElementById("cuadrotolerancia") != null) {
        document.getElementById("cuadrotolerancia").value = webfiltering_tolerancia;
        document.getElementById("cuadrotol").value = webfiltering_tolerancia;
    }

    if((webfiltering_getIntegerPreference("slice.keeptree", true)==0) &&
       webfiltering_getIntegerPreference("slice.mantenerestructura", true)==0){
                document.getElementById("botonmas").disabled = true
                document.getElementById("botonmenos").disabled = true
                document.getElementById("cuadrotolerancia").disabled = true
     }
     else{
                document.getElementById("botonmas").disabled = false
                document.getElementById("botonmenos").disabled = false
                document.getElementById("cuadrotolerancia").disabled = false

     }
}
```

1. Para mostrar el menú llama al fichero xul llamado 'options.xul', que es el que contiene la interfaz de cómo va a ser ese menú. El fichero xul puede verse en el anexo B.

```
    window.openDialog("chrome://SliceToolbar/content/options/options.xul", "slice-options-
    dialog", "centerscreen,chrome,modal,resizable");
```

2. Actualiza la variable global y el cuadro de la tolerancia. Teniendo en cuenta que si el usuario ha seleccionado anteriormente no mantener el árbol y no mantener la estructura el cuadro de la tolerancia estará deshabilitado, ya que, al eliminarse los nodos estos no se pueden recuperar, por lo que no tiene sentido aumentar o disminuir el nivel de tolerancia.
   De igual manera, si el usuario decide mantener el árbol y mantener la estructura y, anteriormente el cuadro estaba deshabilitado, esta función se encarga de volver a habilitarlo para permitir su modificación.

```
    //se actualiza la variable global tolerancia y el cuadro de tolerancia
    webfiltering_tolerancia =webfiltering_getIntegerPreference("slice.tolerance", true);

    if (document.getElementById("cuadrotolerancia") != null) {
        document.getElementById("cuadrotolerancia").value = webfiltering_tolerancia;
        document.getElementById("cuadrotol").value = webfiltering_tolerancia;
    }

    if((webfiltering_getIntegerPreference("slice.keeptree", true)==0) &&
       webfiltering_getIntegerPreference("slice.mantenerestructura", true)==0){
                document.getElementById("botonmas").disabled = true
                document.getElementById("botonmenos").disabled = true
                document.getElementById("cuadrotolerancia").disabled = true
     }
     else{
                document.getElementById("botonmas").disabled = false
                document.getElementById("botonmenos").disabled = false
```

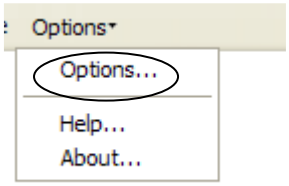
```
                document.getElementById("cuadrotolerancia").disabled = false

    }
```

4.1.17. FUNCIÓN MOSTRAR EL MENÚ DE PREFERENCIAS *(SEEOPTIONS)*

Esta función se encarga de mostrar el menú con las diferentes preferencias que puede seleccionar el usuario.

4.1.17.1. EVENTO

Se llama cuando el usuario selecciona la opción 'Options' del menú de opciones.



4.1.17.2. FUNCIONAMIENTO

A continuación se muestra el código de la función:

```
function webfiltering_seeOptions(){

    webfiltering_mantenerestructura =
                        webfiltering_getIntegerPreference("slice.mantenerestructura",true);
    if(webfiltering_mantenerestructura==1)
        document.getElementById("mantenerestructuraopciones").checked = true;
    else document.getElementById("mantenerestructuraopciones").checked = false;

    webfiltering_formateartamanyoiframes =
                                webfiltering_getIntegerPreference("slice.incrustacion",true);
    if(webfiltering_formateartamanyoiframes==1)
        document.getElementById("formateartamanyoiframesopciones").checked = true;
    else document.getElementById("formateartamanyoiframesopciones").checked = false;

    webfiltering_mantenerarbol = webfiltering_getIntegerPreference("slice.keeptree",true);
    if(webfiltering_mantenerarbol==1)
        document.getElementById("mantenerarbol").checked = true;
    else document.getElementById("mantenerarbol").checked = false;

    webfiltering_tolerancia = webfiltering_getIntegerPreference("slice.tolerance",true);
    document.getElementById("tol").value = webfiltering_tolerancia;
}
```

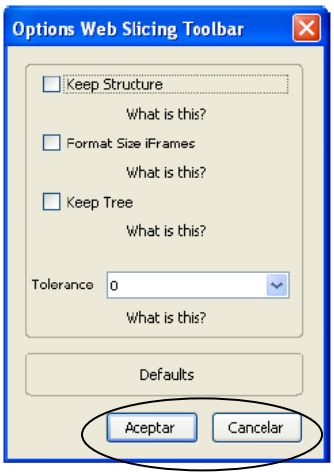
Actualiza los check (seleccionables) del menú de preferencias de cada variable, por los valores actuales que tienen cada una de ellas.

4.1.18. FUNCIÓN GUARDAR OPCIONES *(SAVEOPTIONS)*

Esta función se encarga de guardar los valores actuales de las preferencias.

4.1.18.1. EVENTO

Esta función se llama cuando se pulsa el botón 'Aceptar' o 'Cancelar' del diálogo de las preferencias.



4.1.18.2. FUNCIONAMIENTO

El código de la función es el siguiente:

```
function webfiltering_saveOptions(){
    webfiltering_tolerancia = document.getElementById("tol").value;
    webfiltering_setIntegerPreference("slice.tolerance", webfiltering_tolerancia);

    if (document.getElementById("cuadrotolerancia") != null) {
        document.getElementById("cuadrotolerancia").value = webfiltering_tolerancia;
        document.getElementById("cuadrotol").value = webfiltering_tolerancia;
    }

    if(document.getElementById("mantenerestructuraopciones").checked == true)
        webfiltering_setIntegerPreference("slice.mantenerestructura", 1)
    else webfiltering_setIntegerPreference("slice.mantenerestructura", 0)

    if(document.getElementById("formateartamanyoiframesopciones").checked == true)
        webfiltering_setIntegerPreference("slice.incrustacion", 1)
    else webfiltering_setIntegerPreference("slice.incrustacion", 0)

    if(document.getElementById("mantenerarbol").checked == true)
        webfiltering_setIntegerPreference("slice.keeptree", 1)
    else webfiltering_setIntegerPreference("slice.keeptree", 0)
}
```

1. Primero guarda las opciones del diálogo, de modo que cuando el usuario vuelva a abrirlo, las opciones estarán como las dejó.
2. Segundo actualiza el valor de todas las variables globales que utiliza la aplicación para controlar las preferencias, con las nuevas opciones seleccionadas por el usuario.

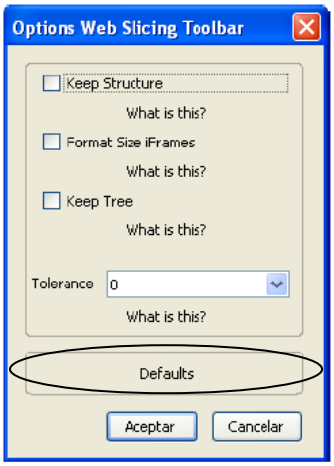
4.1.19. FUNCIÓN VALORES POR *DEFECTO (LOAD_DEFAULT)*

La aplicación tiene unos valores por defecto, que son con los que se instala la primera vez, y el usuario puede recuperarlos siempre que lo desee. Esta función se

encarga precisamente de eso, de recuperar los valores por defecto con los que viene la aplicación.

4.1.19.1. EVENTO

Cuando el usuario pulsa el botón 'Defaults' del menú con las preferencias, se llama a la función 'webfiltering_load_defaults()', y se encarga de recuperar y actualizar las variables de opciones con los valores por defecto.



4.1.19.2. FUNCIONAMIENTO

El código fuente de la misma puede verse a continuación:

```
function webfiltering_load_defaults(){
   // Inicializamos valores por defecto en los componentes gráficos de las opciones
   document.getElementById("mantenerestructuraopciones").checked = true;
   document.getElementById("formateartamanyoiframesopciones").checked = false;
   document.getElementById("mantenerarbol").checked = true;
   document.getElementById("tol").value = 0;
   document.getElementById("cuadrotolerancia").value=0;
}
```

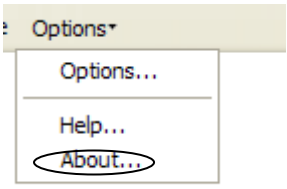
Actualiza el componente gráfico de las opciones (los check) con los valores por defecto de la aplicación. Estos valores son: mantener la estructura, no mantener el tamaño de los iframes, mantener el árbol de nodos y nivel de tolerancia cero.

4.1.20. FUNCIÓN ACERCA DE *(ABOUT)*

Esta función se encarga de mostrar un cuadro de diálogo con información de la aplicación: nombre y versión, fecha de finalización y los autores.

4.1.20.1. EVENTO

Cuando el usuario selecciona la opción 'About' del menú de opciones, se llama a la función 'webfiltering_about()' que llama a un fichero xul que será el encargado de mostrar correctamente toda la información de la aplicación.



4.1.20.2. FUNCIONAMIENTO

El código de la misma se muestra a continuación. El código del fichero xul puede consultarse en el anexo B.

```
function webfiltering_about(){
  window.openDialog("chrome://SliceToolbar/content/about/about.xul", "slice-about-dialog",
        "centerscreen,chrome,modal,resizable");
}
```

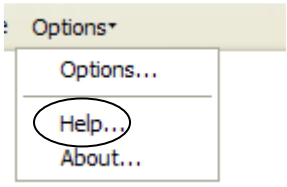
Únicamente se encarga de abrir el fichero 'about.xul', que contiene la información de la aplicación y la interfaz de la misma, es decir, como debe mostrarse esa información por pantalla.

4.1.21. FUNCIÓN AYUDA *(HELP)*

Esta función muestra la ayuda de cómo utilizar la aplicación. Este manual de ayuda se proporciona con la aplicación y consta de ejemplos de cómo queda el filtrado con cada una de las preferencias.

4.1.21.1. EVENTO

Cuando el usuario selecciona la opción 'Help' del menú de opciones, se llama a la función 'webfiltering_help()' que carga una página Web donde se muestra un manual de ayuda con ejemplos de utilización.



4.1.21.2. FUNCIONAMIENTO

A continuación se muestra el código de la función.

```
function webfiltering_help(theURL,winName,features){
        window.open(theURL,winName, features);
}
```

Recibe como parámetros la URL de la página Web donde se mostrará la ayuda, el nombre de la misma y el manual que debe mostrar.

4.1.22. FUNCIONES *OCULTARBARRA* Y *MOSTRARBARRA*

Estas funciones tienen como objetivo mostrar y ocultar la barra siempre que el usuario lo desee. Inicialmente la barra siempre está visible, pero si el usuario lo desea puede ocultarla presionando sobre la equis. Para volver a mostrarla, se debe pulsar la combinación de teclas 'Ctrl+Y'.

4.1.22.1. EVENTO

Cuando el usuario pulsa sobre el icono de la equis  la aplicación llama a la función 'webfiltering_ocultarbarra()' que se encarga de ocultar al usuario la barra de herramientas.
Al pulsar la combinación de teclas que muestra la barra, se produce un evento que se controla llamando a la función 'webfiltering_mostrarbarra()'.

4.1.22.2. FUNCIONAMIENTO

A continuación se puede ver el código de ambas funciones:

```
function webfiltering_mostrarbarra(){
        var mostrar=true;
        if (document.getElementById("barra").hidden == mostrar){
                mostrar=false;
                document.getElementById("barra").hidden=mostrar;
        }
}
```

```
function webfiltering_ocultarbarra(){
        var ocultar=true;
        document.getElementById("barra").hidden=ocultar;
}
```

En xul, muchos elementos poseen una propiedad (llamada hidden) que indica si dicho elemento estará visible siempre, a veces o nunca. Esta propiedad debe llevar un valor por defecto siempre. Cuando definimos la barra de herramientas, con el elemento 'toolbar' que posee xul, el valor por defecto que hemos decidido ponerle es que se muestre siempre. Mediante estas funciones manipulamos el valor de esa propiedad para que pase de visible a no visible, según corresponda.

4.1.23. FUNCIONES GUARDAR PÁGINAS ORIGINALES

Las cuatro funciones que se explican en este punto, son las encargadas de guardar la página Web original, la URL de la misma y la información original de los iframes y de los frames. Estas funciones se crearon para poder recuperar en todo momento la información original (sin filtrados ni resaltados) y trabajar siempre sobre la misma.

4.1.23.1. EVENTO

Estas funciones son utilizas por todas las funciones principales de la aplicación (resaltar, filtrar…), por lo que, son invocadas desde distintos puntos del código. Como ya se ha comentado, se encargan de guardar el contenido de la página que reciben como argumento de entrada.

4.1.23.2. FUNCIONAMIENTO

A continuación solamente se muestra el código de la función 'webfiltering_guardarPag()' y 'webfiltering_guardarURL()'. Las funciones 'webfiltering_guardarFrameOrig()' y 'webfiltering_guardarIFrameOrig()' pueden consultarse en el anexo A.

```
//función encargada de almacenar la página original, sin aplicar ninguna acción
function webfiltering_guardarPag(pagina){
        return pagina.body.innerHTML;
}
```

```
//función encargada de almacenar la URL original, para detectar cuando se cambia de página
function webfiltering_guardarURL(pagina){
        return pagina.URL;
}
```

La primera función recibe una página y devuelve el contenido de la misma con la propiedad body.innerHTML.
La segunda función también recibe una página y devuelve, en este caso la URL de la misma con la propiedad URL de la página.

4.1.24. FUNCIÓN NO MANTENER EL ÁRBOL *(NOKEEPTREE)*

Esta función se encarga de eliminar los nodos que no son relevantes para el usuario, por no contener la palabra buscada. Si el usuario decide no mantener el árbol, todos los nodos que se encuentren en el camino desde la raíz hasta el nodo relevante, serán eliminados, haciendo que el nodo relevante sea hijo directo de la raíz.

4.1.24.1. EVENTO

Esta función se llama desde la función 'webfiltering_funcslice' cuando la preferencia de no mantener el árbol no está seleccionada, es decir, el usuario no quiere mantenerlo.

4.1.24.2. FUNCIONAMIENTO

A continuación se puede ver el código fuente de la misma.

```
function webfiltering_nokeeptree(){
    var n; // Un elemento cualquiera de la página
    var pare,b;
    var cuerpoFrame;
    var nodo; // Un elemento de un frame
    var arbol=new Array();
    var x=new Array(); // Vector de elementos de la página (están todos)
    var vectorDeFrames=new Array(); //Vector de frames

    x=content.document.getElementsByTagName("*");
    vectorDeFrames=content.document.getElementsByTagName("FRAME");

    if(vectorDeFrames.length > 0){ //página con frames
        for(var w=0; w < vectorDeFrames.length; w++){
                webfiltering_nokeeptreeframes(vectorDeFrames[w]);
        }
        if(webfiltering_numFramesSinPalabra == vectorDeFrames.length){
```

```
                alert("La palabra introducida no se encuentra en la pagina. Revise la
                        ortografia");
                webfiltering_verpagentera();
            }
        }
        else{ //páginas sin frames
            // Iniciamos el proceso de ocultacion de elementos antecesores
            if (webfiltering_mantenerestructura==1){
                for(var j=0; j < x.length; j++){
                    n=x[j];
                    if(n.tagName=="IMG"){
                        if((n.alt.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1)  &&
                          (n.name.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1) &&
                          (n.longDesc.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1) &&
                          (n.src.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1)){
                                webfiltering_ocultarnodos(n);
                          }
                          else{
                                n.style.visibility="visible";
                                webfiltering_arraymuestra[webfiltering_iglobal] = n;
                                webfiltering_iglobal++;
                                webfiltering_hideparents(n);
                          }
                    }
                    else{
                        if(n.innerHTML.toLowerCase().search
                                (webfiltering_texto.toLowerCase())!=-1 ||
                           n.tagName == "IFRAME"){
                            if(n.tagName != "HTML" && n.tagName != "BODY"){
                                n.style.visibility="visible";
                                webfiltering_hideparents(n);
                            }
                        }
                        else{
                            if(n.tagName == "HTML" || n.tagName == "BODY"){
                                alert("La palabra introducida no se encuentra en la pagina.
                                        Revise la ortografia");
                                webfiltering_verpagentera();
                                return;
                            }
                            else{
                                webfiltering_ocultarnodos(n);
                            }
                        }
                    }
                }
            } //mantengo la estructura
            else{
                var cambios=true;
                while(cambios){
                    cambios = false;
                    for(var j=0; j < x.length; j++){
                        n=x[j];
                        if(n.tagName == "IMG"){
                            if((n.alt.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1)  &&
                              (n.name.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1) &&
                              (n.longDesc.toLowerCase().search
```

```
                                        (webfiltering_texto.toLowerCase())==-1) &&
                            (n.src.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1)){
                                    cambios = true;
                                    pare = n.parentNode;
                                    pare.removeChild(n);
                            } //IMG sin la palabra
                            else{
                                    pare = n.parentNode;
                                    pare.removeChild(n);
                                    //meterlo en el body
                                    b = content.document.getElementsByTagName
                                                            ("body")[0];
                                    b.appendChild(n);
                             } //IMG con la palabra
                        } //IMG
                        else{
                            if(n.innerHTML.toLowerCase().search
                                    (webfiltering_texto.toLowerCase())!=-1 ||
                              n.tagName == "IFRAME"){
                                if(n.tagName != "HTML" && n.tagName != "BODY"){
                                    pare = n.parentNode;
                                    pare.removeChild(n);
                                    b =content.document.getElementsByTagName
                                                                ("body")[0];
                                    b.appendChild(n);
                                }
                            }
                            else {
                                if(n.tagName == "HTML" || n.tagName == "BODY"){
                                    alert("La palabra introducida no se encuentra en la
                                            pagina. Revise la ortografia.");
                                    webfiltering_verpagentera();
                                    return;
                                }
                                else{
                                    cambios = true;
                                    pare = n.parentNode;
                                    pare.removeChild(n);
                                }
                            }
                        } //no IMG
                    } //for
                } //while
            } //No mantengo la estructura
        } //páginas sin frames
}
```

1. Recupera todos los elementos de la página Web y obtiene todos los posibles frames que pueda tener la página. Si la página tiene frame debe tener un tratamiento diferente.

```
x=content.document.getElementsByTagName("*");
vectorDeFrames=content.document.getElementsByTagName("FRAME");
```

2. Si la página tiene frames se llama a la función 'webfiltering_nokeeptreeframes()', una vez por cada frame.
   Cuando se han ocultado todos los frames, significa que la palabra introducida no aparece en ninguno de los frames. En este caso se avisa al usuario de ello y se recarga la página original.

```
if(vectorDeFrames.length > 0){ //página con frames
      for(var w=0; w < vectorDeFrames.length; w++){
            webfiltering_nokeeptreeframes(vectorDeFrames[w]);
      }
      if(webfiltering_numFramesSinPalabra == vectorDeFrames.length){
            alert("La palabra introducida no se encuentra en la pagina. Revise la
                  ortografia");
            webfiltering_verpagentera();
      }
}
```

3. Si se mantiene la estructura:
    a. Va recorriendo uno a uno todos los nodos de la página y mirando de que tipo son.
        i. Si el nodo es una imagen mira en sus atributos alt, name, longDesc y src si aparece la palabra buscada. Si sí que aparece entonces se pone la propiedad visible a sí y se llama a la función 'webfiltering_hideparents()' para que oculte a todos los antecesores de ese nodo exceptuando la raíz. Guarda la imagen en el vector webfiltering_arraymuestra.
        Si no aparece se llama a la función 'webfiltering_ocultarnodos' para que oculte esa imagen.
        ii. Si el nodo es texto se comprueba en su contenido si contiene la palabra.Si no la contiene y es el nodo raíz se muestra un mensaje al usuario avisando de ello y se llama a la función webfiltering_verpagentera para que recarge la página original. Si sí la contiene y no es el nodo raíz (etiquetado como HTML o BODY) entonces se pone la propiedad de visibilidad a sí y se llama a la función 'webfiltering_hideparents()'.
        Si no contiene la palabra y no es el nodo raíz se llama a la función 'webfiltering_ocultarnodos' para que oculte ese nodo.

```
if (webfiltering_mantenerestructura==1){
// Iniciamos el proceso de ocultación de elementos antecesores
      for(var j=0; j < x.length; j++){
          n=x[j];
           if(n.tagName=="IMG"){
               if((n.alt.toLowerCase().search(webfiltering_texto.toLowerCase())==-1) &&
                  (n.name.toLowerCase().search(webfiltering_texto.toLowerCase())==-1) &&
                  (n.longDesc.toLowerCase().search(webfiltering_texto.toLowerCase())==-1)
                   &&(n.src.toLowerCase().search(webfiltering_texto.toLowerCase())==-1)){
                     webfiltering_ocultarnodos(n);
               }
               else{
                     n.style.visibility="visible";
                     webfiltering_arraymuestra[webfiltering_iglobal] = n;
                     webfiltering_iglobal++;
                     webfiltering_hideparents(n);
               }
           }
           else{
               if(n.innerHTML.toLowerCase().search(webfiltering_texto.toLowerCase())!=-1
                 || n.tagName == "IFRAME"){
                     if(n.tagName != "HTML" && n.tagName != "BODY"){
                           n.style.visibility="visible";
                           webfiltering_hideparents(n);
                     }
               }
               else{
                     if(n.tagName == "HTML" || n.tagName == "BODY"){
```

```
                        alert("La palabra introducida no se encuentra en la pagina.
                                Revise la ortografia");
                        webfiltering_verpagentera();
                        return;
                    }
                    else{
                        webfiltering_ocultarnodos(n);
                    }
                }
            }
        }
    }
```

4. Si no se mantiene la estructura:
    a. Mientras haya cambios, va recorriendo uno a uno todos los nodos del árbol.
        i. Si el nodo es una imagen mira en los atributos nombrados anteriormente si aparece la palabra. En este caso, si sí que aparece la palabra lo que hace es obtener el padre del nodo imagen, eliminar el enlace entre ellos y hacer que el nodo imagen tenga como padre directo la raíz del árbol.
        Si no contiene la palabra se indica que quedan cambios pendientes y se elimina el nodo imagen del árbol, obteniendo su padre y eliminando el enlace entre ambos.
        ii. Si el nodo es texto, se mira en su contenido si aparece la palabra. Si sí que aparece se comprueba que no sea el nodo raíz, etiquetado como HTML o BODY; si no lo es entonces se obtiene el padre del nodo texto, se elimina el enlace entre ellos y se hace que el nodo raíz tenga como hijo directo el nodo texto con la palabra.
        Si no contiene la palabra se pone la variable cambios a verdadera (para continuar con la ocultación) y se elimina ese nodo, que no contiene la palabra, del árbol rompiendo el enlace que lo une a su padre.
        En el caso en que el nodo raíz no contenga la palabra, significa que no la palabra no aparece en la página, se muestra un mensaje avisando al usuario de ello y se llama a la función webfiltering_verpagentera para recargar la página original.

```
else{
    var cambios=true;
    while(cambios){
        cambios = false;
        for(var j=0; j < x.length; j++){
            n=x[j];
            if(n.tagName == "IMG"){
                if((n.alt.toLowerCase().search
                                (webfiltering_texto.toLowerCase())==-1)  &&
                  (n.name.toLowerCase().search
                                (webfiltering_texto.toLowerCase())==-1) &&
                  (n.longDesc.toLowerCase().search
                                (webfiltering_texto.toLowerCase())==-1) &&
                  (n.src.toLowerCase().search
                                (webfiltering_texto.toLowerCase())==-1)){
                            cambios = true;
                            pare = n.parentNode;
                            pare.removeChild(n);
                } //IMG sin la palabra
                else{
```

```
                    pare = n.parentNode;
                    pare.removeChild(n);
                    //meterlo en el body
                    b = content.document.getElementsByTagName ("body")[0];
                    b.appendChild(n);
                  } //IMG con la palabra
                } //IMG
                else{
                    if(n.innerHTML.toLowerCase().search
                                (webfiltering_texto.toLowerCase())!=-1 ||
                       n.tagName == "IFRAME"){
                        if(n.tagName != "HTML" && n.tagName != "BODY"){
                                pare = n.parentNode;
                                pare.removeChild(n);
                                b =content.document.getElementsByTagName
                                                        ("body")[0];
                                b.appendChild(n);
                        }
                    }
                    else {
                        if(n.tagName == "HTML" || n.tagName == "BODY"){
                            alert("La palabra introducida no se encuentra en la
                                        pagina. Revise la ortografia.");
                            webfiltering_verpagentera();
                            return;
                        }
                        else{
                            cambios = true;
                            pare = n.parentNode;
                            pare.removeChild(n);
                        }
                    }
                } //no IMG
              } //for
          } //while
      } //No mantengo la estructura
```

4.1.25. FUNCIÓN *NOKEEPTREEFRAMES*

Esta función se encarga de procesar el contenido de un frame cuando la opción de no mantener el árbol está activada.

4.1.25.1. EVENTO

Se llama desde la función webfiltering_nokeeptree cuando el nodo encontrado es un frame y está seleccionada la opción de no mantener el árbol.

4.1.25.2. FUNCIONAMIENTO

A continuación puede verse el código de la misma. No se va a explicar puesto que se sigue el mismo procedimiento que se sigue con el contenido de páginas sin frames. La única diferencia, radica en la manera de comprobar que la palabra introducida no se encuentra en ninguno de los frames. Para ello, se utiliza una variable global, que contabiliza el número de frames que se ocultan por no contener la palabra. En la función 'webfiltering_nokeeptree()' si el número de frames ocultado coincide con el total de frames contenidos en la páginas, significa que no aparece la palabra en la página.

```
//función que se llama cuando la pagina no tiene frames y no keep tree
function webfiltering_nokeeptreeframes(nodo){
    var cuerpoFrame;
    var arbol=new Array();
    var padre, bodyAux, aux;
    var cambiosFrame=true;

    cuerpoFrame = nodo.contentDocument.body.innerHTML;
    if (webfiltering_mantenerestructura==1){
        if(cuerpoFrame.toLowerCase().search(webfiltering_texto.toLowerCase())==-1){
            //si el frame no contiene la palabra lo ocultamos entero
            webfiltering_ocultarnodos(nodo);
            webfiltering_numFramesSinPalabra++;
        }
        else{ //el frame contiene la palabra
            nodo.style.visibility="visible";
            webfiltering_arraymuestra[webfiltering_iglobal] = nodo;
            webfiltering_iglobal++;
            arbol=nodo.contentDocument.getElementsByTagName("*");

            for(var k=0; k < arbol.length; k++){
                aux=arbol[k];
                if(aux.tagName == "IMG"){
                    if((aux.alt.toLowerCase().search
                                        (webfiltering_texto.toLowerCase()) == -1) &&
                      (aux.name.toLowerCase().search
                                        (webfiltering_texto.toLowerCase()) == -1)&&
                      (aux.longDesc.toLowerCase().search
                                        (webfiltering_texto.toLowerCase()) == -1) &&
                      (aux.src.toLowerCase().search
                                        (webfiltering_texto.toLowerCase()) == -1)){
                                webfiltering_ocultarnodos(aux);
                    }
                    else{
                        aux.style.visibility="visible";
                         webfiltering_arraymuestra[webfiltering_iglobal] = aux;
                        webfiltering_iglobal++;
                        webfiltering_hideparents(aux);
                    }
                } //IMG
                else{
                     if(aux.innerHTML.toLowerCase().search
                                            (webfiltering_texto.toLowerCase())!=-1 ||
                       aux.tagName == "IFRAME"){
                            if(aux.tagName != "HTML" && aux.tagName != "BODY"){
                                    aux.style.visibility="visible";
                                    webfiltering_hideparents(aux);
                            }
                     }
                     else{
                        webfiltering_ocultarnodos(aux);
                     }
                } //no IMG
              } //for frame con palabra
        } //frame con la palabra
    } //mantengo la estructura
    else{
        if(cuerpoFrame.toLowerCase().search(webfiltering_texto.toLowerCase()) == -1){
                webfiltering_ocultarnodos(nodo);
                webfiltering_numFramesSinPalabra++;
        }
        else{
```

```
                arbol=nodo.contentDocument.getElementsByTagName("*");
                //vector con todos los elementos del frame
                while(cambiosFrame){
                    cambiosFrame = false;
                    for(var l=0; l < arbol.length; l++){
                        aux=arbol[l];
                        if(aux.tagName == "IMG"){
                            if((aux.alt.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1)  &&
                              (aux.name.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1) &&
                              (aux.longDesc.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1) &&
                              (aux.src.toLowerCase().search
                                        (webfiltering_texto.toLowerCase())==-1)){
                                    cambiosFrame = true;
                                    padre = aux.parentNode;
                                    padre.removeChild(aux);
                             } //IMG sin la palabra
                            else{
                                    padre = aux.parentNode;
                                    padre.removeChild(aux);
                                    //meterlo en el body
                                    bodyAux = nodo.contentDocument.
                                                getElementsByTagName("body")[0];
                                    bodyAux.appendChild(aux);
                            } //IMG con la palabra
                        } //IMG
                        else{
                            if(aux.innerHTML.toLowerCase().
                                    search(webfiltering_texto.toLowerCase())!=-1 ||
                               aux.tagName == "IFRAME"){
                                 if(aux.tagName != "HTML" && aux.tagName != "BODY"){
                                    padre = aux.parentNode;
                                    padre.removeChild(aux);
                                    bodyAux = nodo.contentDocument.
                                                getElementsByTagName("body")[0];
                                    bodyAux.appendChild(aux);
                                 }
                            }
                            else {
                                cambiosFrame = true;
                                padre = aux.parentNode;
                                padre.removeChild(aux);
                            }
                        } //no IMG
                    } //for
                } //while
            } //frame con la palabra
        } //no mantengo la estructura
}
```

4.1.26. FUNCIÓN OCULTAR PADRES (HIDEPARENTS)

Esta función se encarga de ocultar los antecesores de los nodos que son relevantes para el usuario no por contener la palabra que busca. Se utiliza cuando no se debe mantener el árbol de nodos.

4.1.26.1. EVENTO

Se llama desde la función 'webfiltering_nokeeptree()' y sirve para ocultar los padres de los nodos que contienen la palabra buscada.

4.1.26.2. FUNCIONAMIENTO

El código de la misma se muestra a continuación.

```
//Dado un nodo del árbol, oculta todos los nodos antecesores hasta llegar a body
function webfiltering_hideparents(n){
    if(n.tagName != "HTML" && n.tagName != "BODY"){
        pare = n.parentNode;
        if(pare.tagName != "HTML" && pare.tagName != "BODY"){
                pare.style.visibility="hidden";
                webfiltering_arrayocultos[webfiltering_jglobal] = n;
                webfiltering_jglobal++;
                webfiltering_hideparents(pare);
         }
    }
}
```

Recibe como parámetro el nodo al que se le deben ocultar sus antecesores. La ocultación la hace recursivamente hasta llegar al nodo raíz del árbol. El nodo raíz no puede ocultarse, ya que en ese caso la página se mostraría en blanco porque se perdería toda la información que contenía.

1. Obtiene el padre del nodo a ocultar.
2. Comprueba que no sea el nodo raíz.
    a. Si no es el raíz entonces pone su propiedad de visibilidad a oculto, lo guarda en el vector encargado de controlar los elementos que no deben mostrarse (webfiltering_arrayocultos) y se vuelve a llamar a la función, ahora con el nodo padre.
    b. Si es la raíz la función se detiene.

4.1.27. FUNCIÓN PRESIONAR INTRO (FUNCPRESS)

La funcionalidad de filtrado también se puede realizar pulsando la tecla 'Intro' del teclado. Cuando eso ocurre esta función es la encargada de capturar el evento para procesarlo correctamente.

4.1.27.1. EVENTO

Esta función se invoca cuando el usuario pulsa la tecla 'Intro' y se encarga de procesar dicha pulsación y de llamar a la función encargada de realizar el filtrado.

4.1.27.2. FUNCIONALIDAD

A continuación puede verse su código fuente:

```
function webfiltering_funcpress(event){
        if(event.keyCode == event.DOM_VK_RETURN)
                webfiltering_funcslice();
}
```

Comprueba que efectivamente el usuario ha pulsado la tecla 'Intro' y en ese caso llamar a la función 'webfiltering_funcslice()' para que comience con el filtrado.

4.1.28. FUNCIÓN MOSTRAR ELEMENTOS OCULTOS

Esta función se encarga de mostrar todos los elementos que han sido ocultados por algún proceso de filtrado.

4.1.28.1. EVENTO

Esta función se llama desde 'webfiltering_funcslice()' cada vez que se va a realizar un filtrado nuevo, ya que muestra todos los elementos que estaban ocultos e iniciliza a nulo los vectores que almacenan los elementos a mostra y ocultar, así como las variables que controlan su tamaño.

4.1.28.2. FUNCIONAMIENTO

A continuación se adjunta su código:

```
//Muestra el vector de elementos ocultos e inicializa las variables
function webfiltering_mostrarocultos(){
   for(i=0;i<webfiltering_arrayocultos.length;i++)
         //si esta activada la opción mantener estructura se hace visible el nodo
         if(webfiltering_arrayocultos[i]!=null){
                  webfiltering_arrayocultos[i].style.visibility="visible";
                  webfiltering_arrayocultos[i].style.display="";
         }

   // Inicializamos variables
   webfiltering_arrayocultos = new Array (); //elementos que se ocultan
   webfiltering_arraymuestra = new Array ();  //elementos que se muestran
   webfiltering_iglobal=0; //cuenta los elementos de arraymuestra
   webfiltering_jglobal=0;  //cuenta los elementos de arrayocultos
}
```

1. Para cada nodo almacenado en el vector pone su propiedad de visibilidad a visible y la propiedad de display (forma parte de la página) a vacío, indicando que sí forma parte y debe mostrarse.
2. Inicializa los vectores a nulo para que no quede nada del filtrado anterior y pone a cero las variables que controlan sus tamaños.

4.1.29. FUNCIÓN INICIALIZAR

Esta función se ejecuta siempre que se abre el navegador Firefox y se encarga de actualizar todas las variables del diálogo de preferencias, otorgándoles el valor de las variables globales.

4.1.29.1. EVENTO

Esta función se llama desde el navegador Firefox cada vez que se abre, y se encarga de inicializar todas las variables globales que utiliza la aplicación para controlar las preferencias y el valor de las variables del diálogo de preferencias con el mismo valor con el que las había dejado el usuario.

4.1.29.2. FUNCIONAMIENTO

A continuación puede verse su código.

```
function webfiltering_inicializar(){
    webfiltering_setIntegerPreferenceIfNotSet('slice.mantenerestructura', 1);
    webfiltering_setIntegerPreferenceIfNotSet('slice.incrustacion', 0);
    webfiltering_setIntegerPreferenceIfNotSet('slice.tolerance', 0);
    webfiltering_setIntegerPreferenceIfNotSet('slice.keeptree', 1);

    webfiltering_tolerancia = webfiltering_getIntegerPreference("slice.tolerance",true);
    webfiltering_mantenerestructura =
                        webfiltering_getIntegerPreference("slice.mantenerestructura",true);
    webfiltering_formateartamanyoiframes =
                        webfiltering_getIntegerPreference("slice.incrustacion",true);
     webfiltering_mantenerarbol = webfiltering_getIntegerPreference("slice.keeptree",true);

     if (document.getElementById("cuadrotolerancia") != null) {
         document.getElementById("cuadrotolerancia").value = webfiltering_tolerancia;
         document.getElementById("cuadrotol").value = webfiltering_tolerancia;
     }
     if((webfiltering_getIntegerPreference("slice.keeptree", true)==0) &&
       (webfiltering_getIntegerPreference("slice.mantenerestructura", true)==0)){
         document.getElementById("botonmas").disabled = true;
         document.getElementById("botonmenos").disabled = true;
         document.getElementById("cuadrotolerancia").disabled = true;
     }
     else{
         document.getElementById("botonmas").disabled = false;
         document.getElementById("botonmenos").disabled = false;
         document.getElementById("cuadrotolerancia").disabled = false;
    }
}
```

1. Las cuatro primeras instrucciones sirven para otorgar el valor por defecto a las preferencias la primera vez que el usuario ejecuta la aplicación.
2. Obtiene el valor de las preferencias.

```
    webfiltering_tolerancia = webfiltering_getIntegerPreference("slice.tolerance",true);
    webfiltering_mantenerestructura =
                        webfiltering_getIntegerPreference("slice.mantenerestructura",true);
    webfiltering_formateartamanyoiframes =
                        webfiltering_getIntegerPreference("slice.incrustacion",true);
     webfiltering_mantenerarbol = webfiltering_getIntegerPreference("slice.keeptree",true);
```

3. Otorga el valor obtenido de las preferencias a las variables globales y a las variables que aparecen en el cuadro de diálogo de las preferencias.

```
     if (document.getElementById("cuadrotolerancia") != null) {
         document.getElementById("cuadrotolerancia").value = webfiltering_tolerancia;
         document.getElementById("cuadrotol").value = webfiltering_tolerancia;
     }
     if((webfiltering_getIntegerPreference("slice.keeptree", true)==0) &&
       (webfiltering_getIntegerPreference("slice.mantenerestructura", true)==0)){
         document.getElementById("botonmas").disabled = true;
         document.getElementById("botonmenos").disabled = true;
         document.getElementById("cuadrotolerancia").disabled = true;
     }
     else{
         document.getElementById("botonmas").disabled = false;
         document.getElementById("botonmenos").disabled = false;
```

```
        document.getElementById("cuadrotolerancia").disabled = false;
    }
```

4.2. HERRAMIENTAS UTILIZADAS

En este punto se va comentar todas las herramientas utilizadas para llevar a cabo este proyecto y una breve explicación de todas ellas.

4.2.1. ADOBE DREAMWEAVER

Adobe Dreamweaver, abreviado como Dw, es una aplicación destinada a la construcción y edición de sitios y aplicaciones Web.

Anteriormente conocido como Macromedia Dreamweaver, es uno de los programas más utilizados para el diseño y programación Web por estar basado en los estándares de la W3C y por la total integración con otras herramientas de Adobe, como puede ser Adobe Flash, una de las más populares.

La última versión de Dw se vende como parte del paquete Adobe Creative Suite 3, de ahí que se conozca como Adobe Dreamweaver CS3.

Dreamweaver permite crear y administrar sitios Web profesionales y potentes aplicaciones Web desarrolladas en JavaScript. Con este programa se consiguió tener un único entorno de trabajo para crear, diseñar, programar y administrar las aplicaciones Web.

Una de las ventajas que aporta Dw es que permite visualizar en todo momento, mediante un entorno de trabajo visual, el código que se está implementando y viceversa. Facilitando así controlar el resultado final del sitio Web.

Actualmente está disponible para plataformas MAC y Windows, sin embargo con ayuda de API's de Windows también se puede ejecutar en plataformas Unix.

Permite la previsualización de las páginas Web creadas con cualquier navegador que el desarrollador pueda tener instalado en su ordenador personal.

Uno de los motivos que nos impulsó a utilizar esta herramienta para desarrollar la aplicación es que permite crear código JavaScript sin tener muchos conocimientos sobre él mismo.

Las características que lo haces especialmente útil y atractivo para cualquier desarrollador Web es que es un buen administrador de sitios, permitiendo agrupar todos los archivos de un proyecto. Además, posee un cliente FTP integrado que permite subir los proyectos editados directamente a Internet y por úlitmo, pero no menos útil son las funciones de autocompletado y resaltado de la sintaxis propia del lenguaje de programación que se esté utilizando (HTML, JSP, ASP, PHP y JavaScript).

4.2.2. ALZIP

ALZip es un programa compresor gratuito que se caracteriza por trabajar con varios formatos, está completamente en español y posee una interfaz fácil de usar.

ALZip por defecto crea paquetes con la extensión alz pero soporta más de 40 formatos de compresión entre ellos cabe destacar: alz, ace, cab, gz, iso, jar, rar, tar, zip y 7z.
Es capaz de detectar virus gracias a un escáner que posee para la detección de los mismos, su interfaz de línea de comandos se puede personalizar a gusto del usuario, es capaz de crear archivos comprimidos particionados, permite protegerlos con contraseña y completamente compatible con los sistemas operativos Windows y Linux.
En este proyecto se ha utilizado para crear el paquete con extensión jar, con las carpetas content, skin y locale. [ALZip]

4.2.3. WINRAR

WinRAR es la versión para Windows de RAR, un potente programa compresor y descompresor de datos multi-función (dispone de varios modos de compresión). Está desarrollado por los laboratorios RARLAB.
Sirve para comprimir todo tipo de documentos y/o programas de forma que ocupen menos espacio en disco y se puedan transmitir por Internet más rápidamente.
Los archivos comprimidos con este programa llevan la extensión rar. Para este proyecto se necesitaba crear un paquete zip con toda la información de la extensión y posteriormente cambiar la extensión del paquete por xpi. Puesto que WinRAR también permite la creación de ficheros zip se optó por la utilización de dicha herramienta.

Recientemente WinRAR ha recibido el primer puesto en calidad en la comparativa de programas de compresión que realizó la revista mensual Computer Hoy en su número 211. [WinRAR]

Dicha comparativa puede consultarse en el siguiente enlace:
http://www.winrar.es/common/img/awards/ComputerHoyComparativa.png
El resultado final de la comparativa a través de este enlace:
http://www.winrar.es/common/img/awards/ComputerHoyResumen.png

4.2.4. SCREENHUNTER

ScreenHunter es una herramienta sencilla que permite capturar rápidamente imágenes y capturas de pantalla en 15 formatos diferentes, entre los cuales cabe destacar gif, png y jpg.
Diseñada para capturar exactamente el trozo de imagen que uno necesita y desea en cada momento por pequeño que sea. La captura se realiza en un único paso, lo que lo hace una herramienta precisa y rápida.
Además posee un visualizar integrado que permite editar imágenes, modificar la zona de captura, aumentar y/o reducir el tamaño de las imágenes y añadir texto a las mismas. [ScreenHunter]

Debido a su precisión, rapidez y fácil manejo todas las imágenes de este proyecto han sido capturadas con dicha herramienta.

5. BIBLIOGRAFÍA

[AlZip] Vergara Kevin. "AlZip: compresor gratuito que soporta ACE, RAR, ZIP y más". Blog Informático (Septiembre, 2008). Disponible en: http://www.bloginformatico.com/alzip-compresor-gratuito-que-soporta-ace- rar-zip-y-mas.php

[CSS W3C, 2008] W3C (2005), "Guía breve de CSS" (Abril, 2008). Disponible en: http://www.w3c.es/divulgacion/guiasbreves/HojasEstilo

[DOM, 2008] W3c, 1998. "Especificación del modelo de objeto (DOM), nivel 1". Versión 1.0. (Marzo, 2008). Disponible en: http://html.conclase.net/w3c/dom1-es/cover.html

[Eguíluz, 2008] Eguíluz Pérez, Javier. "Introducción a CSS". Online. (Abril, 2008). Disponible: http://www.librosweb.es/css/. [Capítulo 1]

[Mozilla] Mozilla Europe, Mozilla Foundation (Junio, 2008). "Mozilla bate el récord Guinness de descargas con Firefox 3". Mozilla Europe – Mozilla Firefox. (Consulta: Julio 2008). Disponible en: http://www.mozilla-europe.org/es/press/2008/07/02/1181-mozilla-bate-el-record-guinness-de-descargas-con-firefox-3

[Mozilla Developer Center, 2008] "Portada, MDC" (Marzo, 2008). Disponible en: http://developer.mozilla.org/es

[Historia navegadores] "Breve historia de los navegadores" (Junio, 2008). Disponible en: http://www.consumer.es/web/es/tecnologia/internet/2005/10/26/146455.php

[Historia] Redacción (2006). "Un poco de historia: los primeros navegadores". Tuexperto.com. (Consulta: Junio, 2008). Disponible en: http://www.tuexperto.com/2006/10/20/un-poco-de-historia-los- primeros-navegadores/

[Burns, 2000] Burns, Joe. "Descubre JavaScript", edición 2000; Ed. Prentice Hall.

[Powell, 2002] Powell, Thomas A. "JavaScript: manual de referencia", edición 2002; Ed. McGraw-Hill/Interamericana.

[ScreenHunter] Buen capturador de imágenes en su versión gratuita (Septiembre, 2008) http://screenhunter-free.softonic.com/

[WebDeveloper, 2008] Duplika. "WebDeveloper". Difunde Firedox. (Consulta: Septiembre, 2008). Disponible en: http://www.difundefirefox.com/extensiones/web-developer

[Wikipedia, 2008] "Wikipedia, la enciclopedia libre" (Abril, 2008). Disponible en: http://www.wikipedia.org

[WinRAR] "WinRAR España.Sitio oficial en español" (septiembre, 2008). Disponible en: http://www.winrar.es/

6. ANEXO A: CÓDIGO FUENTE

En este anexo se muestran las funciones que no se han explicado en el punto de implementación, por ser muy parecidas a las explicadas en ese punto.

6.1.1.1. FUNCIÓN ANTERIOR

```
function webfiltering_anterior(){
    var filtrado=0;

webfiltering_AplicadoAnterior=1; //se ha aplicado anterior

    if(webfiltering_sliceAplicado==1){ //se ha aplicado filtrado
        //almacenamos la pagina filtrada anteriormente
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        webfiltering_sliceAplicado=0;
        filtrado=1;
    }
    else{
        //este if solo se ejecutara cuando se trate de la primera página
        if(webfiltering_paginaOriginal==0){ //me guardo  la página
                webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
                webfiltering_URLActual=webfiltering_guardarURL(content.document);
        }
    }

    var urlActual=content.document.URL;
    if(urlActual != webfiltering_URLActual){ //si son distintas URL es que se ha cambiado de
                                              página y la tengo que volver a guardar
        webfiltering_paginaOriginal=webfiltering_guardarPag(content.document);
        webfiltering_URLActual=webfiltering_guardarURL(content.document);
    }

    var pag=webfiltering_paginaOriginal;
    webfiltering_formateartamanyoiframes =
                                webfiltering_getIntegerPreference("slice.incrustacion",true);

    webfiltering_texto = document.getElementById("cuadroblanco").value; //guarda en la
                                          variable texto el contenido del cuadro de slice
    if (webfiltering_texto==""){
        alert("El criterio de busqueda es nulo, debe introducir al menos, una palabra");
        return;
    }

    webfiltering_formateartamiframes(content.document);
    var bodyText;

    if((webfiltering_texto != webfiltering_palabraanterior) ||
      (urlActual != webfiltering_URLActual) || (filtrado == 1)){
        bodyText=pag; //creamos un nuevo body para que reemplace al original
        bodyText=webfiltering_buscarpalabra(bodyText,webfiltering_texto);
        webfiltering_contador=webfiltering_numPalabrasResaltadas;
    }
    else {
        if(webfiltering_AplicadoSiguiente == 1){
                webfiltering_contador=webfiltering_contadorSig-1;
                webfiltering_AplicadoSiguiente=0;
        }
        bodyText= content.document.body.innerHTML; //tengo la misma palabra. Me tengo
                                                    que guardar el texto con las anclas
    }
```

```
    if(webfiltering_palabrasResaltadas.length== 0||
       webfiltering_numPalabrasResaltadas==0)
         alert("La palabra buscada no se encuentra en la pagina");

    content.document.body.innerHTML = bodyText; //reemplazamos  el body de la
                                                  página, por el nuevo con las anclas

    webfiltering_palabraanterior=document.getElementById("cuadroblanco").value;
    webfiltering_numPalabrasFiltradas++;

    //guarda un histórico de lo que se ha buscado
    webfiltering_criteriobusqueda=document.getElementById("cuadroblanco");
    if(webfiltering_numPalabrasFiltradas == 1){
        webfiltering_criteriobusqueda.appendItem(webfiltering_texto,webfiltering_texto);
        webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                              webfiltering_texto; //almacena la palabra buscada en el vector

    }
    else{
        var word=webfiltering_texto;
        var existe=false; //la palabra existe
        var i=1;
        while((i <= webfiltering_numPalabrasFiltradas) && (existe == false)){
                if(word == webfiltering_palabrasBuscadas[i] ) existe=true;
                i++;
        }
        if(existe == false){
                if(webfiltering_numPalabrasFiltradas >= 10) {
                        webfiltering_criteriobusqueda.removeItemAt(0);
                }
                webfiltering_palabrasBuscadas[webfiltering_numPalabrasFiltradas]=
                                                                    webfiltering_texto;
                webfiltering_criteriobusqueda.appendItem
                                                    (webfiltering_texto,webfiltering_texto);
        }
    }
     webfiltering_resaltarpalabraanterior(content.document,webfiltering_contador);
}
```

6.1.1.2. FUNCIÓN RESALTAR PALABRA ANTERIOR

```
function webfiltering_resaltarpalabraanterior(bodyText,contador){
    var numeroAnclas = bodyText.anchors.length;

    if(numeroAnclas == 0) return;

    if(webfiltering_AplicadoSiguiente == 1){ //ponemos la palabra actual sin fondo
        bodyText.anchors[contador].style.background="none";
    }

    if(contador == -1) {
        bodyText.anchors[contador+1].style.background="none";
        contador=numeroAnclas-1; //lo igualamos al número de anclas para volver a
                                   empezar desde el final
        webfiltering_contador=contador;
    }

    var devolver = webfiltering_palabrasResaltadas[contador];
    bodyText.anchors[contador].setAttribute('href','#devolver');

    if(contador < numeroAnclas-1){
        bodyText.anchors[contador+1].style.background="none";
```

```
    }
    bodyText.anchors[contador].style.background="yellow";

    if(contador >= 0) {
        webfiltering_contadorAnt=webfiltering_contador;
        webfiltering_contador--;
    }
    return bodytext;
}
```

6.1.1.3. FUNCIÓN DISMINUIR TOLERANCIA

```
//se activa desde el botón disminuir tolerancia, decrementa la tolerancia en uno
function webfiltering_disminuirtolerancia(){
    if(webfiltering_tolerancia>0){
        webfiltering_tolerancia--;
        webfiltering_setIntegerPreference("slice.tolerance",webfiltering_tolerancia);
        document.getElementById("cuadrotolerancia").value = webfiltering_tolerancia;
        webfiltering_funcslice();
    }
}
```

6.1.1.4. FUNCIONES GUARGAR FRAME E IFRAME ORIGINAL

```
//función encargada de almacenar el cuerpo de un frame original, sin resaltado
function webfiltering_guardarFrameOrig(frame){
        return frame.contentDocument.body.innerHTML;
}
```

```
//función encargada de almacenar el cuerpo de un iframe original, sin resaltado
function webfiltering_guardarIframeOrig(iframe){
        return iframe.contentDocument.body.innerHTML;
}
```

7. ANEXO B: CÓDIGO FUENTE DE LOSARCHIVOS XUL

En este anexo se muestra el código fuente de todos los ficheros XUL utilizados por la aplicación.

7.1.1.1. ARCHIVO SLICETOOLBAR

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="chrome://slicetoolbar/skin/slicetoolbar.css" type="text/css"?>

<overlay id="Slice-Overlay"
       xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">

<keyset>
  <key id="cargarbarra" modifiers="control" key="Y"
                                        oncommand="webfiltering_mostrarbarra()"/>
</keyset>

   <script type="application/x-javascript"
         src="chrome://slicetoolbar/content/slicetoolbar.js" />

   <script type="application/x-javascript"
         src="chrome://slicetoolbar/content/preferences.js" />


   <toolbox id="navigator-toolbox">

      <toolbar id="barra" toolbarname="Web Filtering Toolbar" accesskey="T"
            class="chromeclass-toolbar" context="toolbar-context-menu"
            hidden="false" persist="hidden" align="center"
            key="cargarbarra">

         <toolbaritem flex="0">

         </toolbaritem>

                  <label value="Criteria:"/>

                  <toolbaritem id="cuadro" persist="width">
                           <menulist id="cuadroblanco" editable="true" flex="1"
                            minwidth="100" width="250"
                            onkeypress="webfiltering_funcpress(event)">
                           <menupopup/>
                           </menulist>
        </toolbaritem>

                  <toolbarbutton id="botonhighlight" tooltiptext="Highlight coincidences"
                      label="Highlight" oncommand="webfiltering_match()" />

                  <toolbarbutton id="botonanterior" tooltiptext="Anterior"
                      oncommand="webfiltering_anterior()" />

                  <toolbarbutton id="botonsiguiente" tooltiptext="Siguiente"
                      oncommand="webfiltering_siguiente()" />

                  <toolbarbutton id="botonslice" tooltiptext="Filter this webpage"
                      label="Filter" oncommand="webfiltering_funcslice()" />
                  <label value="Tolerance:"/>

                  <toolbarbutton id="botonmas" tooltiptext="Increase tolerance"
                      oncommand="webfiltering_aumentartolerancia()" />

                  <toolbaritem id="cuadrotol" persist="width">
                           <menulist id="cuadrotolerancia" editable="true" flex="1" width="70"
                                    onkeypress="webfiltering_funcpress(event)">
                            <menupopup>
                           <menuitem id="tol0" label="0"
                                    class="menuitem" tooltiptext="0"
```

```
                                    oncommand="webfiltering_actualizartolerancia(0)"/>

                            <menuitem id="tol1" label="1"
                                    class="menuitem" tooltiptext="1"
                                    oncommand="webfiltering_actualizartolerancia(1)"/>

                            <menuitem id="tol2" label="2"
                                    class="menuitem" tooltiptext="2"
                                    oncommand="webfiltering_actualizartolerancia(2)"/>

                            <menuitem id="tol3" label="3"
                                    class="menuitem" tooltiptext="3"
                                    oncommand="webfiltering_actualizartolerancia(3)"/>

                            <menuitem id="tol4" label="4"
                                    class="menuitem" tooltiptext="4"
                                    oncommand="webfiltering_actualizartolerancia(4)"/>

                            <menuitem id="tol5" label="5"
                                    class="menuitem" tooltiptext="5"
                                    oncommand="webfiltering_actualizartolerancia(5)"/>
                </menupopup>
              </menulist>
            </toolbaritem>


            <toolbarbutton id="botonmenos" tooltiptext="Decrease tolerance"
                        oncommand="webfiltering_disminuirtolerancia()" />

            <toolbarbutton id="botonver" tooltiptext="Show the full page"
                        label="Show The Full Page" oncommand="webfiltering_verpagentera()" />


        <toolbarbutton id="optionsbutton" label="Options" tooltiptext="Options"
                    type="menu">
        <menupopup>
          <menuitem id="options" accesskey="" label="Options..." tooltiptext="Options"
                    oncommand="webfiltering_slice_options()"/>
            <menuseparator id="separator"/>

            <menuitem id="help" accesskey="" label="Help..." tooltiptext="Help"
           oncommand="webfiltering_help('chrome://SliceToolbar/content/help/helpenglish.html','','toolba
r=no,location=no,status=no,menubar=no,scrollbars=yes,height=700,width=1000')"/>

            <menuitem id="about" accesskey="" label="About..." tooltiptext="About"
                    oncommand="webfiltering_about()"/>
          </menupopup>
        </toolbarbutton>

            <toolbarbutton id="botoncerrar" tooltiptext="close"
                        oncommand="webfiltering_ocultarbarra()" />

          <toolbarspring />

        </toolbar>
      </toolbox>
</overlay>
```

7.1.1.2. ARCHIVO ABOUT

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="chrome://SliceToolbar/content/about/about.css" type="text/css"?>


 <dialog buttons="accept"
        id="slice-about-dialog"
        title="About Web Filtering Toolbar"
        xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">
```

```
    <script type="application/x-javascript"
          src="chrome://webdeveloper/content/common/preferences.js"/>
    <script type="application/x-javascript"
          src="chrome://webdeveloper/content/about/about.js"/>

    <vbox id="webdeveloper-about-details">
       <hbox>
          <description class="name">Web Filtering Toolbar 1.3</description>
       </hbox>

       <description class="date">September 2008</description>

       <description>Josep Silva Galiana</description>
       <description>Mercedes García Martínez</description>
       <description>Tatiana Tomás Balsebre</description>

    </vbox>
    <separator class="groove"/>
</dialog>
```

7.1.1.3. ARCHIVO OPTIONS

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="chrome://SliceToolbar/content/options/options.css" type="text/css"?>

<dialog buttons="accept, cancel"
      id="slice-options-dialog"
            title="Options Web Slicing Toolbar"
            onload="webfiltering_seeOptions()"
            ondialogaccept="webfiltering_saveOptions()"
      xmlns="http://www.mozilla.org/keymaster/gatekeeper/there.is.only.xul">

 <script type="application/x-javascript"
         src="chrome://slicetoolbar/content/slicetoolbar.js" />

 <script type="application/x-javascript"
         src="chrome://slicetoolbar/content/preferences.js" />

<groupbox>

      <checkbox id="mantenerestructuraopciones" label="Keep Structure"/>
        <toolbarbutton id="ayuda1" tooltiptext="What is this?"
                  label="What is this?"
oncommand="webfiltering_help('chrome://SliceToolbar/content/help/helpenglish.html','','toolbar=no,loc
ation=no,status=no,menubar=no,scrollbars=yes,height=700,width=1000')" />

       <checkbox id="formateartamanyoiframesopciones" label="Format Size iFrames"/>
         <toolbarbutton id="ayuda2" tooltiptext="What is this?"
                  label="What is this?"
oncommand="webfiltering_help('chrome://SliceToolbar/content/help/helpenglish.html','','toolbar=no,loc
ation=no,status=no,menubar=no,scrollbars=yes,height=700,width=1000')" />

          <checkbox id="mantenerarbol" label="Keep Tree"/>
      <toolbarbutton id="ayuda3" tooltiptext="What is this?"
                  label="What is this?"
oncommand="webfiltering_help('chrome://SliceToolbar/content/help/helpenglish.html','','toolbar=no,loc
ation=no,status=no,menubar=no,scrollbars=yes,height=700,width=1000')" />

          <checkbox id="mantenerbarravisible" label="Toolbar visible"/>
      <toolbarbutton id="ayuda4" tooltiptext="What is this?"
                  label="What is this?"
oncommand="webfiltering_help('chrome://SliceToolbar/content/help/helpenglish.html','','toolbar=no,loc
ation=no,status=no,menubar=no,scrollbars=yes,height=700,width=1000')" />
       <separator/>
       <hbox align="center">
          <label control="" value="Tolerance"/>
                <toolbaritem id="cuadrotol2">
          <menulist id="tol" editable="true" flex="1">
```

```
            <menupopup>
                    <menuitem label="0" value="0"/>
                <menuitem label="1" value="1"/>
                <menuitem label="2" value="2"/>
                <menuitem label="3" value="3"/>
                            <menuitem label="4" value="4"/>
                <menuitem label="5" value="5"/>
            </menupopup>
                </menulist>
                    </toolbaritem>

        </hbox>
<toolbarbutton id="ayuda3" tooltiptext="What is this?"
                        label="What is this?"
oncommand="webfiltering_help('chrome://SliceToolbar/content/help/helpenglish.html','','toolbar=no,loc
ation=no,status=no,menubar=no,scrollbars=yes,height=700,width=1000')" />

    </groupbox>
<groupbox>
                    <toolbarbutton id="botondefaults" tooltiptext="Load Default Options"
                        label="Defaults" oncommand="webfiltering_load_defaults()" />
</groupbox>

</dialog>
```

8. ANEXO C: CÓDIGO DE WEBDEVELOPER

Para poder comunicar la barra de herramientas con el cuadro de diálogo de las preferencias se acudió al software libre WebDeveloper.
WebDeveloper es una herramienta para Firefox que consiste en una barra de herramientas orientada a la ayuda para la realización de páginas Web. De esta herramientas han sido especialmente útiles los métodos 'setIntegerPreference' y 'getIntegerPreference' que han ayudado a guardar el valor de las preferencias y ha actualizar el cuadro de la tolerancia. Para más información acerca de la herramienta y como conseguirla dirigirse a [WebDeveloper, 08].

A continuación se muestra el código que se implementó para la captura y actualización de los valores del cuadro de opciones.

```
var webfiltering_preferencesService = null;

// Gets an integer preference, returning 0 if the preference is not set
function webfiltering_getIntegerPreference(preference, userPreference){

        if(preference) { // If the preference is set
                // If not a user preference or a user preference is set
                if(!userPreference || webfiltering_isPreferenceSet(preference)){
                    try{
                        return webfiltering_getPreferencesService().getIntPref(preference);
                    }
                    catch(exception){
                        // Do nothing
                    }
                }
        }
    return 0;
}

// Sets an integer preference
function webfiltering_setIntegerPreference(preference, value){

    if(preference) {  // If the preference is set
        webfiltering_getPreferencesService().setIntPref(preference, value);
    }
}

// Sets an integer preference if it is not already set
function webfiltering_setIntegerPreferenceIfNotSet(preference, value){

    if(!webfiltering_isPreferenceSet(preference)) {   // If the preference is not set
        webfiltering_setIntegerPreference(preference, value);
    }
}

// Is a preference set
function webfiltering_isPreferenceSet(preference){

    if(preference){ // If the preference is set
        return webfiltering_getPreferencesService().prefHasUserValue(preference);
    }

    return false;
}

// Gets the preferences service
```

```
function webfiltering_getPreferencesService() {

    if(!webfiltering_preferencesService){    // If the preferences service is not set
          webfiltering_preferencesService = Components.classes["@mozilla.org/preferences-
          service;1"].getService(Components.interfaces.nsIPrefService).getBranch("");
    }
    return webfiltering_preferencesService;
}
```