UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

TESINA DE LICENCIATURA

TÍTULO: Servicios de Búsqueda Web adaptables a los usuarios finales

AUTORES: Agustín Cipollone

DIRECTOR: Sergio Firmenich

CODIRECTOR: Gabriela Bosetti

ASESOR PROFESIONAL: -

CARRERA: Licenciatura en Sistemas

Resumen

En la presente investigación se analiza el concepto de búsqueda Web y como los usuarios interactúan con el navegador para realizar este tipo de tareas, introduciendo además el concepto de búsquedas auxiliares. Luego se relevan los últimos avances en el area y cómo intentan solucionar las limitaciones existentes. Aspectos específicos de estos enfoques se integran en una propuesta superadora que se presenta a traves de una herramienta, cuyo objetivo principal es mejorar la experiencia del usuario, permitiendole definir sus propios servicios de búsqueda para realizar consultas a motores de búsqueda externos y obtener los resultados in situ, . Por otra parte, el enfoque ofrece diferentes visualizaciones que intentan adaptarse mejor a las preferencias del usuario final.

Palabras Clave

Búsqueda Web, Servicios de Búsqueda, Búsqueda Primaria, Búsqueda Auxiliar, Browsing, Web Extension, Web Augmentation, Web Scraping, HTML, DOM, Iframe, Xpath, WOA, Datatables, Javascript, GOMS-Keystroke, Notación HAMSTER.

Trabajos Realizados

- Se realizó un exhaustivo análisis de los antecedentes, detallando los aportes especificos que realizan.

- Se presentó el enfoque con soporte en una herramienta que permite definir servicios que encapsulan motores de búsqueda externos y retornan los resultados in situ. Además ofrece distintas visualizaciones y se adapta tanto a las preferencias del usuario como a los objetos del dominio.

- Se evaluó cuantitativamente la mejora que representa al usuario el uso de la herramienta.

Conclusiones

- Al analizar los antecedentes, en comparación con la propuesta presentada, queda en evidencia la originalidad del enfoque, ya que presenta aportes concretos en el área.

- La herramienta diseñada e implementada es un trabajo en desarrollo y, si bien no esta finalizada para estar productiva, demuestra la fiabilidad del enfoque y de cada uno de sus aspectos princiaples.

- El actual trabajo sienta las bases para trabajos futuros que permitan explotar aún más los beneficios aportados.

Trabajos Futuros

- Brindar soporte a mayor cantidad de interfaces de motores de busquedas que sean de diversos dominios.

- Integrar a la herramienta aspectos relacionados al filtrado, ordenamiento y paginación.

- Diseño e implementación de la herramienta en plataformas moviles.

-Mejorar mecanismos de colabración que permitan compartir los servicios de búsqueda entre los propios usuarios.

Fecha de la presentación: Febrero 2020

Tabla de contenidos

Capítulo 1: Introducción

1.1 Motivación

Realizar búsquedas en la web se ha convertido en una tarea que solemos hacer a diario. Según los rankings que miden el tráfico en Internet (Alexa en 2016 o SimilarWeb en 2018) en los primeros 10 puestos figuran motores de búsqueda como Google, baidu.com (China) o yandex.ru (Rusia), los cuales pertenecen a la categoría *Search Engine*. Pero además en el resto de los sitios Web más accedidos pero de otras categorías (redes sociales, videos, documentos, sitios *e-commerce,* etc) no están exentos de las búsquedas sino que, al contrario, estas resultan parte fundamental en el uso de la mayoría de estos sitios.

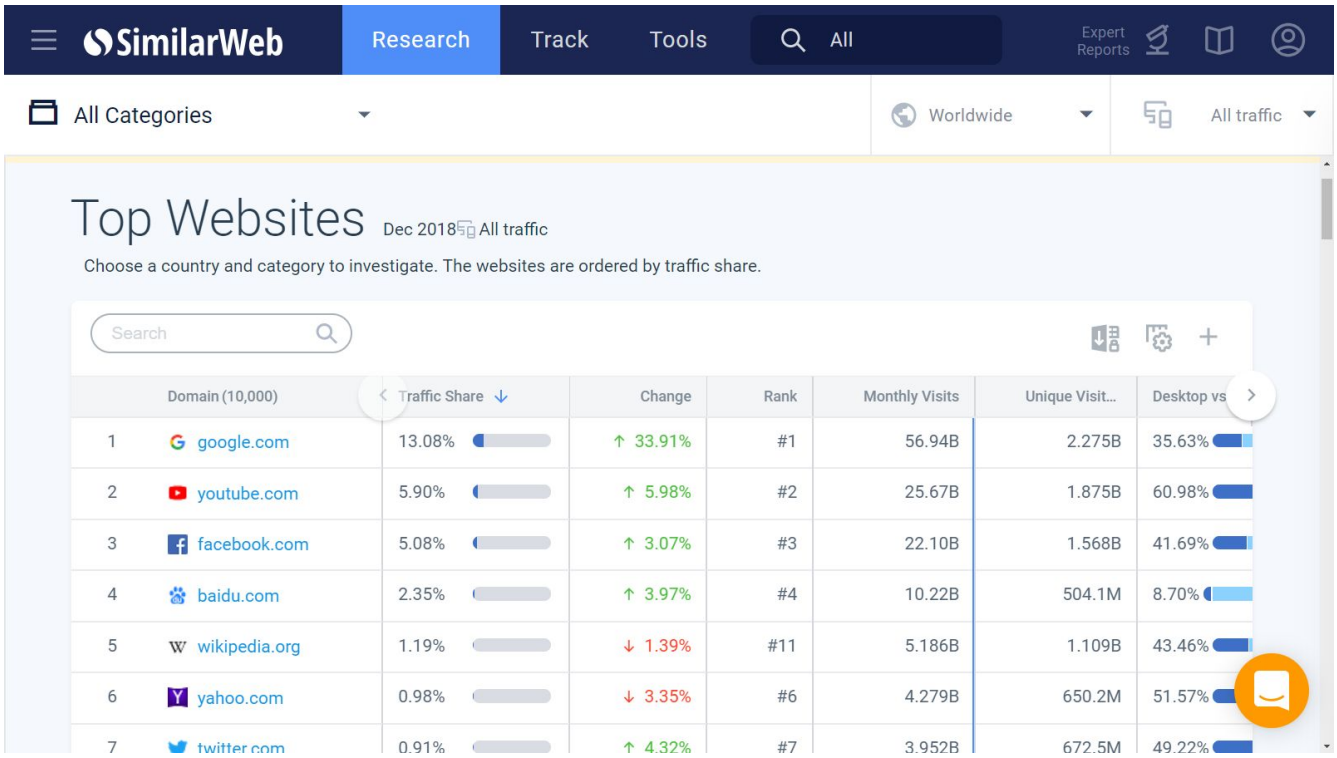


Figura 1.1: Ranking de los Top Websites de Diciembre 2018 ordenado por tráfico de visitas

Los navegadores web y los motores de búsqueda son las herramientas principales que usan las personas para acceder a la vasta información disponible online. Pero no todas las búsquedas son iguales, investigadores del laboratorio informático de Toulouse (IRIT) [1] explican la diferencia entre una búsqueda primaria y una auxiliar, la cual de manera introductoria se puede resumir en que las primeras corresponden a una necesidad primaria del usuario mientras que las últimas surgen a partir de los resultados obtenidos en búsquedas previas y/o información que contiene un sitio Web al momento que el usuario se encuentra navegando en este.

Las búsquedas auxiliares suelen crear contextos anidados que incrementan la distancia articulatoria entre los usuarios y su objetivo final. Cuando surge la necesidad por parte del usuario de realizar una búsqueda de este tipo, podemos descomponer dicho proceso en los siguientes pasos:

i) Salir del contexto; ii) Abrir una nueva ventana o pestaña; iii) Formular una consulta; iv) Procesar la consulta; v) Obtener la información buscada.

Dado que cada paso mencionado conlleva un tiempo específico y teniendo en cuenta que cada búsqueda puede desencadenar en nuevas búsquedas auxiliares, el proceso se vuelve más complejo. Como consecuencia directa de esto ocurre lo mencionado por investigadores de la Universidad de Tampere (Finlandia) [2] donde se comprueba estadísticamente que es muy común tener muchas ventanas o pestañas de navegación abiertas durante un proceso de búsqueda.

Consideremos el caso concreto de un estudiante que está leyendo un artículo en Wikipedia dentro del cual se cita una frase de un autor (por ej. Eduardo Galeano) que capta particularmente su atención. Como consecuencia directa de esta primera interacción, son múltiples las nuevas búsquedas que pueden dispararse. El usuario puede requerir buscar

precios en algún sitio *e-commerce* en el cual suele realizar sus compras (por ejemplo Amazon). Para ello debe realizar los siguientes pasos:

i) Resaltar el nombre del autor y copiarlo; ii) Abrir una nueva pestaña o ventana en el navegador; iii) Ingresar la URL de Amazon; iv) Pegar el nombre del autor sobre el buscador de Amazon; v) Procesar la consulta y obtener los resultados.

A partir de los resultados obtenidos pueden continuar las búsquedas complementarias. Siguiendo con el ejemplo, con los resultados obtenidos de la búsqueda previa, el usuario podría buscar si alguno de los libros se encuentra disponible en el catálogo de la biblioteca de su universidad (por ej. el catálogo de la UNLP). Para ello debe repetir pasos similares:

i) Abrir una nueva pestaña en navegador; ii) Ingresar la URL del sitio web de la biblioteca; iii) Ingresar el título del libro sobre el buscador del catálogo de libros del sitio Web
iv) Procesar la consulta y obtener los resultados

Como podemos observar, por cada búsqueda que requiera el usuario se vuelven a repetir los mismos pasos que implican cambiar el contexto actual, obligándolo a desviar el foco de atención del contexto original (en este caso el documento en Wikipedia), al cual eventualmente retornará para continuar con su tarea original. Además, a mayor complejidad de la tarea de búsqueda mayor será la necesidad de este tipo de búsquedas auxiliares, por ejemplo pensar en todas las búsquedas que involucra planificar un viaje o realizar un trabajo de investigación a través de la Web.

En esto último se encuentra una nueva perspectiva que intenta optimizar las búsquedas tal y como funcionan hoy en dia, proponiendo una solución concreta en la cual el usuario podrá definir sus propios servicios de búsqueda Web a partir de cualquier sitio que tenga un motor de búsqueda y visualizar los resultados *in situ*, todo esto a través de un enfoque y su correspondiente herramienta que se presentan en este trabajo.

En lo que respecta al modo en que se visualizan los resultados, en la actualidad los usuarios suelen adaptarse a las decisiones de los desarrolladores de los sitios en los cuales realizamos las búsquedas. Por ende la novedad de la propuesta no radica únicamente en hacer consultas a motores de búsqueda externos y que los resultados sean visualizados *in situ* sino que a raíz de esto se propone poner al alcance del usuario final un abanico de posibilidades para que sea este quien decida de qué forma visualiza el contenido en base a sus preferencias, dependendiendo de que dominio específico se obtienen los resultados.

En este enfoque se propone que el usuario tenga la potestad de personalizar tanto el contenido que se muestra así como también de qué forma se muestra. Por un lado el conjunto de resultados serán tratados como objetos del dominio (por ej. Libros) con sus respectivas propiedades (por ej. titulo, autor, precio) y será el usuario quien determine cuáles son las propiedades que conforman el estado del objeto. Por el otro lado, el usuario podrá optar de que forma visualizar el contenido, por ejemplo, en forma de carrusel de imágenes, de gráficos estadísticos o de simples tablas que se generarán en base a los datos resultantes.

1.2 Objetivos

Como consecuencia de las limitaciones mencionadas anteriormente con las que se enfrenta el usuario a la hora de realizar búsquedas en la Web, y sobre todo búsquedas auxiliares, se investigan múltiples aspectos que pueden ser mejorados. A pesar de que algunos de ellos ya fueron tratados de forma aislada, en este trabajo se propone una solución transversal ante estas carencias actuales y una mejora que intente adaptarse a las preferencias de cada usuario en particular.

Basado en lo dicho anteriormente se plantea como objetivo general de este trabajo investigar toda posible mejora en la interacción del usuario con el navegador al momento de realizar tareas de búsqueda en la Web y desarrollar una herramienta para aplicar este enfoque.

Este objetivo general a su vez se puede descomponer en los siguientes objetivos específicos que tiene el actual trabajo:

- Analizar los antecedentes provistos hasta el momento tanto para realizar consultas desde cualquier sitio Web a motores de búsqueda externos así como también para visualizar los resultados de estas consultas *in situ*.

- Analizar, diseñar e implementar un mecanismo que permita definir servicios de búsqueda que encapsulan motores de búsquedas externos escogidos por el usuario.

- Analizar y diseñar e implementar un mecanismo para visualizar los resultados de las búsquedas externas dentro del contexto de cualquier sitio Web desde el cual se utilicen los servicios de búsqueda previamente definidos.

- Permitir adaptar el contenido de los resultados en base a las preferencias del usuario.

- Integrar la abstracción de los resultados de búsqueda como objetos del dominio para posibilitar interacciones más fructíferas con el usuario.

1.3 Estructura del documento

Ya realizada la introducción donde se ha explicado la motivación de este trabajo y se han descrito los objetivos, a continuación se detalla la estructura de la tesis y el contenido de cada capítulo:

- Capítulo 2: Búsquedas Web. Se detalla que son y cómo se clasifican las búsquedas Web. Se explican las búsquedas desde una perspectiva más general, comenzando por conocer cuál es la motivación de un usuario al realizar una tarea de búsqueda Web, para luego profundizar en las estrategias más utilizadas para satisfacer sus necesidades. Luego se analiza más en detalle las tareas de búsqueda Web utilizando una notación de modelos de tareas llamada HAMSTER que permite desglosar la búsqueda Web en distintas subtareas para comprender mejor el proceso. Finalmente se introduce el concepto de búsquedas auxiliares y se las diferencia de las primarias.

- Capítulo 3: *Trabajos relacionados*. En este capítulo se recopilan y se analizan los avances que han habido en el área hasta el momento, tanto en lo que respecta a las búsquedas Web como también a la posterior visualización de los resultados obtenidos. Este *background* de las investigaciones, los enfoques y las herramientas existentes, será de utilidad no solo para conocer las tendencias sino para comprender cuales son las mejoras que el trabajo propuesto aporta en el área de las búsquedas Web y cómo integra las virtudes de otros enfoques.

- Capítulo 4: Presentación del enfoque. Se presenta el enfoque propuesto en el actual trabajo. Se introduce el concepto de servicios de búsqueda y luego se desarrolla la arquitectura, la cual se divide en 2 capas principales: la capa de definición de servicios de búsqueda y la capa de visualización. Además a lo largo del capítulo se detalla qué aspectos se toman de los enfoques ya existentes y como se los mejora e integra en una única propuesta superadora.

- Capítulo 5: Presentación de la herramienta . Se presenta la herramienta desarrollada como una *Web extension*, que da soporte al enfoque propuesto. A partir de un ejemplo concreto de cómo es la interacción de los usuarios con la herramienta, se demuestra

la fiabilidad del trabajo, ya que se logran implementar gran parte de los aspectos que ofrece la propuesta, y que además sienta las bases para futuras mejoras.

- Capítulo 6: Análisis y posibles mejoras. En base a una evaluación cuantitativa se analiza la mejora concreta en la experiencia del usuario final al utilizar herramienta que da soporte al enfoque. Luego se analizan cuales son las posibles mejoras y futuros trabajos, tomando como punto de partida el enfoque presentado y los aspectos ya implementados por la herramienta.

Capítulo 2: Búsquedas Web

2.1 Introducción

En este capítulo se desarrolla un concepto crucial del actual trabajo como son las búsquedas Web. Desde la perspectiva de diferentes autores se analiza cuales son los objetivos del usuario al momento de realizar una búsqueda en la Web y de qué manera intenta alcanzarlos. Se comienza describiendo las distintas categorías de búsquedas en general, para luego profundizar en las estrategias y los tipos de búsqueda que derivan de estas. Luego se analiza más en detalle las tareas de búsqueda Web utilizando una notación de modelos de tareas llamada HAMSTER que permite desglosar la búsqueda Web en distintas subtareas para comprender mejor el proceso. Por último se definen y se diferencian las búsquedas primarias de las auxiliares y los motores de búsqueda genéricos de los de dominio específico.

2.2 Categorías de búsquedas Web

Como un primer acercamiento a las búsquedas Web trataremos de entender porque los usuarios realizan búsquedas, es decir cuál es su motivación a la hora de interactuar con un motor de búsqueda.

Para describir esto último Daniel Rose y Danny Levinson explican que se pueden organizar las búsquedas en diferentes categorías [3]. En base a estadísticas han diferenciado 3 tipos de categorías de búsquedas: navegacional, informacional y transaccional. A continuación se definen y ejemplifican las mismas:

1. **Navegacional**: Mi objetivo es ir a un sitio web específico previamente conocido. La única razón por la que hago la busqueda es porque es más ágil que tipear la URL o quizá porque no conozco la URL exacta.
    - Ejemplos de búsquedas navegacionales:
        - *Aloha airlines*
        - *Harvard University*
2. **Informacional**: Mi objetivo es aprender algo leyendo o viendo sitios Web
    a. Directa: Quiero aprender algo en particular sobre un tema. Puede ser de 2 tipos:
        i. Cerrado: Quiero obtener una respuesta a una pregunta que tiene una respuesta simple y no ambigua. Ejemplo: “Fecha próximas elecciones 2019”
        ii. Abierta: Quiero obtener una respuesta a una pregunta sin restricciones en cuanto a la profundidad. Ejemplo “porque los metales preciosos brillan”
    b. Indirecta: Quiero aprender sobre cualquier cosa o todo lo que haya relacionado a un tema. Una consulta de este tipo podría ser interpretada como “cuéntame sobre determinado tema”
        i. Ejemplo: “racismo”
    c. Asesorar: Quiero un consejo, ideas, sugerencias o instrucciones:
        i. Ejemplo: “cómo dejar de fumar”
    d. Ubicación: Mi objetivo es encontrar un lugar físico donde puedo obtener un servicio o producto
        i. Ejemplo: “tarjeta sube”
    e. Lista: Mi objetivo es encontrar una lista de sitios Web sugeridos, los cuales puedan ser candidatos para ayudarme a conseguir algún objetivo final
        i. “lugares para visitar en Amsterdam”

ii. “darios de Córdoba”

3. **Recursos/Transaccional**: Mi objetivo es obtener un recurso (no información) disponible en páginas Web.
   a. Descarga: Mi objetivo es descargar un recurso en mi computadora u otro dispositivo
      i. Ejemplo: “ccleaner”
   b. Entretenimiento: Mi objetivo es entretenerme viendo items disponibles en la página resultado
      i. Ejemplo: “película Disney”
   c. Interaccion: Mi objetivo es interactuar con un recurso usando algún programa o servicio disponible en el sitio web que encuentre: “conversor de monedas”

Esta clasificación de las búsquedas Web se focaliza en la primer interacción (cualquiera sea el objetivo final) entre el usuario y el motor de búsqueda, proceso que se conoce como búsqueda primaria. Sin embargo esto último simplemente puede ser el comienzo de un proceso de búsqueda más complejo, sobre todo en lo que respecta a las búsquedas informacionales que como se ve en la Fig.2.1 son las más comunes, y además pueden ser simplemente un disparador a la hora de recuperar información de la Web. A continuación se explican 2 estrategias que pueden desencadenar de lo que en primera instancia se conoce como búsqueda informacional.

GOAL		SET 1	SET 2	SET 3
	directed	2.70%	3.30%	7.30%
	undirected	31.30%	26.50%	22.70%
	advice	2.00%	2.70%	5.00%
	locate	24.30%	25.90%	24.40%
	list	2.70%	2.90%	2.10%
informational total		**63.00%**	**61.30%**	**61.50%**
	download	4.30%	4.30%	5.60%
	entertain	4.00%	8.20%	5.80%
	interact	5.70%	4.30%	6.00%
	obtain	7.70%	10.30%	7.70%
resource total		**21.70%**	**27.00%**	**25.00%**
navigational		**15.30%**	**11.70%**	**13.50%**

Figura 2.1: Resultado de clasificar las consultas según los objetivos de búsqueda

2.3 Estrategias para recuperar información de la Web

A lo largo de los años los usuarios se han acostumbrado a recuperar información de la Web, y para ello desarrollaron distintas estrategias, las cuales podemos resumir en:

- Recuperacion de informacion
- Búsqueda exploratoria

Marchionini [4] hace una distinción entre estas 2 estrategias de búsqueda, como se describe a continuación:

La recuperación de información es similar a la recuperación de datos o a la respuesta simple a preguntas, y se satisfacen con información breve y discreta como pueden ser números, fechas, nombres o sitios Web.

Respecto a las búsquedas exploratorias, estas se dividen en tareas de aprendizaje y tareas de investigación. Las primeras requieren más de un simple par pregunta-respuesta y lleva al usuario a pasar tiempo escaneando y leyendo multiples items de información y sintetizando contenido para generar un nuevo aprendizaje.

Por otra parte las tareas de investigación se refieren a procesos más largos que involucran múltiples interacciones que toman lugar quizá en periodos de tiempo más largos y pueden retornar resultados que son evaluados críticamente antes de ser integrados como un conocimiento personal o profesional. Algunos tipos de búsqueda de investigación se ocupan de encontrar toda o una gran parte de la información relevante disponible como puede ser la investigación académica.

Además Marchionini [4] encontró que el proceso de búsqueda de información consiste en una serie de búsquedas interconectadas pero diversas. También se encontró que los resultados de una búsqueda con un objetivo tienden a disparar nuevos objetivos y por lo tanto encauzar la búsqueda en nuevas direcciones. El contexto del problema y las búsquedas previas son acarreadas de una búsqueda a la siguiente. Por último el principal valor de la búsqueda reside en el aprendizaje acumulado y la adquisición de información que ocurre durante el proceso de búsqueda y no en el conjunto de resultados finales.

Hasta aquí se han clasificado las búsquedas según el objetivo del usuario y se han explicado las diferentes estrategias para satisfacer dicho objetivo. Para entender cómo se llevan a cabo estas estrategias es importante diferenciar el concepto de navegación o browsing, con el de búsqueda simple, como se explica en la siguiente subsección.

2.4 Navegación Vs Búsquedas

Cabe destacar que no todas las búsquedas empiezan con tipear una palabra clave en un input de búsqueda. Muchos sitios Web y algunos motores de búsqueda permiten a los usuarios examinar una estructura de información de algún tipo para seleccionar un punto de comienzo para la búsqueda. Los términos navegación o *browsing* son usados para referirse a una estrategia donde el usuario busca una estructura de información, y se mueve entre las vistas de la información disponible, en una secuencia de escanear y seleccionar operaciones. Los usuarios suelen preferir el *browsing* por sobre una búsqueda a partir de palabras clave cuando la estructura de información (por ejemplo *links* dentro de un sitio Web) están emparejadas a su necesidad de información.

Vicki O'Day y Robin Jeffries [5] explican como regla general que, los usuarios no se oponen tanto a seguir muchos enlaces, sino que se oponen a tener que seguir enlaces que no parecen acercarlos a su objetivo. En relación con esto último, Yates y Neto [6] argumentan que los usuarios pueden buscar información en la Web ya sea navegando documentos y/o usando herramientas especializadas de recuperación de información (los ya mencionados motores de búsqueda). Como se ve en la Fig. 2.2, aunque estas tareas sean diferentes, ambas están interconectadas. Por un lado el objetivo final de los motores de búsqueda es para direccionar a los usuarios a páginas Web que contienen la información buscada. Por otro lado, a partir de la navegación de estos documentos los usuarios pueden encontrar información complementaria que no está disponible a través de la navegación simple. Todo este proceso se conoce como tarea de búsqueda Web y será analizado en la siguiente sección.

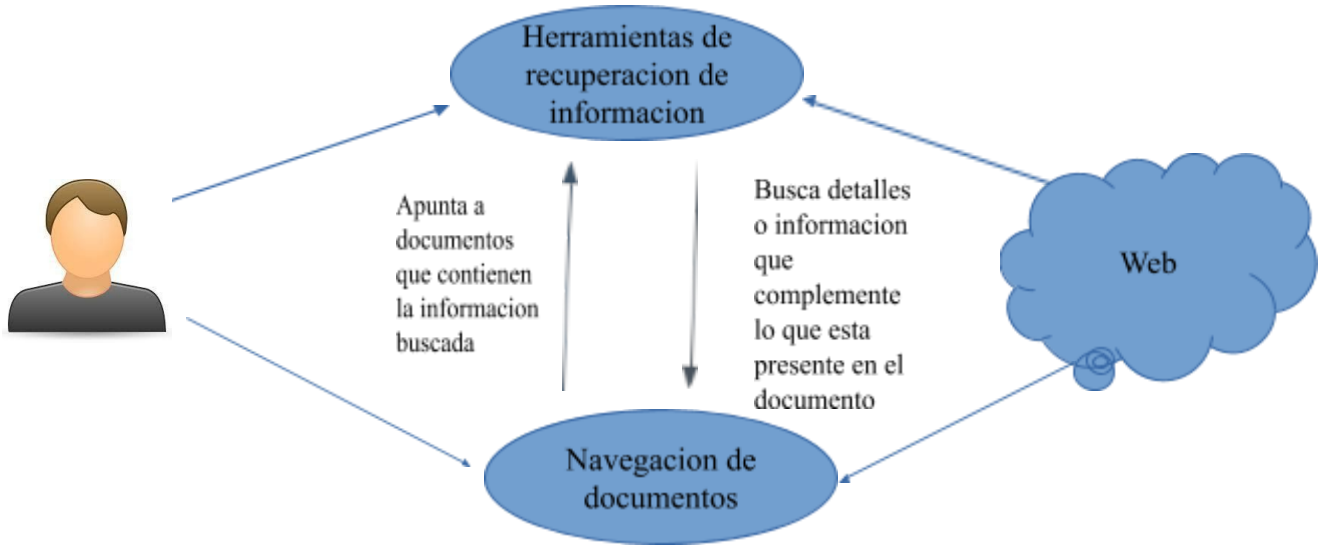


Figura 2.2: Tarea de búsqueda Web

2.5 Análisis de las tareas de búsqueda Web

En esta sección sólo se analizan las tareas de búsqueda desde un nivel alto de abstracción que aun no implica la implementación de ninguna herramienta (como si se verá en próximos capítulos). Se asume que el usuario puede realizar búsquedas ya sea usando una herramienta dedicada de recuperacion de informacion asi como tambien disparando una búsqueda directamente desde un documento Web. Winkler et al [7] resumen una búsqueda como un conjunto de subtareas:

1. El usuario formula una consulta.
2. El sistema procesa la consulta y muestra los resultados.
3. El usuario puede refinar o filtrar los resultados.
4. El usuario selecciona la entrada apropiada que corresponde con la información buscada.
5. En cualquier momento el usuario puede finalizar la búsqueda.

Mediante HÁMSTER (una notación de modelos de tareas) se ilustra en la Fig. 2.3 la tarea de búsqueda mediante una estructura de árbol para alcanzar el detalle requerido por este análisis:

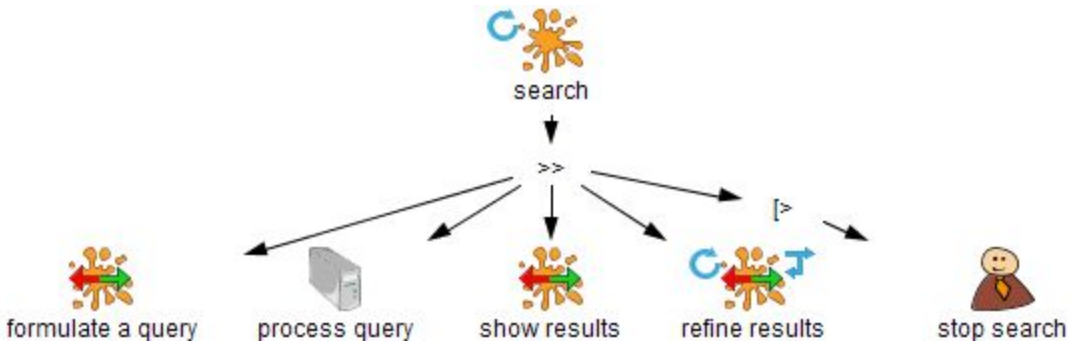


Figura 2.3: Tarea de búsqueda Web

Nótese que la tarea del nivel más alto *search* (búsqueda) está decorada con un símbolo azul a su izquierda que indica que la tarea es iterativa, es decir que puede ser repetida indefinidamente por el usuario. Las subtareas están conectadas por el operador >> que determina las secuencia de la ejecución de la tarea. El operador [> asociado a la tarea *stop search* (detener búsqueda) indica que, cuando la tarea es realizada, interrumpe la secuencia de otras subtareas. Además debe notarse que iconos específicos son usados para indicar la tipología de la tarea, por ejemplo, *process a query (*procesar una consulta*)* es típicamente automatizada por el sistema, *stop search* es una tarea de usuario, que requiere una simple decisión del mismo, y por último tareas como *formulate a query* (formular una consulta), *show results* (mostrar resultados) y *refine results* (refinar resultados)**,** requieren interacción del usuario con el sistema, por lo tanto se llaman interactivas. El símbolo junto a *refine results* es usado para indicar que la tarea es opcional.

La Fig. 2.4 muestra un nivel más allá de la descomposición de tareas con alternativas para realizar algunas tareas. Como el caso de *formulate query* (formular consulta), se pueden identificar subtareas adicionales incluyendo *provide keywords* (proveer palabras clave) y

select a database (seleccionar base de datos). El operador |=| indica que estas tareas pueden ser realizadas en cualquier orden. Se puede observar que la tarea *provide keywords* se descompone en subtareas alternativas: *type keyword (*tipear palabra clave) y *select word* (seleccionar palabra). La elección entre tareas es indicada por el operador [ ]. Algunas de las tareas alternativas consideran el caso de la ejecución manual vs la automática. Es el caso de *select database* que puede realizarse solicitando al usuario que explícitamente identifique una base de datos (*identify database*). La alternativa automática asume que el sistema usa una base de datos predefinida (*predefined database*) y el usuario no puede cambiarla.

La siguiente tarea es *create the display* que se muestra como una simple tarea de salida en la taxonomía de HAMSTER. La creación del display ofrece muchas alternativas para realizar la tarea *interact with display***.** Básicamente, desde este punto, las tareas de usuarios dependen de la ubicación del display y el número de entradas en el conjunto de resultados. Para la ubicación, los usuarios pueden usar una nueva ventana (*use new window*) o la actual (*use the current window*). Para el número o entradas, un usuario puede tanto obtener todas de una vez (*get all at once*) o dividir los resultados en subconjuntos que pueden ser navegados por el usuario (*browse subset*).

Los usuarios pueden adoptar 2 estrategias para refinar resultados (*refine results*): aplicar mecanismo de filtrado (*filtering***)** o realizar una nueva busqueda (*new search*). Una tarea nueva de búsqueda sigue el mismo patrón de una *search task* completa, lo que significa que esta última puede ser recursiva. Para indicar que una subtarea corresponde a un patrón de una tarea conocida, HAMSTER provee un tipo particular de construcción llamado *copy task*, el cual es claramente indicado en la Fig. 2.4 como decoración alrededor de la tarea *new search*. El nivel de descomposición de tareas en HÁMSTER es arbitrario, pero se asume que los detalles provistos en la Fig. 2.4 son suficientes para ilustrar este análisis.

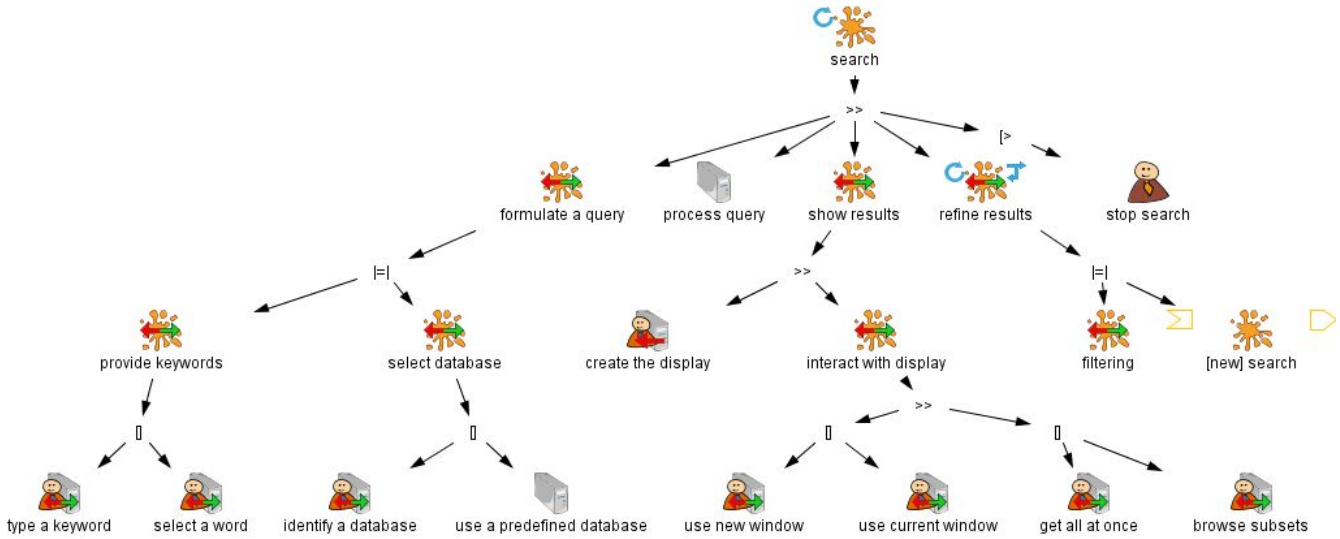


Figura 2.4: Tarea de búsqueda Web (mayor nivel de descomposición)

2.6 Diferencia entre tareas de búsqueda primarias y tareas auxiliares

Winkler [7] también afirma que observando el comportamiento del usuario se puede llegar a la conclusión que los mismos suelen tener muchas búsquedas corriendo en paralelo (o ejecutándose en intervalos muy cortos de tiempo). Mientras algunas búsquedas pueden referirse a objetivos completamente desunidos (por ejemplo, buscar un restaurant en La Plata para esta noche y planear un viaje para el fin de semana), otras búsquedas forman parte de un objetivo más general del usuario (por ejemplo buscar un hotel, luego un vuelo y luego alquilar un auto, siendo el objetivo general planificar un viaje). También es interesante notar que muchas búsquedas paralelas no corresponden a un objetivo primario del usuario pero son realizadas para obtener información que pueda acercarnos a lograr una tarea previa (por ejemplo buscar la cotización para calcular precios en una moneda extranjera mientras se busca un hotel). La Fig. 2.5 ilustra las diferencias entre una tarea de búsqueda de acuerdo al objetivo del usuario, que puede ser formalizada de la siguiente manera:

- Tarea de búsqueda primaria: Corresponde a una necesidad primaria de información. Idealmente, las búsquedas primarias abarcan un ciclo simple de interacción

pregunta-respuesta. SIn embargo, si los resultados no satisfacen al usuario, puede reformular los términos usados en la consulta y realizar una nueva búsqueda. Nótese que los usuarios pueden realizar tantas consultas como quieran pero cada consulta será tratada como única por el sistema. En consecuencia, las entradas en los resultados provistos por diferentes buscadores pueden ser muy diferentes según las palabras usadas

- Tarea de búsqueda auxiliar: Su objetivo es proveer detalles bajo demanda de los resultados que se muestran en pantalla en un momento dado. Las búsquedas auxiliares dependen de los resultados obtenidos de una búsqueda previa y/o información disponible en una página Web. Idealmente, una vez que los usuarios encuentran las respuestas deberían volver al contexto de la tarea que estaban realizando en primera instancia, antes de lanzar la búsqueda auxiliar.

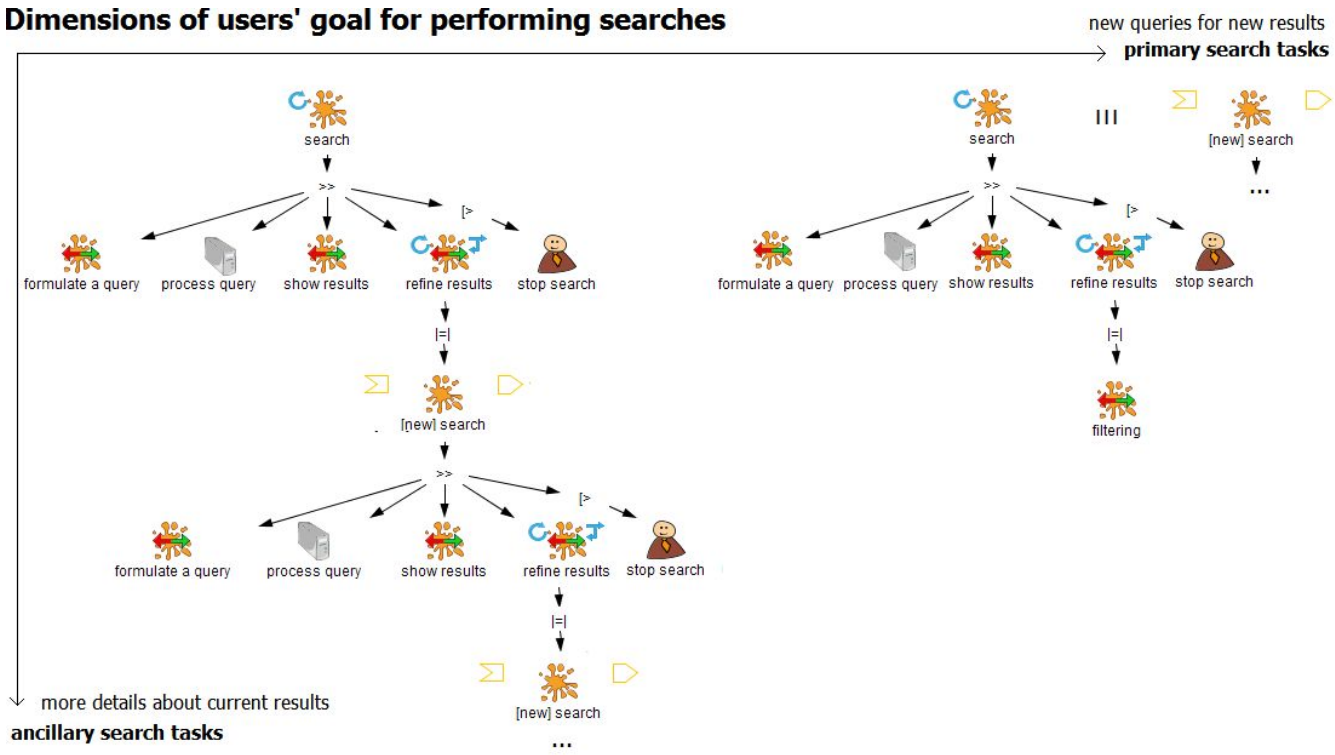


Figura 2.5: Dimensiones del objetivo del usuario para realizar tareas de búsqueda (separando búsquedas auxiliares anidadas de las búsquedas primarias)

Es importante remarcar que las tareas de búsqueda web primarias son típicamente tratadas como tareas separadas por el sistema, por lo tanto pueden ocurrir en paralelo. Sin embargo, las búsquedas auxiliares dependen en gran medida de los contenidos existentes para los que está destinado a proporcionar más detalles. Además las búsquedas auxiliares llevan a consultas anidadas que crear una seguidilla de búsquedas realizadas por el usuario mientras que las búsquedas primarias son tratadas independientemente por el sistema. Ambos tipos de tareas pueden combinarse de acuerdo a la necesidad actual de información que tenga el usuario. De esta manera, una tarea de búsqueda primaria puede correr en paralelo con otras búsquedas que requieren descomposición en búsquedas auxiliares.

Si bien se explico la diferencia entre ambos tipos de búsqueda no se hizo mención de los tipos de motores de búsquedas a los que el usuario consulta. Existen motores de búsqueda genéricos como Google y otros de dominio específico como IMDB (películas). Los primeros cubren un amplio alcance y su objetivo es proporcionar entradas a las fuentes de datos/sitios Web que contienen la información que los usuarios están buscando. En contraste, McCallum et al [8] explican que los motores de búsqueda de dominio específico se centran en datos con un alcance limitado pero más precisos para aquellos usuarios que desean ser provistos de algunos detalles especializados sobre objetos en un dominio concreto. En efecto, la mayoría de las tareas de búsquedas auxiliares se desencadenan porque el usuario necesita información complementaria, la cual frecuentemente es de dominio específico. Morris et al [9] demuestran que los usuarios navegan sobre diferentes tipos de páginas y acostumbran a volver a visitar páginas del mismo tipo y estructura. Esto sugiere que a partir de objetos estructurados se puede tomar las ventajas del contenido existente y del comportamiento ya disponible en la Web, ya que comparten el mismo dominio. Y esta última idea, es además una de las bases del trabajo propuesto.

Capítulo 3: Trabajos relacionados

3.1 Introducción

El propósito de este capítulo es analizar y hacer un relevamiento de los avances que hubo hasta el momento en el campo de las búsquedas Web así como también en lo relativo a la visualización de la información. Esto incluye tendencias en cuanto a tecnologías, enfoques novedosos propuestos por navegadores Web como es el caso de Mozilla Firefox, *Web extensions* desarrolladas tanto por particulares como por grandes corporaciones como Google y, por último, investigaciones de especialistas en la temática. El objetivo es entender cómo se fueron adaptando a las necesidades de los usuarios y mostrar las mejoras alcanzadas hasta el momento, principalmente en lo que respecta a cómo mejorar la experiencia del usuario final al realizar búsquedas y ,posteriormente, al visualizar los resultados obtenidos de dichas búsquedas.

En resumen, la primer sección se focaliza en los avances que hubo en el área de la visualización de la información, los cuales abarcan un proceso que comienza en la extracción de datos en la Web y finaliza en la renderización de los resultados. Se analiza en detalle el proceso, así como también qué tecnologías entran en juego y cómo se adaptan a la estructura del sitio Web que contiene la información. En la segunda sección se realiza un relevamiento y se analizan distintos enfoques vigentes actualmente, como *Search Bar* (un enfoque integrado a Mozilla Firefox) y algunas *Web extensions* relacionadas con las búsquedas en la Web o que presentan algún enfoque novedoso respecto a la visualización de resultados. Finalmente se presenta un enfoque de investigadores del LIFIA, llamado WOA, el cual en es predecesor, sobre todo en un aspecto particular, al enfoque propuesto en este trabajo.

De cada enfoque se intenta destacar las mejoras particulares que aportan, para luego en el siguiente capítulo mostrar cómo se integran muchos de esos aspectos en una propuesta superadora como la que se presenta en el actual trabajo.

3.2 Visualización de la información

Actualmente, cada vez más sitios Web incrustan técnicas de visualización de la información para presentar sus datos en contexto. Sin embargo , esta práctica no es extendida y la mayoría de los sitios Web todavía siguen mostrando datos semi-estructurados. Las operaciones requeridas para visualizar datos semi-estructurados obtenidos de la Web no son sencillas, y requieren un tratamiento adecuado antes de que las técnicas puedan ser aplicadas. Price et al [11] describen que estas técnicas pueden mejorar tanto las habilidades cognitivas del usuario, así como también el rendimiento de estos al realizar tareas ya que alivian la memoria de trabajo y mejoran la precisión de las decisiones.

En esta sección se apunta a conocer mejor el proceso de visualización de la información, el cual permite transformar datos semi-estructurados en representaciones gráficas. Estos avances resultan útiles para comprender cómo se resuelve en el enfoque presentado en este trabajo el problema de extracción de datos a partir de diferentes sitios Web, luego de que haya sido disparada la búsqueda. Además, una vez obtenidos los datos, cómo se visualizan dentro del contexto desde el cual se disparó la búsqueda.

3.2.1 Proceso de visualización de la información

Muchos aspectos del diseño de la visualización son manejados según el tipo de datos que estamos observando. Para que estos puedan ser entendidos por los usuarios, las fuentes de datos son a menudo transformadas a través de un proceso que transforma datos crudos en una representación gráfica que se amolda a la pantalla del usuario como investigadores del LIFIA explican en [12] y se detalla a continuación:

El proceso es mostrado en la Fig. 3.1, en el mismo incluye 4 operaciones:

1. Adquisición de datos (**data acquisition**)
2. Filtrado (**filtering**)
3. Mapeo visual (**visual mapping**)
4. Renderización (**rendering**)

Nótese que la interacción del usuario (representada en la imagen por las líneas discontinuas) puede afectar todas las operaciones a lo largo del proceso.

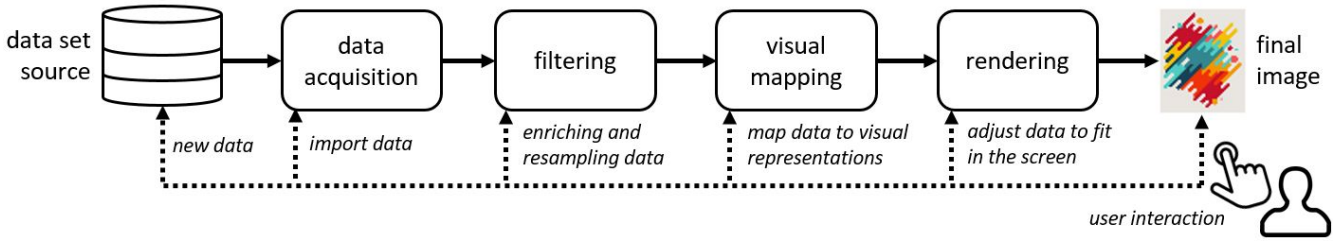


Figura 3.1 Pipeline de la visualización de la información, adaptado de Munzner (2014) [12]

Uno de los problemas a ser resueltos es la adquisición de datos, ya que el conjunto de datos puede estar disponible en varios formatos y la mayoría de las técnicas de visualización son diseñadas específicamente para manejar un tipo de dato particular (como tablas o listas). En la mayoría de los casos si el conjunto de datos no está en el formato de la herramienta, resulta difícil obtener los beneficios de las técnicas de visualización. En la siguiente sección del capítulo se analizan conceptos que explican cómo adquirir datos a partir de sitios Web.

Cuando un conjunto de datos es cargado en la herramienta, puede contener información que no es relevante para resolver el problema que se está buscando. En ese momento entra en juego el filtrado para remover ruido, arreglar atributos (por ej. cuyo codificador de caracteres es erróneo), y para enriquecer conjuntos de datos (por ej. agregar etiquetas que falten).

El mapeo visual está en el núcleo del diseño de las técnicas de visualización de la información. Permite asociar atributos (como por ejemplo género y edad) a variables visuales (como forma, color, tamaño o textura). Este mapeo puede ser hardcodeado adaptable por parte de los usuarios. Un buen ejemplo es el de uVis Studio [13], el cual permite a los desarrolladores componer visualizaciones mediante drag-and-drop, luego enlazando controles a los datos y visualizando resultados obteniendo un feedback inmediato. Este tipo de técnicas como el binding dinámico provee flexibilidad e interactividad para usuarios que quieran personalizar las vistas acorde a sus necesidades.

Las operaciones de renderización definen como las técnicas de visualización son mostradas a los usuarios. En este punto, las herramientas deben realizar transformaciones geométricas para hacer que los datos se adapten a la pantalla. La renderización además define si la visualización va a ser mostrada en una aplicación estándar o en un elemento que se puede integrar en otro contexto de uso.

El proceso en la Fig. 3.1, es parte de todas las herramientas de visualización, sin importar la tecnología utilizada para la implementación. En lo que respecta a la tecnología Web, existen muchas librerías basadas en *JavaScript* que manipulan el DOM y CSS para construir técnicas de visualización que puedan ser mostradas dentro del browser. Por ejemplo Datatables para visualizar datos en forma de tabla o ChartJs, una librería que ofrece una gran colección de visualizaciones interactivas, útiles para visualizar contenido en forma de gráficos estadísticos. El uso común de estas librerías es para alimentar las visualizaciones con datos que vienen de cierta API o datos fijos especificados por desarrolladores Web. En la mayoría de los casos, la creación y el uso de técnicas de visualización, siguen siendo algo muy técnico que requiere habilidades de programación.

3.2.2 Extracción de datos en la Web

Como se mencionó anteriormente la primera operación en el proceso de visualización está relacionada con la extracción o adquisición de datos. En esta sección se introducen algunos

conceptos necesarios que ayudarán a entender mejor cómo se realiza la misma en enfoque del actual trabajo, ya que los servicios de búsqueda que definen los usuarios con la herramienta presentada en el Capítulo 5, deben extraer los datos de alguna forma, y a partir de esta extracción poder retornar resultados cuando se dispara una consulta (resultados obtenidos de un contenido en la Web).

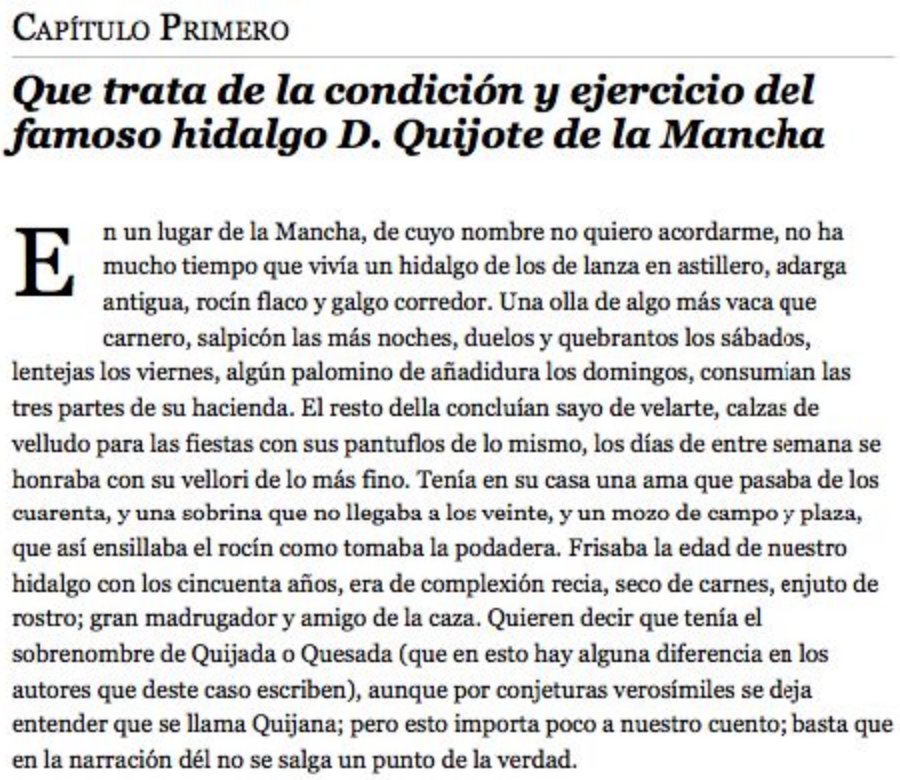
E. Ferrara et al [14] explican que una de las características más explotadas en la extracción de datos en la Web es la naturaleza semi-estructurada de las mismas. Como punto de partida en la extracción de datos cabe aclarar la diferencia entre datos estructurados, semi-estructurados y no estructurados:

- Datos estructurados: Tienen perfectamente definida la longitud, el formato y el tamaño de sus datos. Se almacenan en formato tabla, hojas de cálculo o bases de datos. En la Fig. 3.2 podemos observar un ejemplo.

	nombre	color	edad	altura	peso	puntuacion
1:	Paco	Rojo	24	182	74.8	83
2:	Juan	Green	30	170	70.1	500
3:	Andres	Amarillo	41	169	60.0	20
4:	Natalia	Green	22	183	75.0	865
5:	Vanesa	Verde	31	178	83.9	221
6:	Miriam	Rojo	35	172	76.2	413
7:	Juan	Amarillo	22	164	68.0	902

Figura 3.2: Datos estructurados

- Datos no estructurados: Se caracterizan por no tener un formato específico. Se almacenan en múltiples formatos como pueden ser PDF o Word, e-mails, ficheros multimedia de imagen, audio, video, etc (Fig. 3.3)



CAPÍTULO PRIMERO

Que trata de la condición y ejercicio del famoso hidalgo D. Quijote de la Mancha

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda. El resto della concluían sayo de velarte, calzas de velludo para las fiestas con sus pantuflos de lo mismo, los días de entre semana se honraba con su vellori de lo más fino. Tenía en su casa una ama que pasaba de los cuarenta, y una sobrina que no llegaba a los veinte, y un mozo de campo y plaza, que así ensillaba el rocín como tomaba la podadera. Frisaba la edad de nuestro hidalgo con los cincuenta años, era de complexión recia, seco de carnes, enjuto de rostro; gran madrugador y amigo de la caza. Quieren decir que tenía el sobrenombre de Quijada o Quesada (que en esto hay alguna diferencia en los autores que deste caso escriben), aunque por conjeturas verosímiles se deja entender que se llama Quijana; pero esto importa poco a nuestro cuento; basta que en la narración dél no se salga un punto de la verdad.

Figura 3.3: Datos no estructurados

- Datos semi-estructurados: Son una combinación de los 2 anteriores, ya que no representan una estructura perfectamente definida como los estructurados pero sí presentan una organización definida en sus metadatos donde describen los objetos y sus relaciones y que en algunos casos están aceptados como convención. Por ejemplo HTML, XML o JSON (Fig 3.4).

```
{
    "marcadores": [
      {
        "latitude": 40.416875,
        "longitude": -3.703308,
        "city": "Madrid",
        "description": "Puerta del Sol"
      },
      {
        "latitude": 40.417438,
        "longitude": -3.693363,
        "city": "Madrid",
        "description": "Paseo del Prado"
      },
      {
        "latitude": 40.407015,
        "longitude": -3.691163,
        "city": "Madrid",
        "description": "Estación de
          Atocha"
      }
    ]
}
```


Figura 3.4: Datos semi estructurados

En lo que respecta a las páginas Web las mismas pueden ser representadas como árboles ordenados y etiquetados, donde dichas etiquetas representan tags HTML, y la jerarquía del árbol representa los diferentes niveles de anidamiento de los elementos que constituyen la página Web. La representación de esta última usando un árbol etiquetado y ordenado con una raíz se conoce como DOM (*Document Object Model*). La idea general de este es que las paginas Web HTML sean representadas por significados de texto plano que contengan tags HTML, es decir palabras claves particulares definidas en el lenguaje de marcas de hipertexto, las cuales puedan ser interpretadas por el browser para representar los elementos específicos de una página Web (botones, imágenes, links, etc). Los tags HTML pueden ser anidados dentro de otros, formando una estructura jerárquica. Esta jerarquización es capturada en el DOM por parte del árbol del documento cuyos nodos representan tags

HTML. El árbol del documento (también conocido como DOM *tree*) ha sido exitosamente explotado para propósitos de extracción o adquisición de datos en la Web a partir de diferentes técnicas como la que explicaremos a continuación y que es utilizada en el actual trabajo.

3.2.3 Identificación de elementos en el árbol del documento: Xpath

Una de las principales ventajas de la adopción de DOM para el lenguaje HTML es la posibilidad de explotar algunas herramientas típicas de lenguajes XML (HTML es, en efecto, un dialecto de XML). En particular, el XML *path language* (conocido como XPath) provee mediante una poderosa sintaxis la posibilidad de identificar elementos específicos de un documento XML (o de igual manera en páginas Web HTML) de una forma simple. XPath ha sido definido por la *World Wide Web Consortium*, al igual que el DOM.

Aunque describir la sintaxis de XPath no es el objetivo de la sección, en la Fig. 2.10 se provee un ejemplo que explica cómo se puede usar XPath para seleccionar elementos de una página Web. Existen 2 maneras posibles de usar XPath: i) Para identificar un elemento simple en el árbol del documento o ii) Para identificar múltiples ocurrencias del mismo elemento. En el caso formal ilustrado en la Fig. 3.5(A), el XPath definido identifica múltiples instancias del mismo tipo de elemento (una celda de una tabla) compartiendo la misma ubicación jerárquica.

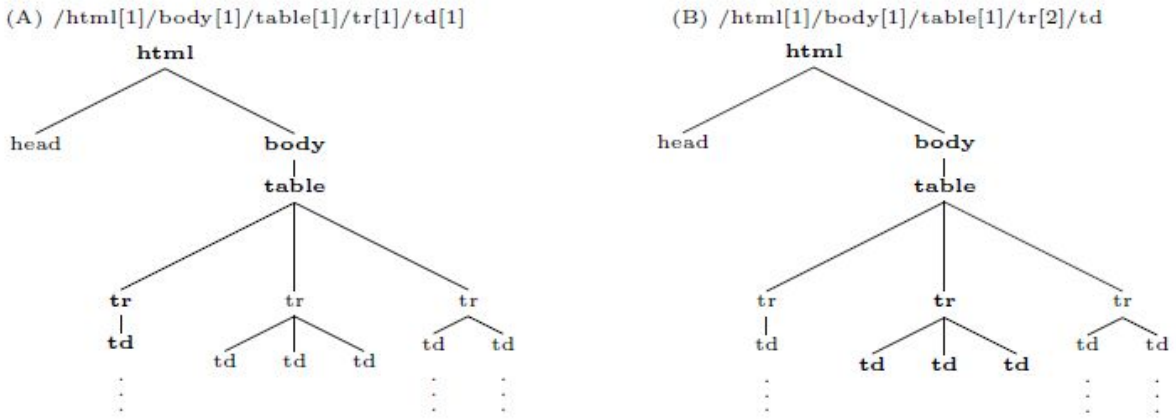


Figura 3.5: Ejemplo de XPath(s) en el árbol del documento, seleccionando 1 (A) o múltiples ítems (B)

Para el propósito de la extracción de datos de la Web, la posibilidad de explotar una herramienta tan poderosa ha sido muy importante: la adopción de XPath como una herramienta para seleccionar elementos en la Web ha sido largamente explotada. La mayor debilidad de XPath está relacionada con la ausencia de flexibilidad: cada expresión XPath está estrictamente relacionada a la estructura de la página Web en la cual ha sido definida. Sin embargo, esta limitación ha sido parcialmente mitigada introduciendo las *relative path expressions*, en las últimas versiones. En general, incluso los cambios menores en la estructura de una página Web pueden corromper el correcto funcionamiento de una expresión XPath definida en una versión previa de la página.

Para clarificar mejor este concepto, considere páginas Web generadas por un script (por ej. pensar en la información sobre un libro en un sitio Web *e-commerce*). Luego asuma que dicho script sufre un cambio: podemos esperar que la estructura del árbol del HTML generado por el script cambie en concordancia. Para mantener el proceso de extracción de datos en la Web funcional, se debería actualizar la expresión cada vez que un cambio ocurre en el modelo de la página subyacente, y dicha operación requerirá un compromiso humano muy grande y por lo tanto su costo podría ser demasiado alto. A raíz de esto algunos autores [15][16] introdujeron el concepto de wrappers robustos.

Dave et al [17] explican que la similitud estructural de las páginas Web permite una posible extracción de información usando simples reglas, denominadas wrappers. Una vez que los wrappers son aprendidos por el sitio Web podrá extraerse la información actualizada de una determinada página Web. El problema es que los wrappers dependen en gran medida de la estructura de las páginas Web para extraer datos, y como explicamos anteriormente la misma puede cambiar frecuentemente generando inconsistencias en el wrapper.

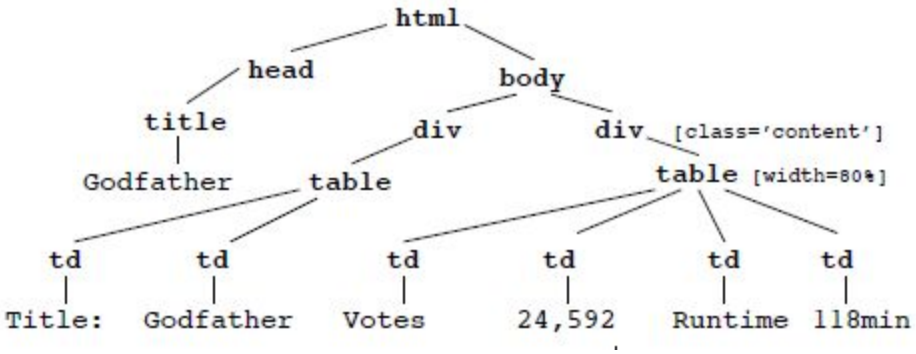


Figura 3.6: Una página Web HTML

Para ilustrar en la Fig. 3.6 se representa un árbol de un documento XML de una pagina de peliculas. Si se quisiera extraer el número de votos se podría usar un XPath como el siguiente:

$$W_1 = /html/body/div[2]/table/td[2]/text()$$

Sin embargo, existirían muchos pequeños cambios que pueden romper este wrapper. Por ejemplo si el primer div es borrado o mergeado con el segundo div, una nueva tabla o tr es añadido bajo el segundo div, el orden de Votes y Runtime es invertido, una nueva fuente de elementos es añadida, y así sucesivamente.

Estos problemas han sido ha sido ampliamente reconocidos en varios estudios empíricos. Se ha observado que un wrapper que no es robusto tiene un promedio de vida útil de 2 meses. Por lo tanto para manejar este problema, hay una necesidad de manualmente volver a etiquetar (es decir volver a proveer la ubicación en la página que contenga la información de interés) las páginas web con bastante frecuencia e indicar nuevamente al wrapper cómo usar las nuevas etiquetas. Este proceso resulta muy costoso.

Sin embargo existen trabajos sobre la robustez en los wrappers, como Myllymaki y Jackson [18] que observaron que ciertos wrappers son más robustos que otros, y en la práctica pueden tener significativamente menos posibilidad de romperse. Por ejemplo los siguientes XPaths pueden ser usados como alternativa al W_1 para extraer el número de votos en la Fig. 3.6:

$$W_2 = \text{//div[@class='content']/*/td[2]/text()}$$

$$W_3 = \text{//table[@width='80\%']/td[2]/text()}$$

Intuitivamente, se puede ver que estos wrappers explotan mayor informacion "local" que W_1 y son inmunes a algunos cambios que puedan afectar W_1. Estos autores construyen wrappers robustos manualmente y dejan abierta la posibilidad de aprender dichas reglas automáticamente. El primer framework formal que captura la noción de robustez fue propuesto recientemente por Dalvi [19] mediante la definición de un modelo que captura como las páginas Web evolucionan a lo largo del tiempo, y el modelo puede ser usado para evaluar la robustez de los wrappers. Sin embargo a pesar de existir técnicas que permiten elegir entre un conjunto de XPaths alternativos según cuán robustos sean, el problema sigue abierto y es de importancia en la mejora del trabajo propuesto, principalmente en lo que respecta a la extracción de datos de los servicios de búsqueda que defina el usuario final.

3.3 Web Scraping y Web Augmentation

Investigadores del LIFIA [12] sugieren que la visualización de información a partir de un conjunto de resultados en la Web puede ser mejorada con tecnologías como *Web Scraping* y *Web Augmentation*.

Web Scraping permite transformar datos no estructurados disponibles en la Web, típicamente en formato HTML, en datos estructurados que puedan ser analizados y almacenados en bases de datos centralizadas. Por ejemplo, MeatBrain [20] es una herramienta que extrae datos de sitios Web y, eventualmente, los agrega en una nueva página Web. También es muy común que los *scrappers* permitan a sus usuarios definir que parte de sitios Web extraer, mientras otros pueden hacerlo automáticamente.

Los Web *scrappers* son a menudo la base de otras aplicaciones, como motores de búsqueda y *Web* Augmentation. Es interesante notar que, aunque no todas las herramientas de Web Augmentation emplean Web Scraping, la mayoría de estas contiene alguna funcionalidad scrapper que es usada para parsear los DOMs de las páginas Web con el objetivo de materializar la aumentación.

Tradicionalmente el webmaster decide las reglas de personalización sobre el sitio Web. En contraste, dar lugar a la personalización implica empoderar terceras partes para personalizar sitios Web ya existentes.

Web Augmentation está dentro de la categoría general de técnicas de *transcoding*, es decir transformar contenido bajo demanda a otros formatos. En este caso el código a ser transformado generalmente es HTML (o en *runtime* es más preciso decir que es el árbol DOM). Anteriormente el *transcoding* tomaba lugar en un servidor dedicado o en un proxy que que se encargaba de interceptar la petición HTTP. Recientemente, este proceso de

transformación fue movido al cliente, donde el DOM es transformado bajo demanda como explica Bouvin [28]. Inicialmente estas técnicas eran utilizadas para mejorar la accesibilidad o el soporte de aplicaciones Web para varios dispositivos. En la actualidad muchos especialistas en *Web Augmentation* se han esforzado por abordar las demandas de los usuarios para conseguir maneras más sofisticadas de controlar la experiencia de Web, conduciendo a la denominada *Personal Web* [29]. En esta Web personal los usuarios tendrán la libertad de adaptar sus sitios Web de uso diario convirtiendo a *Web Augmentation* en la última técnica de personalización.

Díaz y Arellano [26] resumen que *Web augmentation* permite adaptar sitios Web de terceros para añadir nuevo contenido o funcionalidad y puede ser una alternativa adecuada para integrar técnicas de visualización en sitios Web que carezcan de características de visualización. Existen diferentes alternativas para lograr *Web Augmentation* tanto del lado del cliente como del servidor. Scripts de usuarios y *Web Extensions* son los 2 mecanismos más comunes para soportar Web Augmentation. Para Enero de 2015 y solo tomando Mozilla Firefox, mas de 14 billones de extensiones fueron descargadas, dejando en evidencia la vitalidad de este movimiento.

A diferencia de la arquitectura de software típica, los usuarios pueden a través de *Web Augmentation* personalizar algo que ya fue desarrollado y desplegado por otro, con el objetivo indicado por Chilton et al [27] de que los usuarios customizan la Web para mejorar la adaptabilidad a sus necesidades haciendo que las tareas sean menos repetitivas o la navegación menos aburrida. Además otra importante razón para personalizar la Web es eludir las restricciones legales o las intenciones originales del sitio.

Díaz y Arellano [26] revisan a partir de analizar 45 Web extensions, 3 aspectos principales de Web Augmentation:

- *Refactoring*: Reestructurar un código existente para mejorar atributos no funcionales (Ej. usabilidad).

- Customización: Aprovechar un código existente para adaptar los requerimientos de una minoría dentro del mismo dominio del host (ej. atajos para facilitar tareas repetitivas)

- *Modding*: Modificar un código existente para realizar una función que no es originalmente concebida por el host designer (ej. soportar tareas inter sitio o burlar restricciones legales).

Lo que diferencia el modding de la customización no es la implementación sino el fin. *Modding* no tiene como objetivo adaptar funcionalidad existente del host sino que introduce nuevas características. Por ejemplo, en Wikipedia hay restricciones legales que previenen alojar contenido con *copyright*. *Reflect* es un aumentador Web que refleja la detección de compuestos químicos dentro de artículos y, a continuación, incrusta dibujos y fórmulas (algunos sujetos a derechos de autor). Nótese que esta última es una funcionalidad que WIkipedia que nunca ha ofrecido, por lo tanto calificar a *Reflect* cómo customización resultaría engañoso. Por lo tanto, el término modding representa mejor este tipo de escenarios que complementan, o incluso entran en conflicto con la funcionalidad del host, pero no extienden la misma. Mediante esta última distinción entre "customización" y "*modding*" se intenta destacar un aspecto distintivo de la *Web Augmentation*, en el cual los usuarios ahora son empoderados, no solo para alinearse al host sino también para entrar en conflicto con este.

Estos escenarios muestran la oportunidad de Web Augmentation de convertirse en un facilitador de la visión descrita por Raman [32] donde la Web está evolucionando hacia una nube de aplicaciones customizables y datos, mediante las técnicas como las descriptas anteriormente de manera introductoria: *refactoring*, customización y *modding*.

3.4 Trabajos relacionados

3.4.1 Search bar Mozilla Firefox

El primer enfoque que se analiza en detalle y que es similar en una primer instancia, es el utilizado por Mozilla Firefox para realizar búsquedas desde el contexto actual en el cual se encuentra el usuario, a otros motores de búsqueda externos. Como se puede apreciar en la Fig.3.7 cuando el usuario empieza a escribir en la barra de búsqueda o en la barra de direcciones, aparecen iconos de otros motores de búsqueda que pueden ser consultados.

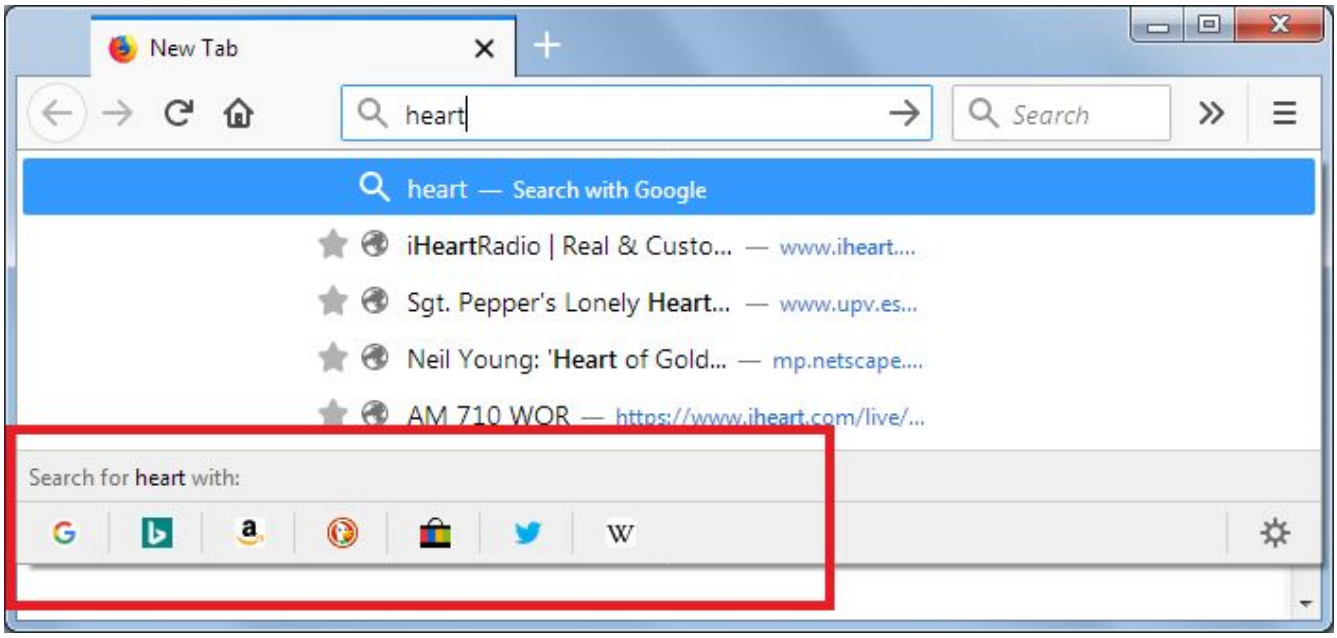


Figura 3.7: Search Bar Mozilla Firefox

Además de los buscadores por defecto, algunos proveedores ofrecen también complementos que permiten agregar muchos más motores de búsqueda, y a su vez estos pueden administrarse como se observa en la Fig. 3.8.

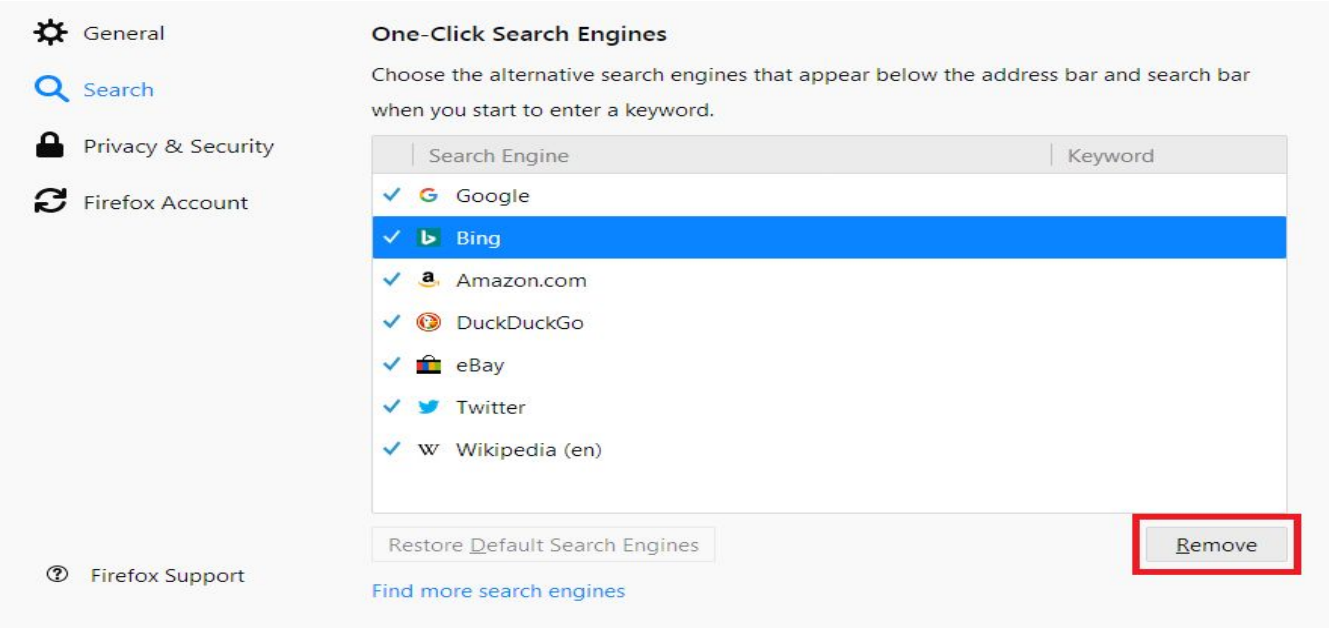


Figura 3.8: Administrar motores de búsquedas.

Los motores de búsqueda de determinadas páginas que cuenten con motor de búsqueda (no todas) son detectados automáticamente y pueden agregarse desde la barra de búsquedas de la siguiente forma (Fig. 3.9):

1. Visitar una página web que ofrezca un motor de búsqueda. Como ejemplo youtube.com.

2. Hacer clic en la lupa de la barra de búsqueda y luego hacer clic en Añadir "Búsqueda de videos en YouTube".

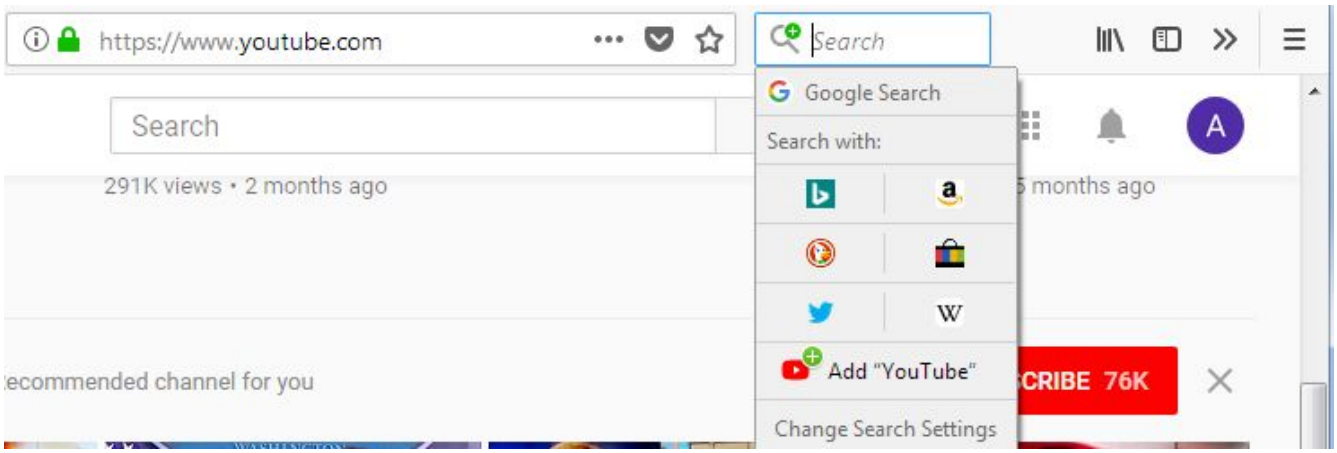


Figura 3.9 Añadir moto de búsqueda detectado.

Como resumen de esta implementación que ya viene integrada al instalar Mozilla Firefox, lo que se puede observar a simple vista es el ahorro de un paso en el caso de que el usuario requiera buscar en una página de dominio específico (por ej. un artículo en Wikipedia), este no deberá ingresar a Wikipedia y hacer la búsqueda o ingresar al motor de búsqueda por defecto (por ej. Google) y agregar al final de la búsqueda la palabra clave wikipedia como una suerte de filtrado.

Pero la experiencia del usuario concluye una vez disparada la búsqueda y abierta la nueva ventana o pestaña a otro sitio Web desde el cual se mostraran los resultados. Esta es una de las principales diferencias con el actual trabajo como se comprenderá a lo largo del mismo.

3.4.2 Web Extensions

Las *Web extensions* son pequeños programas de software que sirven para customizar la experiencia de *browsing* agregando nuevas características y funciones al navegador. Se construyen utilizando tecnologías Web ya conocidas como html, javascript y css. Son muchas las cosas que se pueden hacer mediante una *Web extension*, como por ejemplo:

- Agregar o quitar contenido de una página Web: las extensiones pueden ayudar a los usuarios a ver la Web de la manera que deseen, brindando la capacidad de acceder y actualizar tanto html como css de una página. Ejemplos de este tipo de extensiones son aquellas que bloquean los anuncios intrusivos de las páginas web (Fig.3.10), así como también aquellas que proporcionan una guía de viajes siempre que un país o ciudad se menciona en una página Web u otros tantos casos de extensiones existentes.

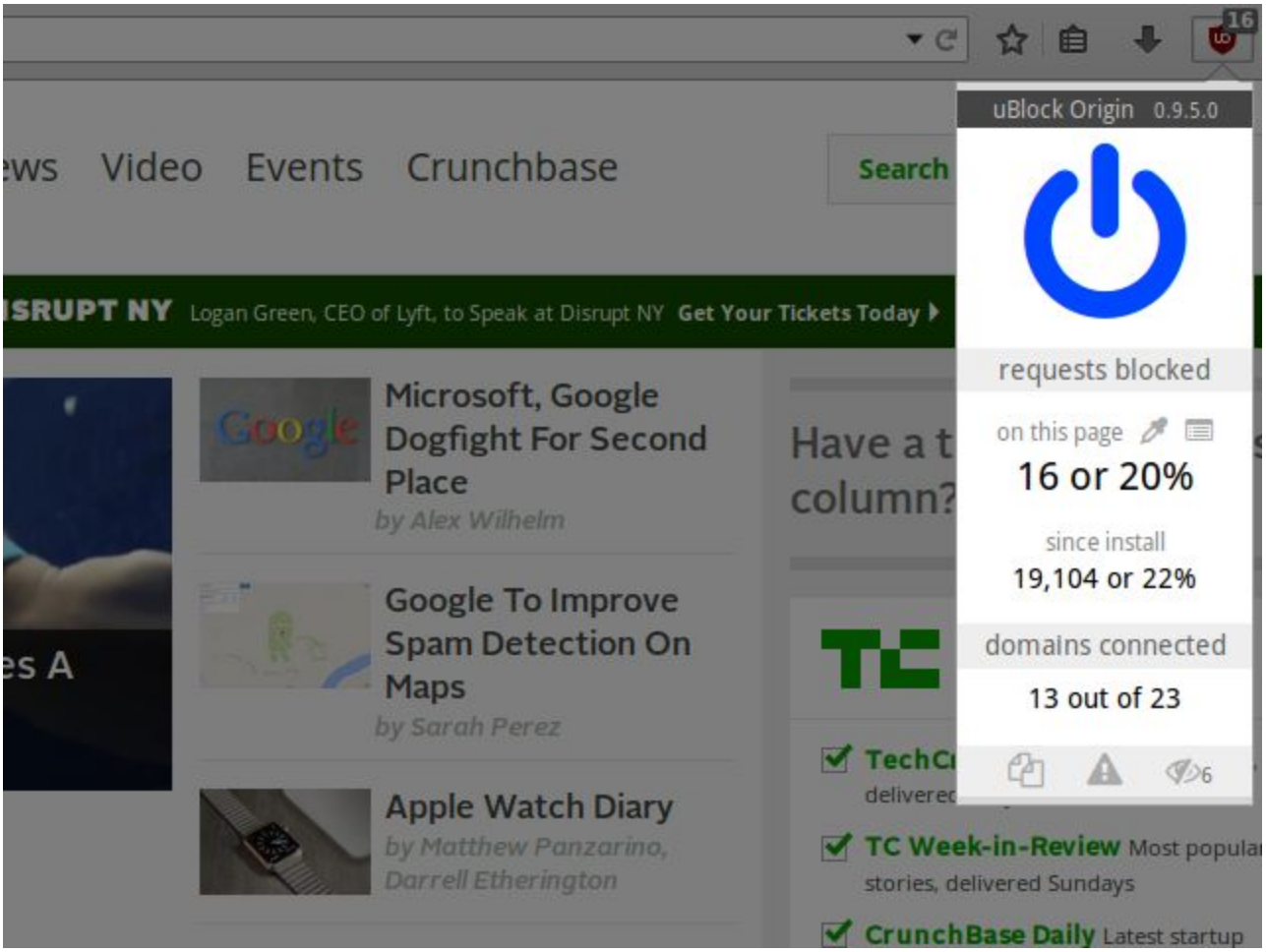


Figura 3.10:. Ejemplos de Web extension que bloquea anuncios en una página Web.

- Agregar contenido y nuevas funciones de navegación: Con opciones flexibles de interfaz de usuarios y la potencia de las APIs de Web extension se pueden añadir

fácilmente nuevas funcionalidades a un navegador, lo cual permite realzar casi cualquier característica o funcionalidad de un sitio Web (que no tiene porque ser propio). Por ejemplo, en la Fig.3.11 se generan imágenes de código QR a partir de URLs.

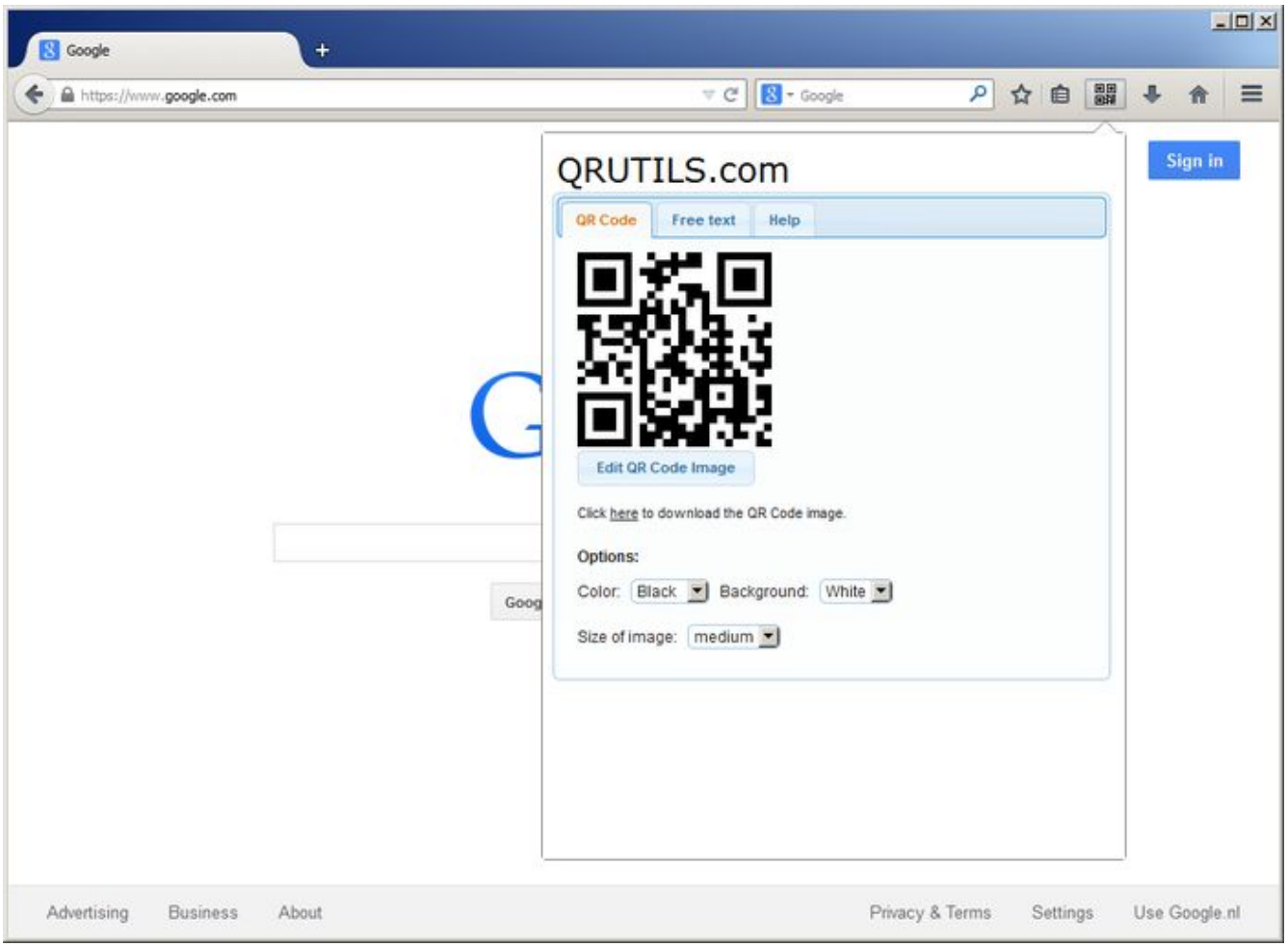


Figura 3.11:. Ejemplos de Web extension que genera códigos QR.

- Mejorar o complementar un sitio Web particular: como es el caso de Amazon Asisistant que permite descubrir y comparar productos relacionados dentro del sitio como se observa en la Fig 3.12.

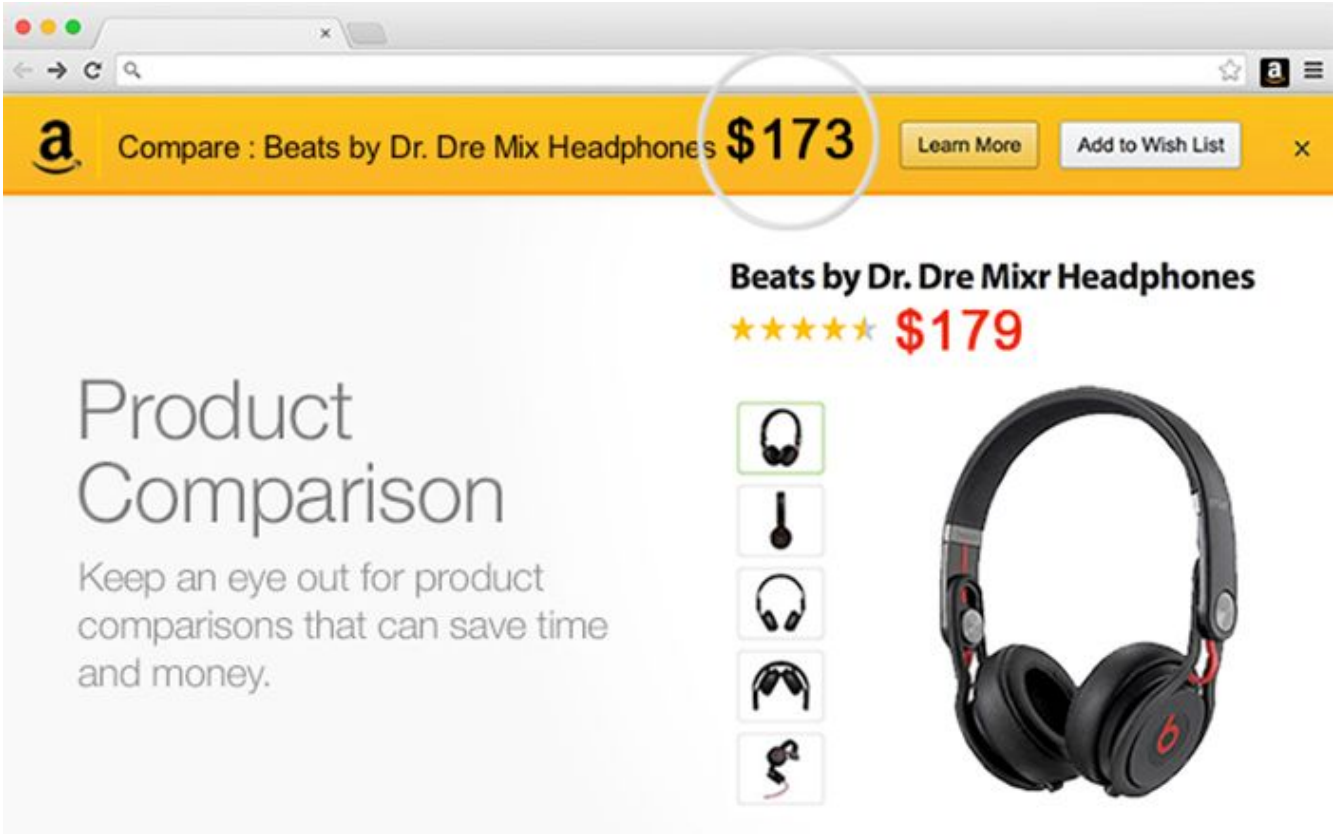


Figura 3.12:. Ejemplos de Web extension que compara y sugiere productos en Amazon.

Existen una gran cantidad de extensiones de distinto tipo, pero en este trabajo se hará foco en aquellas vinculadas al campo de las búsquedas Web y de la visualización de la información que son las más relacionadas al alcance del trabajo actual. Dentro de esta categoría de Web extensions, son muchas las opciones disponibles para instalar en los navegadores más elegidos por los usuarios. Por ejemplo en Google Chrome existe una categoría específica denominada “Herramientas de búsqueda” que agrupa este tipo de extensiones, y si bien el abanico es muy amplio, las más utilizadas actualmente son aquéllas que permiten disparar búsquedas, tanto de dominio específico como primarias, a distintos motores de sitios Web que el usuario frecuenta, y que puede configurar previamente.

Al hacer un relevamiento de estas extensiones, entre las más utilizadas se encuentran:

- Selection search: Cuenta con casi 18mil usuarios y soporte actualizado.
- Context Menu Search : Si bien cuenta con casi 85.000 usuarios no es actualizada desde 2014.
- Simple = Select + Search:Cuenta con mas de 25.000 usuarios y con soporte actualizado.
- Search Bar : Cuenta con casi 47mil usuarios y con soporte actualizado.

La cantidad de usuarios y además lo bien valoradas que están las extensiones mencionadas, evidencian el éxito y la necesidad imperiosa por parte de los usuarios de buscar alternativas para mejorar la experiencia cuando realizan búsquedas en la Web.

Estas extensiones son similares entre sí y también respecto al enfoque presentado anteriormente que se encuentra integrado al navegador Mozilla Firefox, aunque tienen algunas particularidades, por ejemplo en lo que respecta a cómo añadir buscadores a la herramienta o como disparar la búsqueda hacia estos motores previamente definidos por el usuario. A continuación se presenta una de ellas parque permite analizar determinadas diferencias .

3.4.2.1 Selection Search

Esta herramienta permite buscar el texto previamente seleccionado con el mouse en diferentes motores de búsqueda. La primer diferencia que se puede mencionar respecto a la herramienta de Mozilla Firefox es la manera en la que se dispara la búsqueda. En este caso no se ingresa el texto en un input sino que se selecciona a partir del texto existente en la página en la que el usuario se encuentra navegando (Fig.3.13)

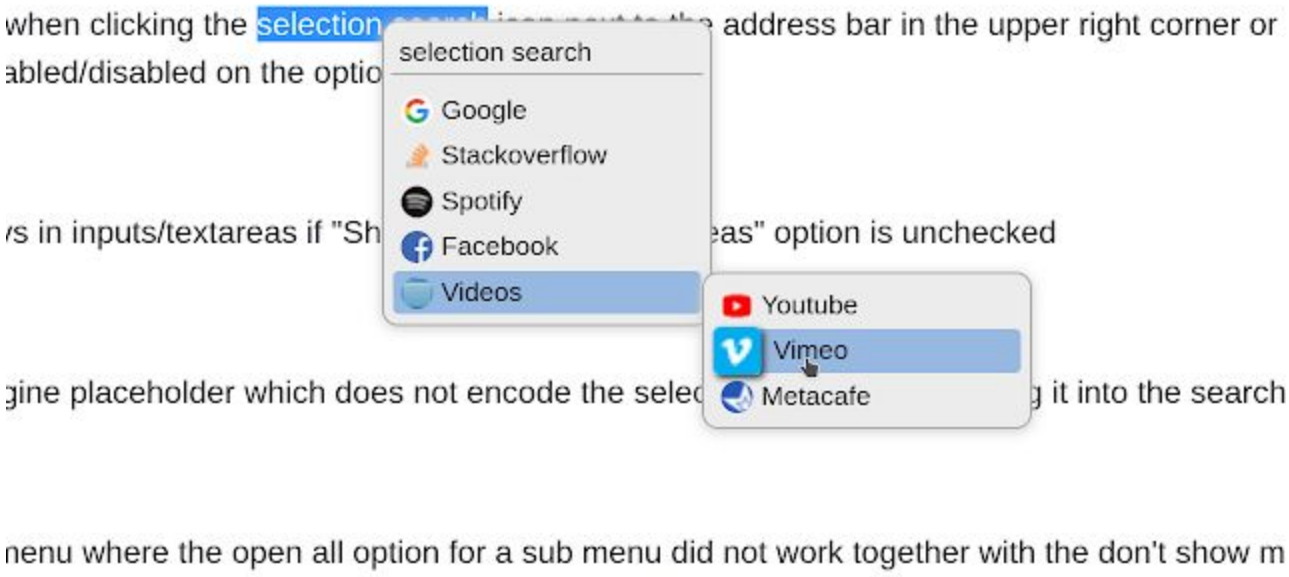


Figura 3.13: Disparar búsqueda con la herramienta Selected Search.

Aunque también se puede buscar de la forma tradicional ingresando el texto en el input, haciendo click en el icono de la extensión como se observa en la Fig.3.5 (si hay un texto seleccionado igualmente por default lo carga en el input de búsqueda pero este se puede ser modificado).

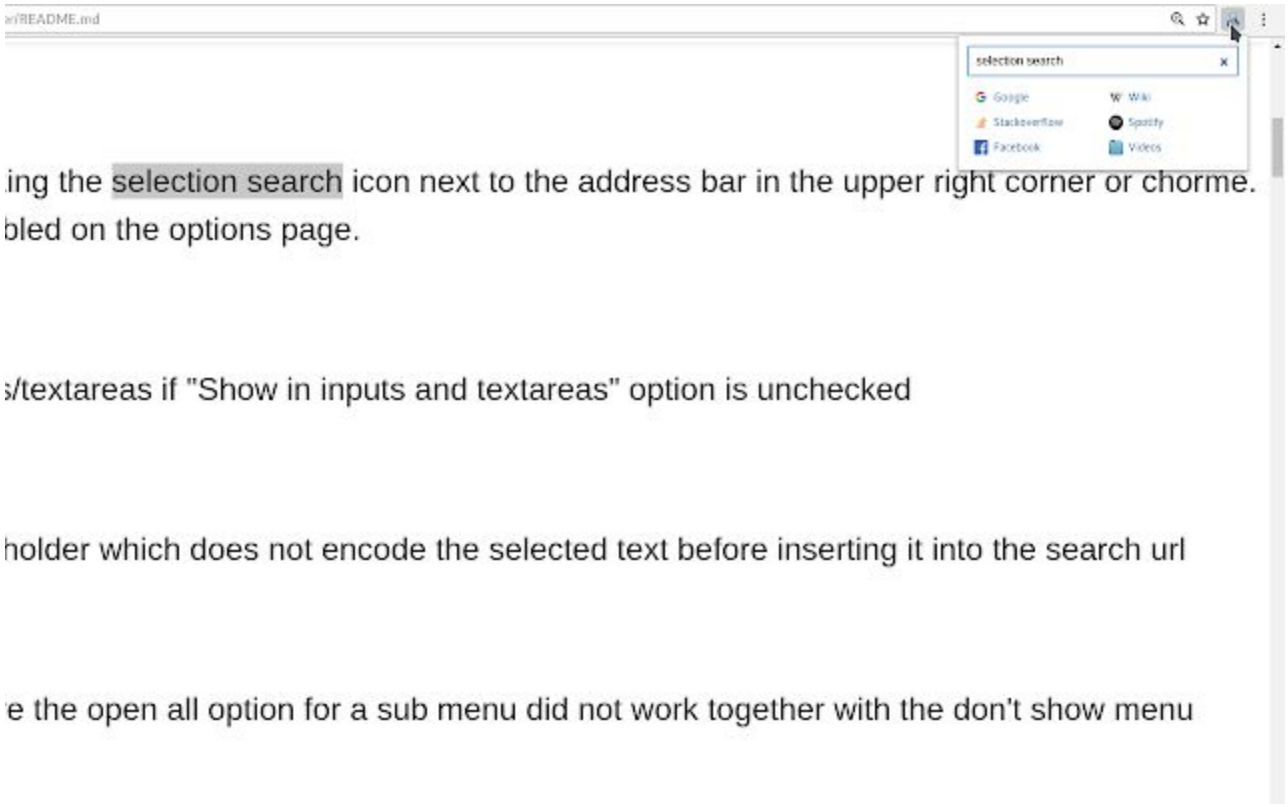


Figura 3.5: Disparar búsqueda con la herramienta Selected Search.

Pero la diferencia principal radica en la forma en la cual se añaden los motores de búsqueda a los que luego se les realizarán las consultas. En este caso no son detectados automáticamente como vimos en la herramienta anterior sino que es el usuario el que debe agregarlos, para realizar futuras consultas a los motores de búsquedas que añada.

Para agregarlos propone diferentes alternativas. La primera se puede notar al abrir las opciones de la extensión y observar una lista con los motores de búsqueda por default, al final de la lista también hay un botón que ofrece la opción de agregar nuevos (Fig. 3.14). Pero para agregarlo de esta forma se requiere cierto conocimiento basico de programacion por parte del usuario, ya que debe ingresar la url que se usa al momento de realizar la búsqueda (con sus respectivos parámetros) y que contenga la cadena especial %s, la cual

será reemplazada por el texto seleccionado al momento de realizar la búsqueda. En caso de usar el método POST, los parámetros POST deberán agregarse de la siguiente forma http://example.com/search/**{POSTARGS}**q=**%s**&type=123

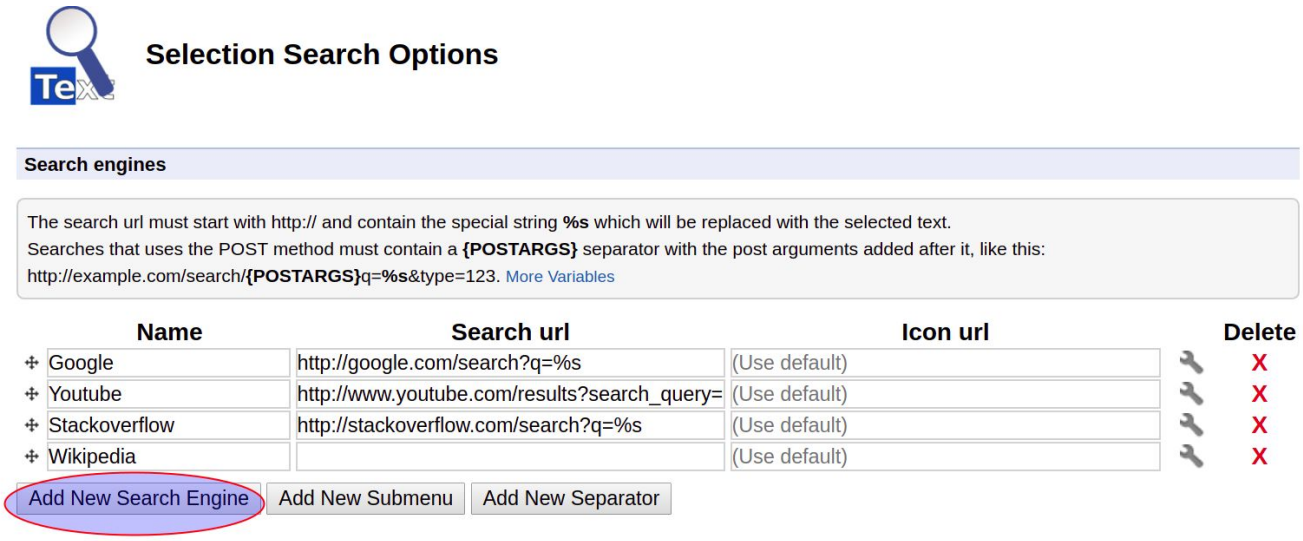


Figura 3.14: Agregar nuevo motor de búsqueda.

Otra alternativa es que el usuario se ubique en el input desde el cual se dispara la búsqueda (en el sitio que contenga el motor de búsqueda a agregar) y realice la siguiente combinación de teclas: Ctrl + Alt + Click, luego aparecerá un pop up como el que se observa en la Fig 3.15. Esta alternativa es más amigable para el usuario y no requiere conocimientos de programación. Sin embargo un texto aclara desde la propia extensión que esta opción es soportada por muchos motores de búsqueda pero no por todos.

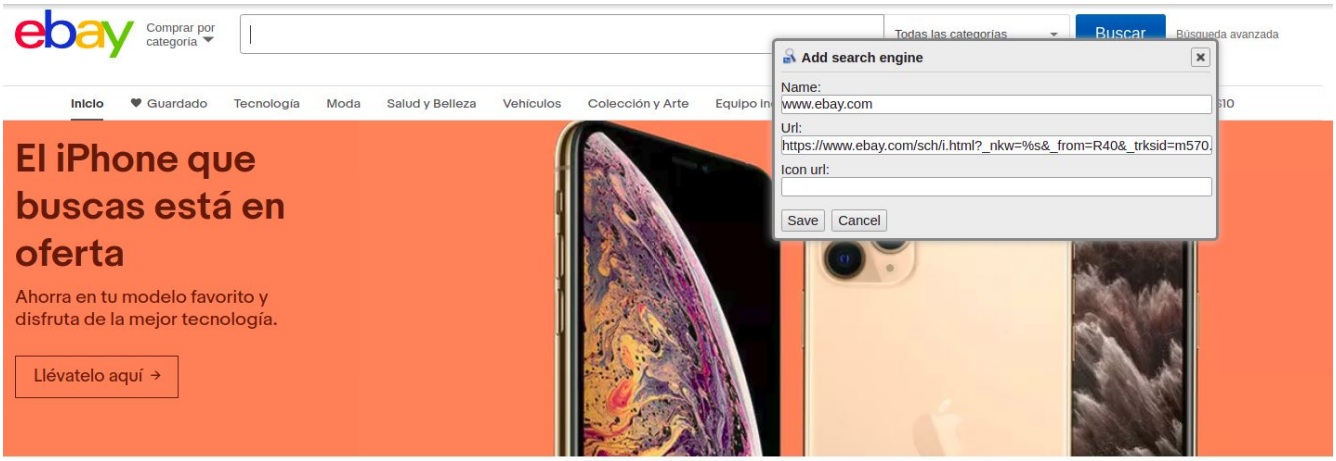


Figura 3.15: Agregar nuevo motor de búsqueda (opcion 2).

Por último la herramienta ofrece un repositorio con varios sitios soportados permitiendo agregar sus motores de búsqueda mediante un solo click y constantemente actualizado.

Tanto Selection Search como las extensiones mencionadas anteriormente mejoran un aspecto fundamental de las búsquedas Web, ya que permiten al usuario desde cualquier sitio Web en el que se encuentre disparar búsquedas hacia otros sitios Web previamente configurados. Pero como se mencionó, resta otro aspecto fundamental por mejorar sobre todo al momento de realizar búsquedas auxiliares que es el de obtener los resultados de la búsqueda *in situ*, de manera de evitar salir del contexto actual, sobre todo si la búsqueda corresponde a algo relacionado con este (como se explicó al definir las búsquedas auxiliares). A continuación se analiza una Web extension que si bien no brinda los beneficios de las ya presentadas, permite evidenciar las ventajas de no salir del contexto al realizar búsquedas auxiliares.

3.4.2.2 Google translate extension

Esta herramienta muestra a las claras los beneficios de no salir del contexto a la hora de realizar una búsqueda auxiliar en la Web, ya que la extensión permite ver traducciones de

palabras *in situ*. Cuando un usuario se encuentra en un sitio Web navegando contenido en otro idioma, traducir toda la página no siempre tiene los mejores resultados. Y si este conoce el idioma, por más avanzado que sea su nivel, siempre aparecen palabras que no conoce o propias de un dominio específico.

Entonces cuando surge la necesidad de traducir solo una palabra o frase el usuario selecciona con el mouse, copia el contenido e interrumpe la lectura para abrir una nueva pestaña y busca pegar en *google translate* o cualquier otro sitio de traducciones, lo cual hace que el mismo pierda el hilo de la lectura y retarda el objetivo final del usuario que en este caso sería finalizar la lectura. A mayor cantidad de palabras más termina afectando la tarea del usuario. Por eso Google propone una solución, que permite mediante esta simple extensión resaltar con el mouse el contenido a traducir y que la extensión previamente instalada añada la opción para traducir (Fig. 3.16).

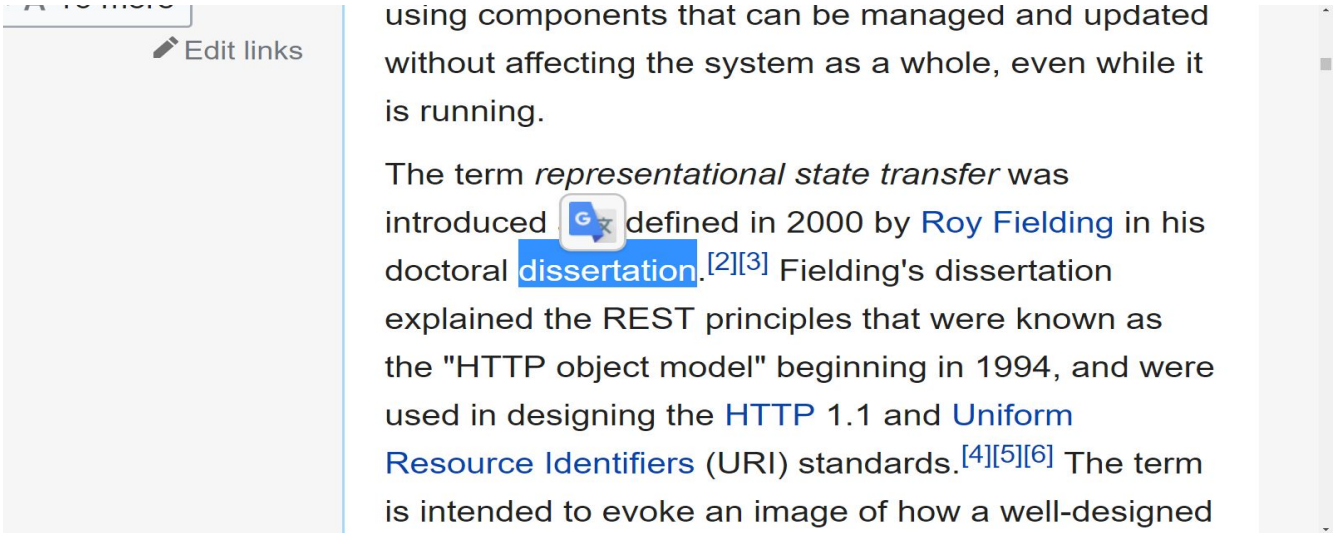


Figura 3.16: Web Extension Google Translator.

Luego al hacer click en el icono generado por la extensión el usuario podrá ver el resultado *in situ*, es decir dentro del contexto actual, como se puede observar en la Fig. 3.17.

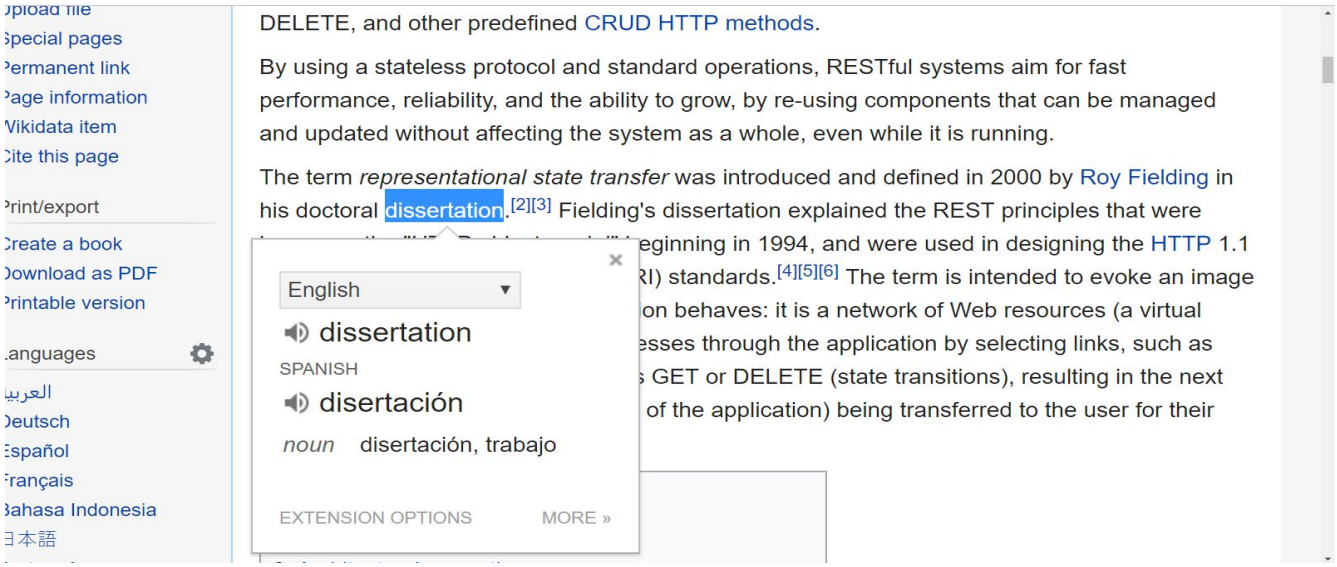


Figura 3.17: Resultado de la traducción dentro del contexto actual.

Hay que tener en cuenta que a diferencia de las herramientas ya presentadas, en este caso se muestra un solo resultado, que está acotado a la respuesta de un solo motor de búsqueda. En cambio en las herramientas anteriores, la fuente de los resultados puede ser cualquier sitio Web que cuente con un motor de búsqueda y que está disponible para agregarse en dicha herramienta.

3.4.3 WOA

Hasta aquí se han visto enfoques que presentan características específicas relacionadas en algún punto con el actual trabajo. Ya sea en cuanto a la realizacion de busquedas a motores de otros sitios Web o en cuanto a los beneficios de visualizar contenido sin salir del contexto.

En el caso de WOA el aspecto más relevante que se toma de este enfoque es la forma en la cual se identifican objetos a partir de resultados de una búsqueda Web.

A modo de resumen, WOA es una herramienta desarrollada por investigadores del LIFIA desplegada como una extensión de Mozilla Firefox, que soporta abstracción y mecanismos de estructuración de contenidos en la Web. Cómo definen en [10] su objetivo es permitir al usuario crear/extraer contenidos en la Web en forma de objetos que se puedan manipular para crear experiencias Web personalizadas.

El enfoque de WOA consiste en identificar una serie de objetos (por ej, Libros o Películas) para luego materializarlos y que estos tengan un estado interno y comportamiento, que les permita ser usados en diferentes contextos. El mayor punto de coincidencia con el enfoque presentado en el actual trabajo radica en el paso de la identificación de estos objetos. Por un lado a nivel de UI en cómo el usuario identifica estos objetos y por otro, a nivel de la herramienta, en cómo se obtienen a través del DOM.

Para entender este proceso, en la Fig. 3.18 se muestra la herramienta en cuestión, en la cual se añaden las opciones necesarias que permiten al usuario crear objetos, sin importar qué recurso web haya sido cargado en el navegador. En este caso el recurso representa una noticia de un sitio Web (que puede ser un diario o un portal de noticias), pero el mismo ejemplo se puede aplicar para productos de un sitio e-commerce, peliculas de sitios como IMBD, libros en el sitio Web de una biblioteca, etc.

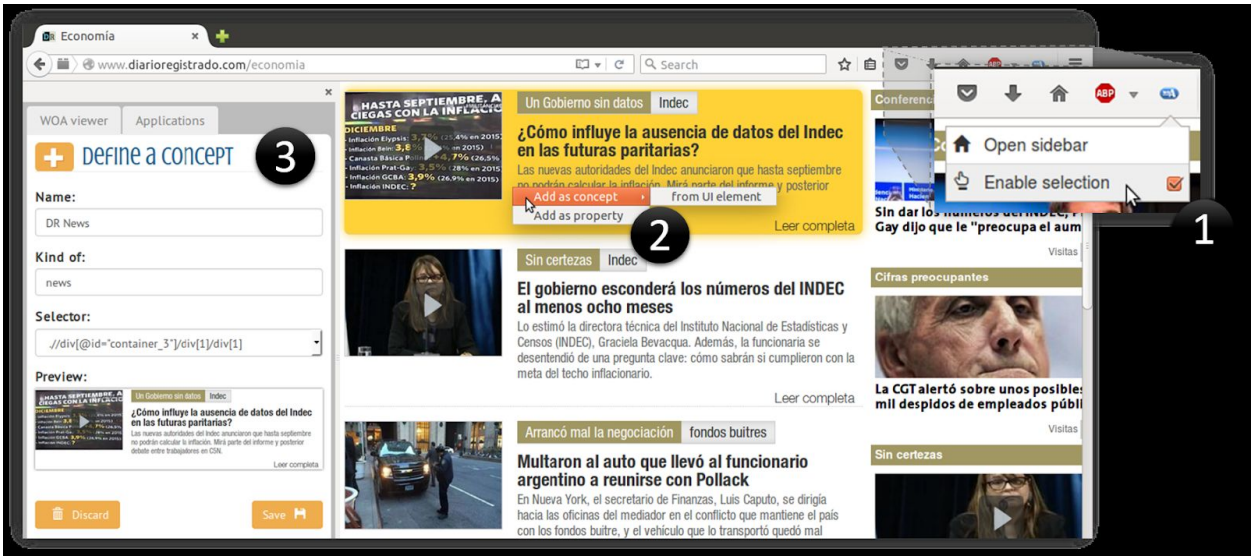


Figura 3.18: Identificando y abstrayendo conceptos [10]

El paso 1 de la Fig. 3.18 muestra simplemente un menú disponible luego de instalar la extensión que permite habilitar la selección de elementos dentro del DOM para que el usuario pueda delimitar el contenido del recurso en cuestión. En este caso al ser una noticia incluye un encabezado, una imagen, un resumen, entre otras propiedades.

A continuación, en el paso 2, se puede observar que al pasar por encima de cualquier elemento del DOM se realza en amarillo este elemento, junto con todos los elementos hijos que el mismo contenga. A partir de la identificación de un recurso, la herramienta podrá detectar el resto de los elementos que tengan la misma estructura. En ese momento entra en juego el paso 3 de la Fig. 3.18, donde desde un menú en el mismo contexto el usuario define un nombre de clase que represente al recurso (por ej. noticia), y la cantidad de ocurrencias, que la herramienta podrá identificar gracias el concepto explicado anteriormente de Xpath, y así obtener un conjunto de noticias que tengan la misma estructura que la seleccionada por el usuario.

De la misma manera que el usuario selecciona un recurso, puede hacerlo con sus respectivas propiedades. Al pasar por encima del elemento del DOM que contiene el

encabezado (por ej. el tag html <h2> </h2>), el cual puede ser definido como la propiedad “título”, y estará asociada al recurso definido previamente, de manera que cuando se recuperen las noticias desde otro contexto, cada una cuente con su titulo especifico y cualquiera de las otras propiedades que el usuario opte por visualizar.

En el caso del enfoque presentado en el actual trabajo, este proceso de identificación de los recursos será muy similar a la hora de definir servicios de búsqueda. Ya que el usuario de esta manera tendrá la potestad de definir qué información será mostrada cuando realice las posteriores consultas a los servicios de búsqueda que definió previamente. Esta idea será ampliada y mejor desarrollada en el siguiente capítulo, donde se presenta el enfoque propuesto.

Capítulo 4: Presentación del enfoque

4.1 Introducción

En el actual trabajo se propone un enfoque para realizar búsquedas auxiliares (aunque también es aplicable a las búsquedas primarias) mientras se navega por la Web mediante la utilización de aspectos de Web Augmentation.

El objetivo es permitir a los usuarios: i) Crear consultas a motores de búsqueda genéricos y/o de dominio específico; ii) Disparar consultas como búsquedas auxiliares a motores de búsqueda externos desde cualquier sitio Web que estén visitando; iii) Mostrar los resultados de la búsqueda dentro del contexto en el cual se encuentra el usuario (in situ); iv) Adaptar la visualización de los resultados en base al contenido y a las necesidades del usuario final. Para lograr lo descrito anteriormente las búsquedas customizadas de los usuarios son desplegadas como servicios de búsqueda que pueden ser accedidos desde el navegador mientras el usuario se encuentra navegando por la Web. Este enfoque evita al usuario moverse de la página Web que está visitando actualmente ya que se propone obtener los resultados de la búsqueda transparentemente y mostrarlos en el mismo contexto. A partir de esta idea el usuario podría realizar sus tareas de manera fácil y más rápidamente que si tuviera que repetir los mismos procesos para búsquedas similares y también se le habilita la posibilidad de comparar sus resultados sin moverse entre pestañas o ventanas. Los servicios de búsqueda son especificados en términos de un modelo, y esta especificación puede ser compartida hacia otros usuarios. Una herramienta de soporte, exitosamente utilizada en un conjunto de sitios Web populares, demuestran la fiabilidad de este enfoque en el Capítulo 5. Además se evidenciara en lo que resta del trabajo, que el actual enfoque incluye e integra múltiples aspectos que por separado fueron exitosos y muy utilizados, como los que se desarrollaron en el Capítulo 3.

Los motores de búsqueda retornan una página de resultados con una lista de documentos Web (URIs) que coincidan con el criterio de búsqueda. Los resultados son usualmente presentados como una lista de titulos de pagina, 1 o 2 frases tomadas del contenido y alguna imagen relacionada. Avances recientes en las interfaces de búsquedas Web proporcionan, para un conjunto de tipos de elementos predefinido, como películas, libros o recetas, *snippets* enriquecidos que ayuden al usuario a reconocer características relevantes de cada elemento del resultado (por ej. películas que se van a estrenar o *reviews* para determinada película). En algunos casos estos *snippets* incluyen los datos que el usuario está buscando, esto es posible ya que los creadores del sitio Web incluyen datos estructurados en sus páginas que pueden ser reconocidos e interpretados y, por lo tanto, pueden ser usados para crear aplicaciones. Ver la Web como un repositorio de datos estructurados e interconectados puede ser visto como el último objetivo de la Web Semántica. Sin embargo los usuarios finales no siempre tienen los medios para explotar o añadir información a la Web Semántica. Por ello, las herramientas que se presentan en este trabajo permiten a los usuarios extraer y usar datos estructurados a partir de cualquier sitio Web, sin necesidad de que estos tengan ningún entrenamiento como desarrolladores Web.

En el actual enfoque se provee a los usuarios finales la posibilidad de ejecutar búsquedas auxiliares, con el agregado que no se necesitan componentes *server-side* (a diferencia de otros enfoques) sino que el soporte es puramente *client-side* sin necesidad incluso de habilidades de programación por parte de los usuarios. Adicionalmente, se permite definir la semántica y la estructura de los resultados que se obtienen, sin restricciones en cuanto al tipo de elemento. Esto sugiere una misma estructura de información para todos los usuarios finales que usen la herramienta, la cual posteriormente puede ser usada para consultar un repositorio común o utilizar visualizaciones similares para los objetos que compartan una misma clase (por ej. libro o película).

Finalmente, las investigaciones respecto al comportamiento en búsquedas de información se focalizaron mayormente en cómo los individuos buscan información. Sin embargo, en

muchos contextos, este proceso involucra colaboración. La herramienta para la definición de servicios de búsqueda propone almacenar la definición de los servicios en el local storage del navegador. Esta propiedad permite que los datos persistan almacenados entre las diferentes sesiones del navegador. De esta manera los usuarios pueden exportar e importar definiciones, y así permitir que se compartan, por ejemplo, vía email.

Para comenzar a comprender con más detalle el enfoque, a continuación se detallan los 2 aspectos fundamentales. En primer lugar se explica el concepto de servicios de búsqueda y cómo deben ser definidos por el usuario para su posterior uso. Luego se describe la arquitectura del enfoque para comprender más en detalle cómo el usuario puede definir estos servicios de búsqueda y posteriormente como se visualizan los resultados dentro del contexto. Además, a lo largo del capítulo, se analizan las ventajas de este enfoque respecto a los ya vistos y cómo se integran mucha de los beneficios explicados en una única propuesta.

4.2 Servicios de búsqueda orientados al usuario final

Un amplio número de escenarios de búsqueda permitirían apreciar que la integración ofrecida por los sitios Web y los navegadores, en combinación, pueden afectar la experiencia del usuario final, ya que si este tuviera la necesidad de realizar repetidamente operaciones extra para obtener la información deseada debería realizar el proceso explicado anteriormente (abrir una nueva pestaña o ventana, ingresar la URL de la Web, realizar la búsqueda en la nueva Web, etc.). Además si el usuario está realizando una búsqueda auxiliar, querría refinar la consulta o volver al contexto original de información para realizarlo.

El enfoque que se presenta está basado en una arquitectura flexible que permite a los usuarios finales customizar la forma en la cual realizan búsquedas Web. Los usuarios podrán realizar búsquedas auxiliares sin salir del contexto actual. Mientras otras búsquedas son realizadas en relación a un objetivo específico en mente, la búsqueda auxiliar es

anidada entre otras búsquedas y su objetivo es proveer información complementaria a la que se encuentra en el contexto actual del usuario. La solución propuesta propone permitir tratar con tareas de búsquedas auxiliares mientras el usuario navega en la Web mediante Web Augmentation, reduciendo los esfuerzos del usuario y, por lo tanto, los tiempos de ejecución y de evaluación.

El proceso de búsqueda apuntado implica los siguientes pasos: 1) definir una consulta; 2) seleccionar un motor de búsqueda; 3) ingresar la consulta y disparar la busqueda; y 4) inspeccionar e interactuar con los resultados. El objetivo es mejorar la experiencia del usuario con tareas de búsqueda, particularmente entre el paso 2 y el 4. Para el primer paso, se mantiene el lenguaje de consulta impuesto por el buscador subyacente. Para los pasos siguientes el enfoque propone:

- Disparar búsquedas desde la página Web actual para reducir la interacción requerida para realizar una búsqueda en cualquier motor de búsqueda exterior.
- Transformar los resultados de la búsqueda (elementos del DOM) en objetos del dominio con una semántica y con una estructura específicas.
- Integrar las instancias de dominio resultantes en la página Web actual para ofrecer visualizaciones alternativas e interactuar con los resultados.

Para lograr estos objetivos, se propone en una primera instancia permitir a los usuarios encapsular motores de búsqueda de aplicaciones Web existentes en servicios de búsqueda. Dado que no todas las aplicaciones Web que soportan búsquedas proveen una API, se propone reproducir automáticamente la interacción UI requerida para realizar una búsqueda en la página Web. Esto implica que el usuario debe seleccionar los componentes UI del motor de búsqueda para crear los servicios de búsqueda.

En segunda instancia, integrar los nuevos servicios de búsquedas con el mecanismo de búsqueda del navegador Web para poder realizar búsquedas. Con esto, los usuarios deben

ser capaces de usar los servicios de búsqueda creados desde cualquier otro sitio Web existente.

Finalmente se deben mostrar los resultados en el contexto del sitio Web actual, ofreciendo diferentes formas de visualizaciones que soporten búsquedas auxiliares, así como también primarias. Esto se puede lograr parseando el DOM, mediante la extracción de los resultados de la búsqueda y la creación de instancias del objeto dominio.

4.3 Arquitectura

La arquitectura del enfoque puede dividirse en 2 grandes capas, una con lo relacionado a la definición de búsquedas Web y otra a la visualización de los resultados. Por un lado se desarrolla una arquitectura que soporta la creación de servicios de búsqueda desde una Web extensión para permitir a usuarios finales usar estos nuevos servicios que define, desde cualquier sitio Web que se encuentre visitando. Mientras que por otro lado, se ofrece la posibilidad de extraer los resultados y renderizarlos *in situ* al enviar consultas a estos servicios previamente definidos, permitiendo complementar la información que ve el usuario, sin tener la necesidad de salir del contexto actual.

La propuesta incluye varios aspectos ya analizados en los trabajos relacionados pero a su vez los integra en una única herramienta y es superadora en otros aspectos que se analizan a continuación y serán útiles para presentar el enfoque con mayor nivel de detalle, y al mismo tiempo, compararlo con los enfoques ya presentados en el Capítulo 3.

4.3.1 Capa de definición de servicios de búsqueda

En lo que respecta a los enfoques existentes para realizar búsquedas Web a distintos motores de búsqueda externos, se vio que los buscadores a los cuales se permite realizar consultas son acotados. Pero en el enfoque actual, el usuario puede definir servicios de

búsqueda a partir de cualquier sitio Web que provea un motor de búsqueda sin necesidad de que el mismo provea una API. Esto es gracias a que se basa en la definición de un servicio que el usuario crea a partir de su forma de interactuar con determinado motor de búsqueda, permitiendo customizar la información relevante que quiere obtener luego.

Típicamente la interfaz de un motor de búsqueda dentro de un sitio Web se compone de un formulario con inputs y un botón (submit) para enviar la consulta, así como también opciones para filtrar resultados y ordenarlos, y finalmente algún mecanismo de paginación de resultados. En la Fig.4.1 se observa un ejemplo de un conocido sitio Web *e-commerce*.

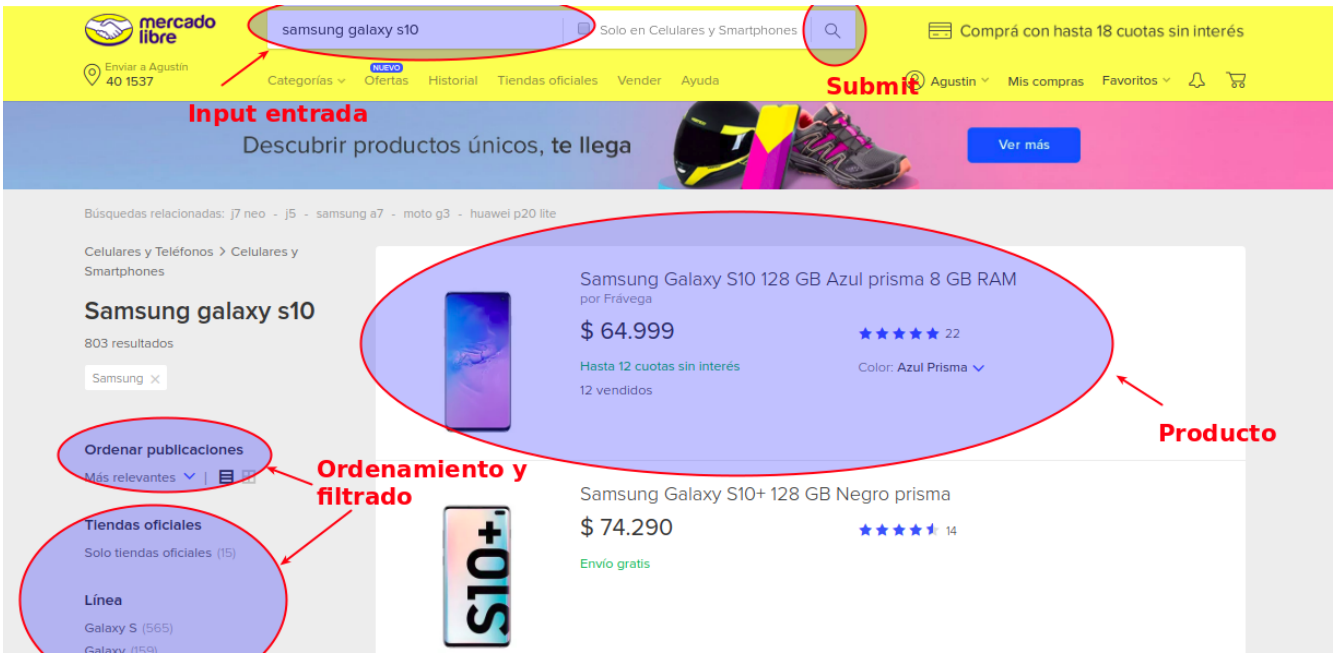


Figura 4.1: Interfaz de un motor de búsqueda

Lo que se propone es abstraer todos estos componentes UI, que al fin y al cabo son elementos del DOM, creando a partir de estos un servicio de búsqueda. Luego estos servicios de búsqueda podrán emular el comportamiento del usuario y recuperar los resultados correspondientes a partir de las consultas que reciban. El comportamiento incluye

acceder a la URL del sitio Web que contiene el motor de búsqueda que se desea consultar, ingresar un texto en el input de entrada, disparar la búsqueda y obtener los resultados.

Además los resultados no solo son elementos del DOM sino más bien abstracciones de objetos de dominio subyacentes. En el ejemplo de la Fig.4.1 los resultados pueden ser abstraídos como el objeto de dominio "Producto", el cual puede incluir propiedades de dominio como título, precio, imagen, etc. Incluso el objeto puede tener además propiedades cuyos valores son tomados de otro DOM obtenido de otra URL, por ejemplo información que se obtiene al acceder a la sección específica de determinado producto.

Los servicios de búsqueda son especificados a través de herramientas visuales que permiten a los usuarios crearlos mediante la selección de elementos del DOM. El proceso es similar a lo explicado en WOA, pero en este caso es utilizado tanto para la definición del servicio de búsqueda como de la estructura de los objetos que serán retornados al consultar el servicio.

Por último, la otra gran diferencia respecto a los enfoques presentados, como Seach Bar de Mozilla o las Web extension existentes, radica en que los resultados serán renderizados dentro del sitio actual del usuario, gracias a la capa de visualización que se introduce en la siguiente sección.

4.3.2 Capa de visualización

Dentro de la herramienta además de la definición de los servicios de búsqueda, existe otra capa que es la de visualización, la cual toma los resultados de las búsquedas realizadas a dichos servicios y permite a los usuarios interactuar con estos resultados de diferentes maneras, habilitando diferentes tipos de visualización en base al contenido. Esta característica tiene un punto de coincidencia con el enfoque de Google Translator en la forma en que visualiza el resultado de la traducción dentro del contexto actual en el cual se encuentra el usuario. Con la clara diferencia que en la presente propuesta se permite mostrar

resultados de cualquier sitio Web de diferentes dominios y no solo traducciones de un solo sitio Web. Además permite diferentes tipos de visualizaciones adaptables a las preferencias del usuario final.

Entre la definición de servicios y su posterior utilización para visualizar resultados, entra en juego una capa intermedia, la cual a partir de la selección de un texto en cualquier página Web, permite realizar consultas a cualquier servicio de búsqueda que el usuario seleccione (el cual fue definido previamente). Algo similar a lo mostrado en la Web Extensión Selection Search, específicamente en la Fig. 3.13

Una vez que el servicio de búsqueda obtiene los resultados del DOM del sitio Web al que se disparó la búsqueda, puede visualizarlos en el contexto gracias a la capa de visualización, que genera dinámicamente un nuevo elemento en el sitio Web actual renderizando los resultados obtenidos. Este elemento html se define con el tag <iframe> (qué significa inline frame) y el mismo representa un contexto de navegación anidado, el cual permite incrustar otra página html en la página actual. En este caso solo será un porcentaje de la pantalla, que permite visualizar los resultados in situ. Este no es fijo en la pantalla sino que su posición puede cambiarse al seleccionarlo con el mouse y moverlo a través de la página, para que el usuario lo ubique donde crea más conveniente.

Además, otro beneficio que el enfoque puede brindar al usuario es la capacidad de proveer diferentes alternativas a la hora de visualizar estos resultados. Dependiendo de qué objeto de dominio represente, se pueden ofrecer distintas visualizaciones. En el primer ejemplo, el enfoque ofrece una visualización en forma de tabla gracias a la integración de la librería Datatables que permite generar una tabla a partir de los datos obtenidos de los resultados de la búsqueda, y luego la misma se inserta dentro del elemento iframe y se renderiza como se observa en la Fig. 4.2

Figura 4.2: Visualización de resultados en forma de tabla

Asi como esta visualización, el enfoque permite que se integre cualquier otra librería javascript para generar todo tipo de visualizaciones como imágenes, gráficos estadísticos, etc, dependiendo la fuente de datos, de los resultados obtenidos y de las preferencias del usuario.

Por ejemplo, en lo que respecta a los gráficos estadísticos, se ofrece una visualización que permite generar diferentes tipos de gráficos a partir de datos de entrada numéricos como por puede ser el precio de un producto e-commerce o cualquier otro objeto que maneje propiedades de este tipo (vuelos, hoteles, etc). Esto es posible mediante la integración de una librería javascript, en este caso ChartJs, que soporta una amplia cantidad de tipos de graficos (grafico de torta, de barra, de líneas, etc).

En la Fig.4.3 se observa un ejemplo con la visualización de precios de productos de un sitio e-commerce (en este caso libros), ordenados de menor a mayor según su precio (indicado en eje Y) mientras que en el eje X se indica su título. Esta misma idea se puede aplicar a

distintas entradas de datos, y generar gráficos más complejos. Por ejemplo si la fuente de datos permite obtener los precios promedio de un vuelo o de hoteles por cada mes, el usuario puede a simple vista saber cual es el mes más barato para viajar a determinado destino.

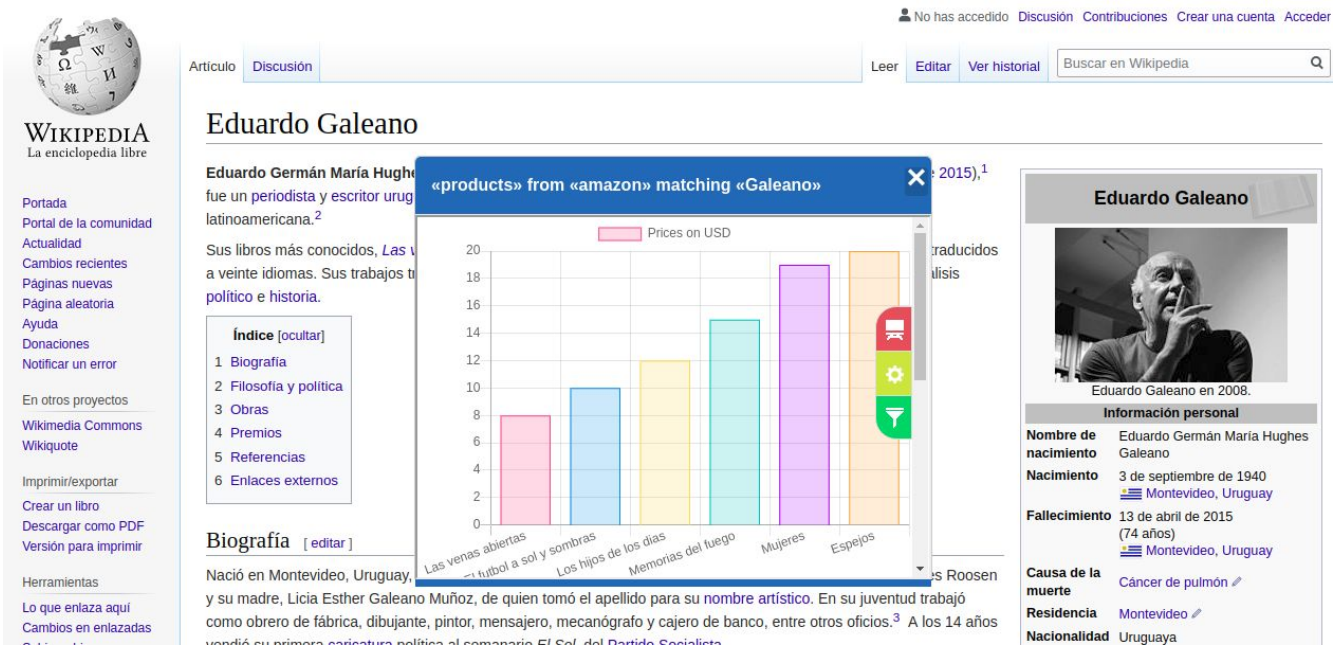


Figura 4.3: Visualización de resultados en forma de gráficos estadístico

Por último el enfoque está diseñado para permitir al usuario switchear dinámicamente entre las diferentes visualizaciones según cuales sean las soportadas por los objetos representados, es decir si estos tienen algunas propiedades de tipo numérico o imágenes se podrán ofrecer una mayor cantidad de visualizaciones. En cambio para un objeto que solo cuente con propiedades de tipo texto, las visualizaciones son acotadas (por defecto se renderizan en forma de tabla).

En el siguiente capítulo se presenta la herramienta que da soporte a este enfoque,mediante un ejemplo concreto que permite entender mejor el funcionamiento de la misma.

Capítulo 5: Presentación de la herramienta

5.1 Introducción

Para entender mejor el enfoque, y adentrarse en el uso de la herramienta propuesta, se presenta un simple escenario, donde un usuario se encuentra navegando en el sitio Web Wikipedia en un artículo de un escritor (en este caso Eduardo Galeano). En un momento dado el usuario puede requerir buscar libros del mismo en el catálogo de libros de la biblioteca de su universidad. Con esto en mente, y considerando el enfoque presentado, el usuario sería capaz de lanzar una búsqueda desde el sitio Web actual, resaltando el nombre del autor y posteriormente seleccionado el servicio de búsqueda deseado. En el caso de este ejemplo, los resultados son obtenidos del sitio Web de la biblioteca de New York (nypl).

5.2 Presentación de la herramienta

Continuando con el ejemplo introducido previamente, el primer paso que debe realizar el usuario es definir el servicio de búsqueda que posteriormente consumirá, lo cual tiene más sentido si es un usuario que realiza consultas frecuentes al motor de búsqueda en cuestión, ya que con solo definirlo una vez podrá consultarlo las veces que desea, sin importar en qué sitio Web se encuentre navegando. En la Fig. 5.1 entra en acción el primer paso de la definición del servicio de búsqueda, en este caso el usuario ingresa al sitio nypl, realiza el proceso típico de una búsqueda y luego de obtener los resultados, al hacer clic en el icono de la Web extension (marcado con un círculo rojo en la esquina superior derecha), se abre el menú lateral que se observa a la derecha, y da inicio a la fase de definición del servicio de búsqueda al clickear el botón rojo *New Service*.

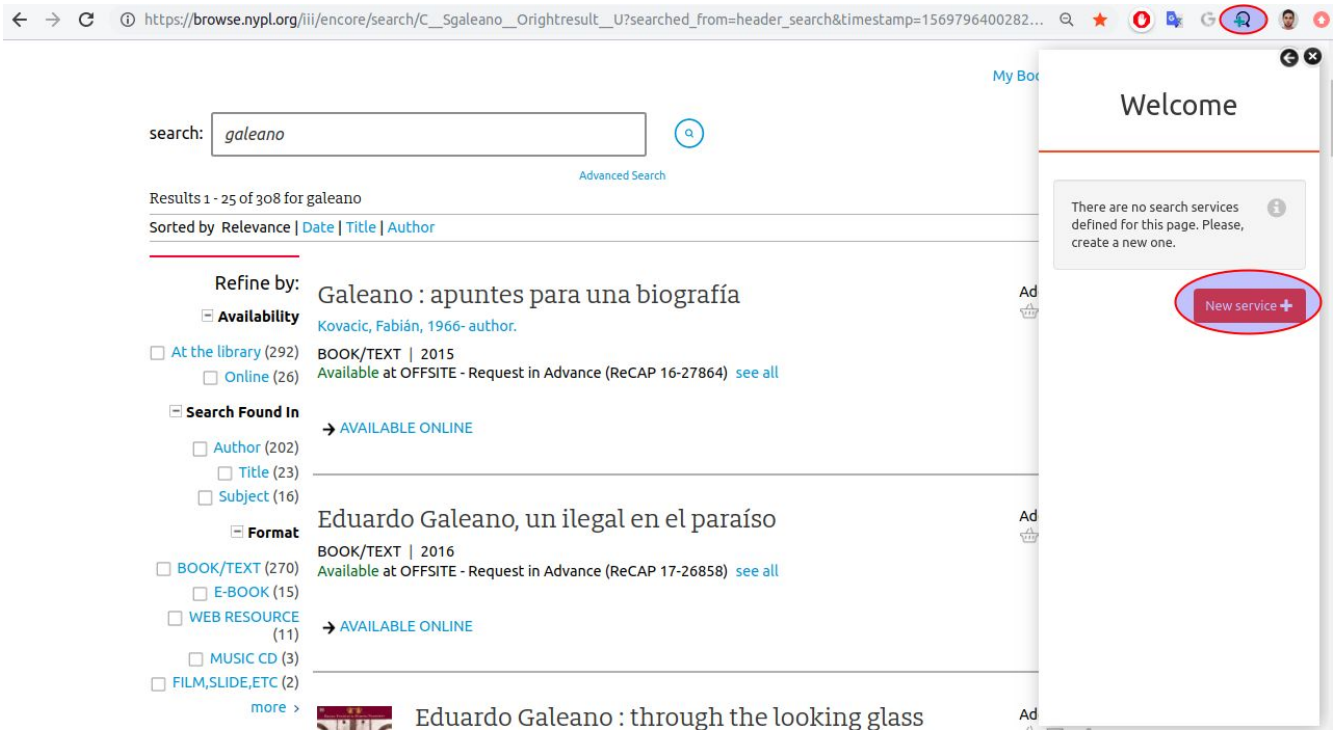


Figura 5.1: Paso 1 de la definición de un servicio de búsqueda

Si bien en el ejemplo la etapa de definición coincide la búsqueda con la que se realiza posteriormente (Eduardo Galeano), es una mera coincidencia y debe quedar en claro que los resultados se adaptan al texto seleccionado y en caso de ser otro autor (por ej. Julio Cortázar) la herramienta emulara la misma búsqueda que realizó el usuario pero ingresando en el input de entrada el texto seleccionado en ese momento dado.

Continuando con el proceso de creación del servicio de búsqueda, el siguiente paso comienza luego de clickear el botón *New Service* del paso 1. En este segundo paso se despliega un input que permite definir el nombre del servicio. Como se puede observar en la Fig 5.2 se define nypl (será el nombre que aparecerá posteriormente cuando tengamos que seleccionar a qué motor disparar una determinada búsqueda).

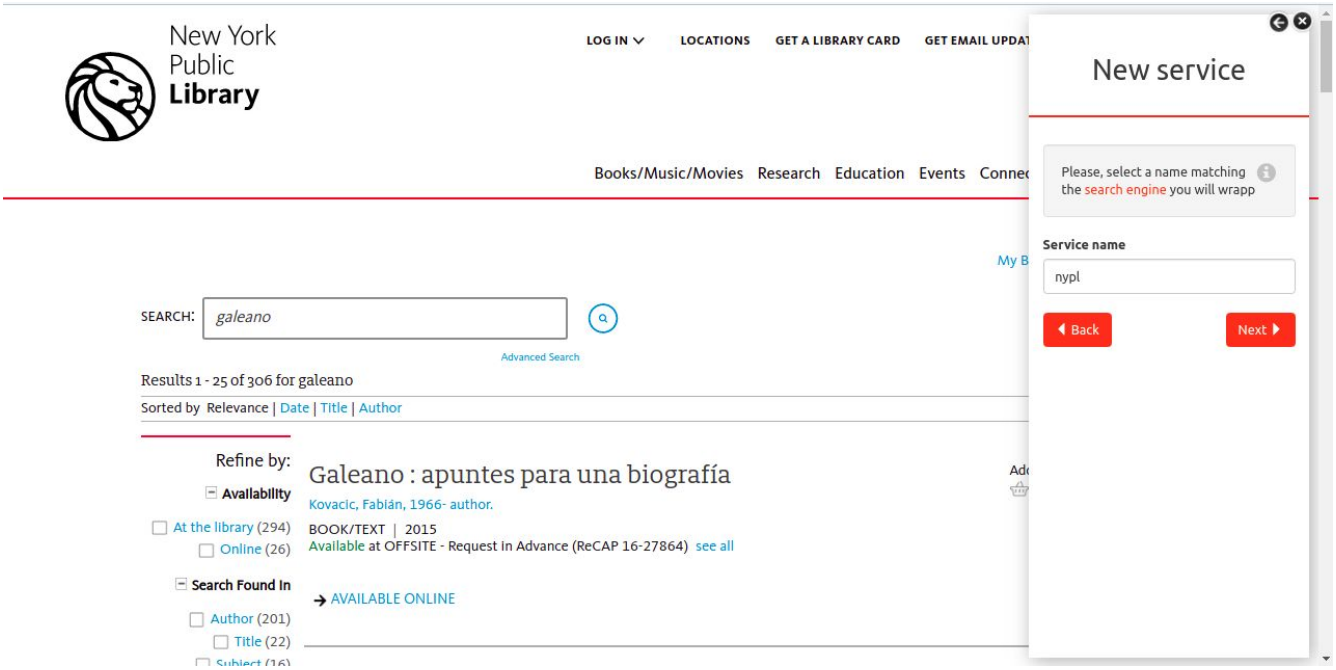


Figura 5.2: Paso 2. Ingresar nombre de servicio de búsqueda

Luego de tipear el nombre y clickear el botón *Next*, se da lugar al tercer paso. Aquí el usuario debe seleccionar los componentes UI a partir de la interfaz del motor de búsqueda que ofrece el sitio Web. En primera instancia debe clickear el elemento de la página donde se ingresa el texto de la búsqueda (paso 3) y en segunda instancia (paso 4) deberá clickear sobre el elemento que dispara la búsqueda, que suele ser un botón.

Esto se puede observar en la Fig. 5.3 y Fig. 5.4 respectivamente. Cabe aclarar que en caso de que el formulario de búsqueda no cuente con un elemento para disparar la búsqueda, y la misma se realice con la tecla *Enter*, esto puede indicarse en el paso 4, desplegando el combo que se observa en el menú lateral y cambiando la opción seleccionada por default.

Figura 5.3: Paso 3. Seleccionar input donde se ingresa el texto de la búsqueda

Figura 5.4: Paso 4. Seleccionar botón que dispara la búsqueda

Una vez que el usuario indica a la herramienta cuales son los elementos para realizar una búsqueda en determinado sitio Web, el siguiente paso es que a partir de los resultados de la búsqueda que se observan en la pantalla, el usuario delimite en el DOM cual es la estructura del objeto de dominio, en este caso “Libro”. Esto se puede observar en los pasos 5 y 6 en la Fig. 5.5 y la Fig. 5.6 respectivamente.

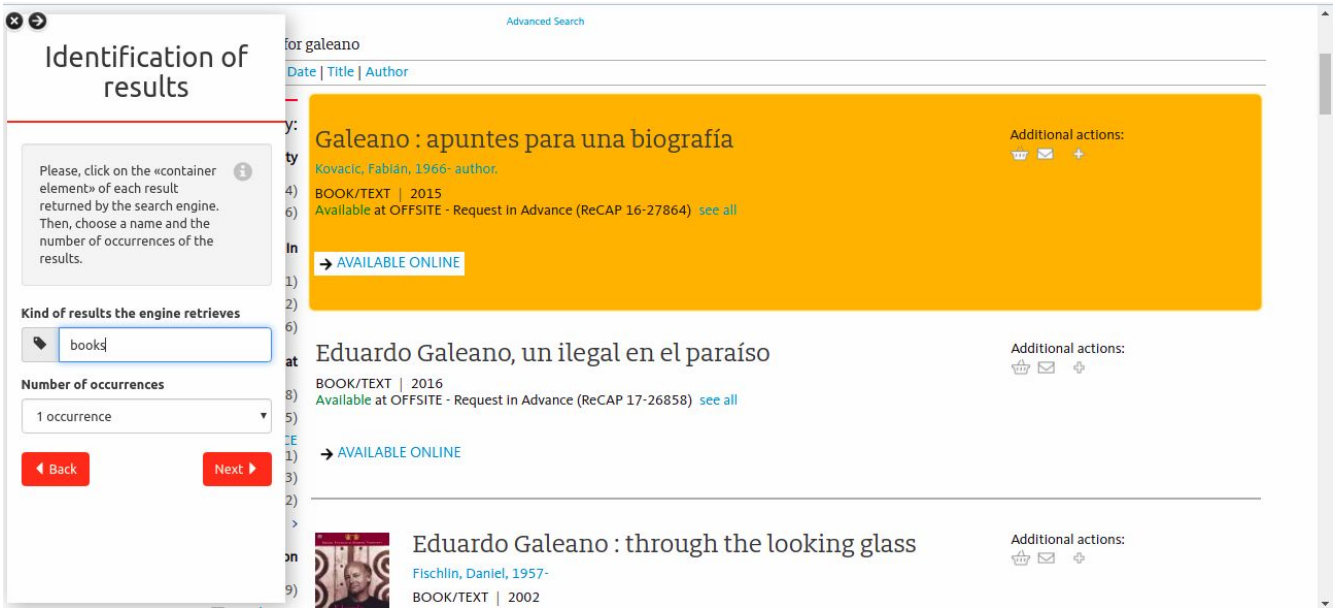


Figura 5.5: Paso 5. Delimitación de la estructura del objeto de dominio

En el paso 5 (Fig. 5.5) la herramienta habilita al usuario de manera que al pasar con el mouse se “pinte” de amarillo el elemento del DOM (y los elementos anidados que este contenga), lo cual permite delimitar manualmente el objeto de dominio. Luego en el paso 6, a partir de las ocurrencias que automáticamente detecta la herramienta, el usuario confirma cual es la cantidad total y verifica al hacerlo que se “pinte” la totalidad de resultados que se deben incluir en futuras búsquedas.

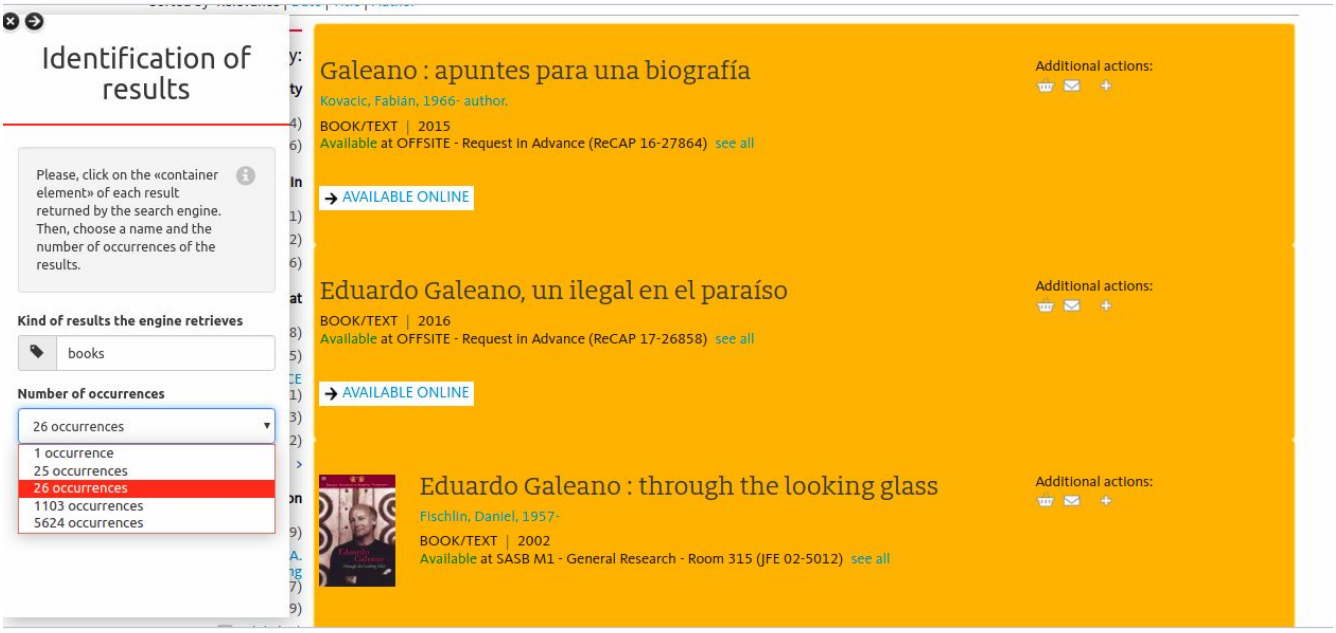


Figura 5.6: Paso 6. Seleccionar cantidad de ocurrencias del objeto.

El paso 7 permite al usuario customizar la información que se muestra de cada objeto del dominio (en el ej. Libro) cada vez que se obtengan los resultados desde otros sitios Web.

Para ello, en base a la estructura de cualquiera de los objetos previamente delimitados, podrá seleccionar a partir de elementos internos del DOM cuales son las propiedades que conforman el objeto. En la Fig.5.7 se observa cómo al seleccionar el título de uno de los libros de despliega un input para ingresar el nombre asociado a dicha propiedad (“Título”). De la misma forma podrá definir propiedades como autor, año, disponibilidad del libro, etc.

Si bien agregar un nuevo servicio de búsqueda requiere de este esfuerzo extra, el proceso es muy ágil e intuitivo y no requiere conocimientos de programación por parte del usuario. Además con definir un servicio una única vez luego se ahorra tiempo en cada futura utilización del mismo.

Figura 5.7: Paso 7. Definir propiedades del objeto.

Una vez definido el servicio de búsqueda, el usuario puede consumirlo desde cualquier sitio Web en el que se encuentre navegando. Volviendo al ejemplo, si se encuentra en Wikipedia puede realizar la consulta al sitio Web de la biblioteca de New York, para ello basta con seleccionar el texto, hacer clic derecho para que se despliegue el menú del navegador donde la herramienta ya habra desplegado los servicios que haya definido el usuario y finalmente clickear el servicio que desea consultar, como se puede observar en la Fig. 5.8.

Los resultados obtenidos por la herramienta a través del proceso ya explicado, pueden observarse en la Fig. 5.9. Dependiendo de la cantidad y el orden de las propiedades definidas por el usuario, pueden observarse las primarias en primera instancia, y las propiedades secundarias ocultas, las cuales pueden desplegarse como se observa en la Fig. 5.10, característica que ofrece la librería Datatables.

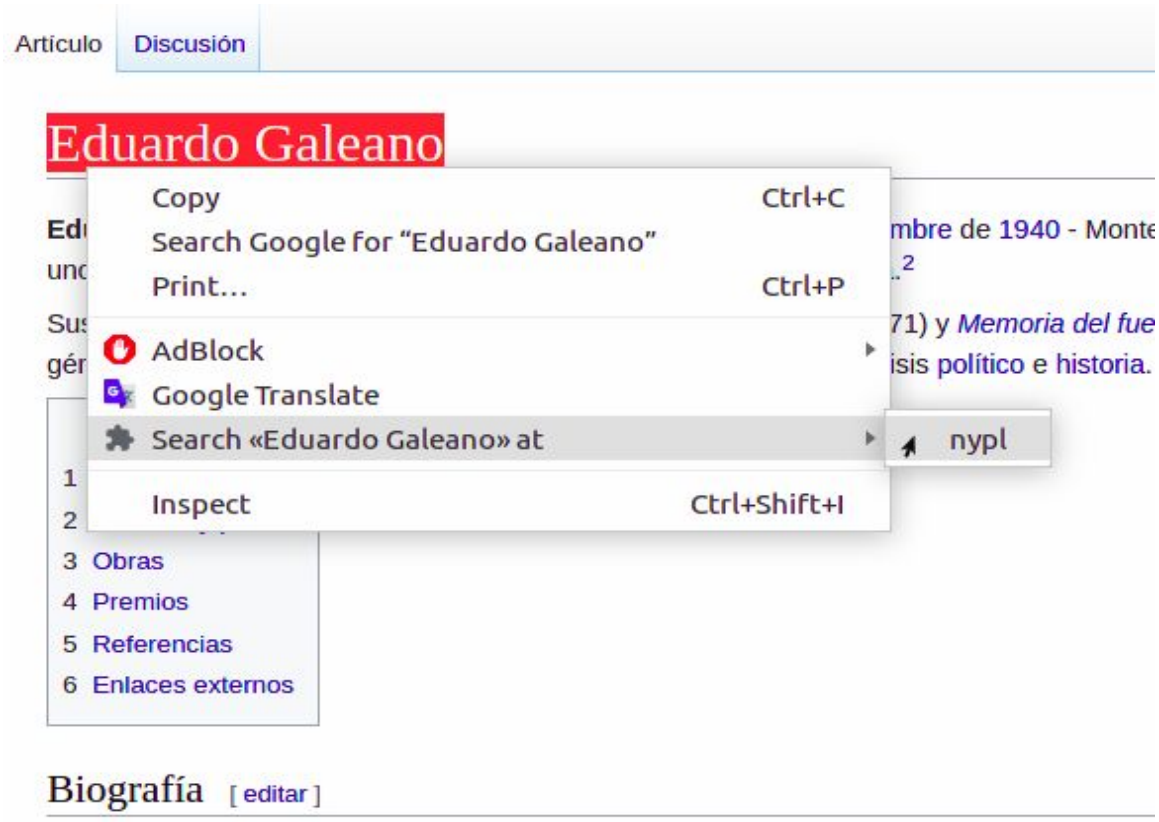


Figura 5.8: Búsqueda auxiliar desde Wikipedia a nypl

Figura 5.9: Visualización de los resultados in situ

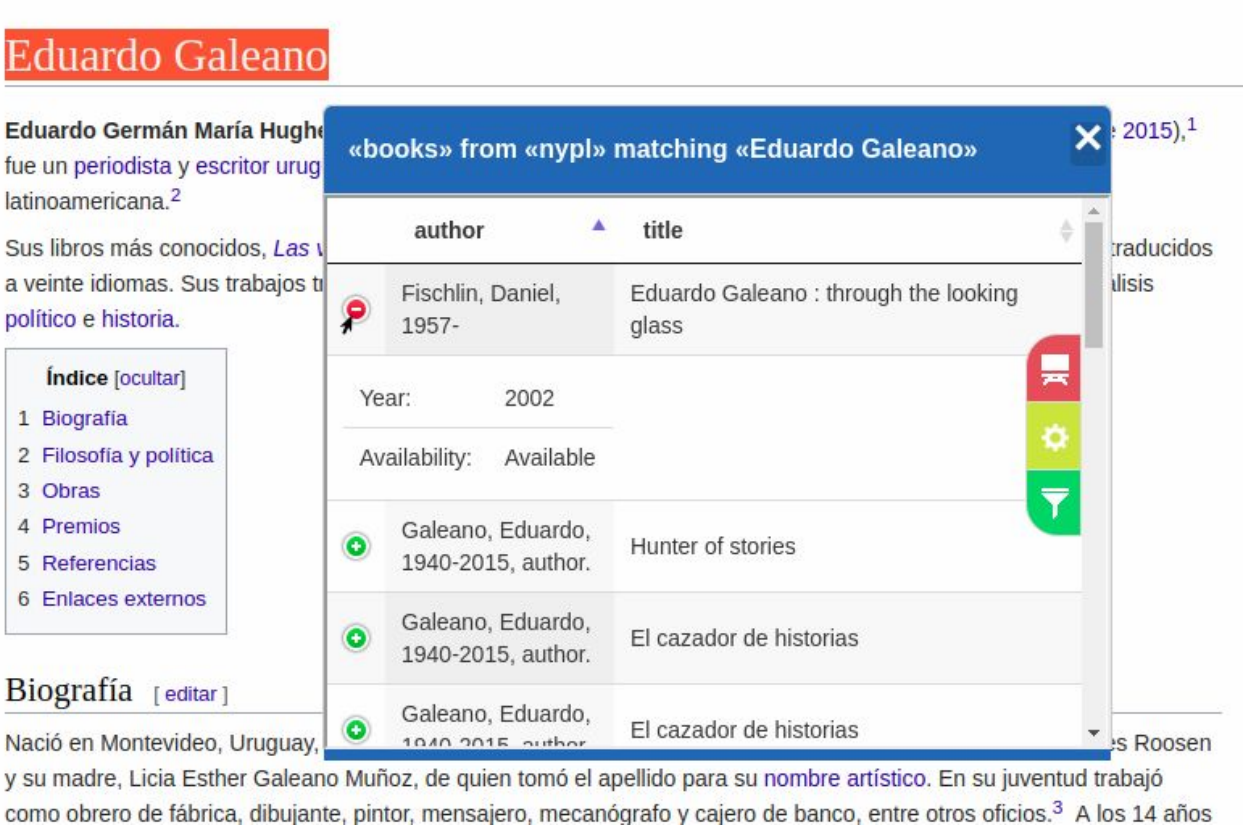


Figura 5.10: Visualización de las propiedades ocultas

Lógicamente las consultas pueden ser disparadas desde cualquier sitio Web, todos los servicios de búsqueda previamente definidos estarán disponibles en el menú contextual del navegador. Además dentro del mismo contexto pueden realizarse múltiples consultas, ya sea al mismo servicio de búsqueda o a otros. Siguiendo con el ejemplo, suponiendo que el usuario definió nuevos servicios de búsqueda para obtener los precios de los libros en sitios e-commerce como Amazon, esto le permitira disparar nuevas búsquedas auxiliares a partir de resultados previamente obtenidos. Como se puede observar en la Fig. 5.11, donde el usuario selecciona el título de un libro específico y dispara la consulta a un nuevo motor de búsqueda (en este caso Amazon) obteniendo los resultados como se observa en la Fig. 5.12

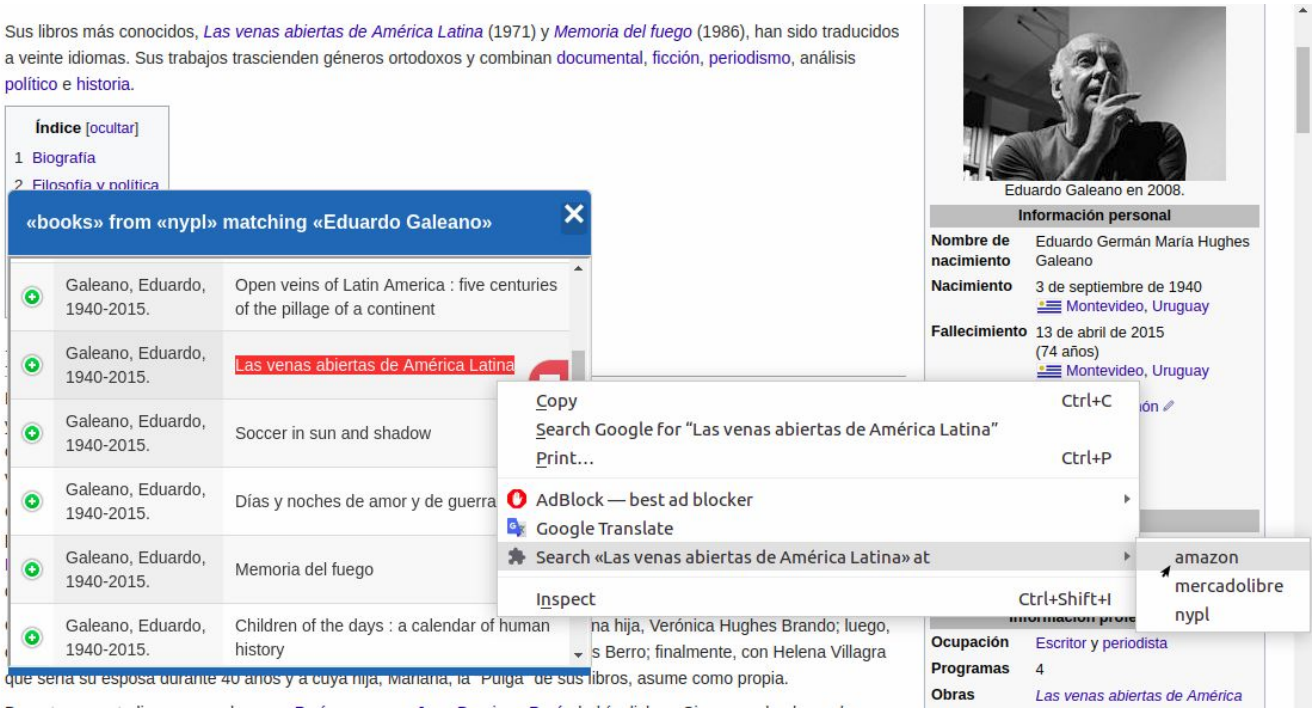


Figura 5.11: Nueva búsqueda auxiliar a partir de los resultados previos

Figura 5.12: Visualización de los resultados in situ de la nueva búsqueda

Por último, la herramienta ofrece un menú a partir del cual se podrán cambiar dinámicamente los diferentes tipos de visualizaciones provistos por la herramienta y basados en las preferencias del usuario. Si bien este es un aspecto a explotar en cuanto a las posibles mejoras, se puede observar como la herramienta soporta la renderización de distintas visualizaciones en la Fig. 5.13, donde se despliega el menú para, a partir de los resultados obtenidos, generar un gráfico estadístico.

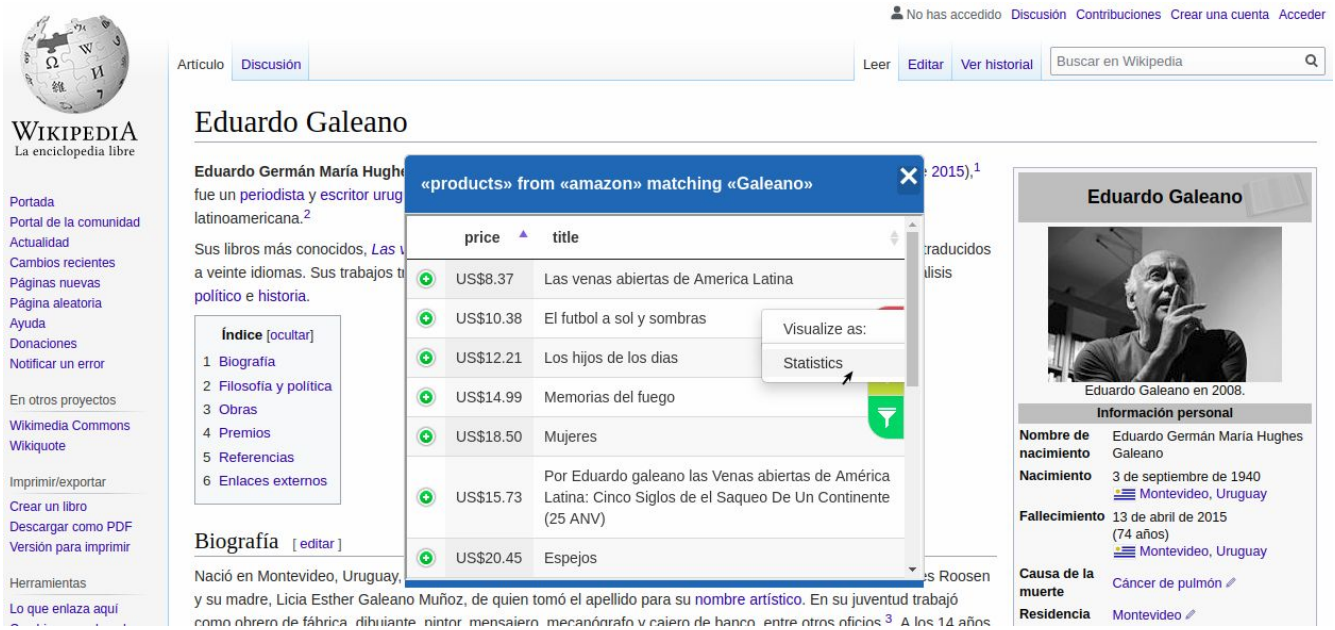


Figura 5.13: Menú para intercambiar de visualización

En este caso, continuando con el ejemplo anterior, a partir de los resultados de amazon el usuario puede visualizar la información en forma de gráfico estadístico (en este caso gráfico de barras), de los precios según el producto, ordenados de menor a mayor. Esta última visualización se observa en la Fig. 5.14.

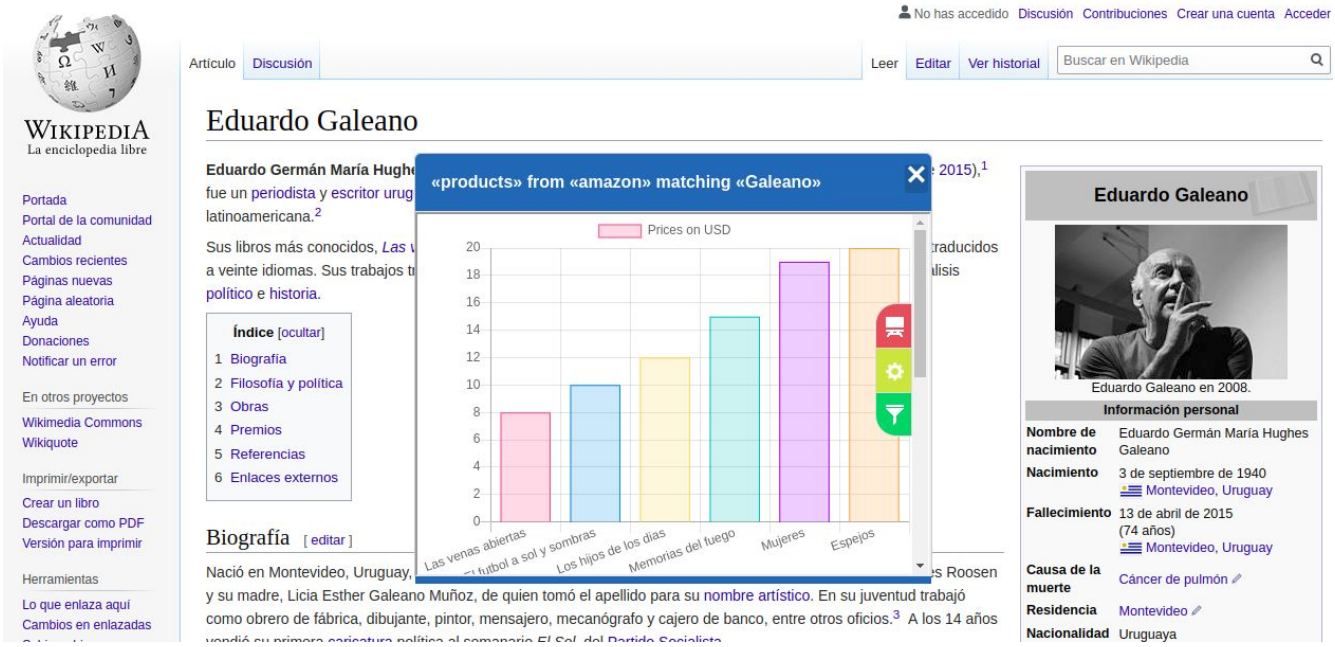


Figura 5.14: Visualización en forma de gráfico estadístico

De esta manera se demuestra como la herramienta da soporte al enfoque presentado, aunque cabe aclarar que la misma es una primera versión que demuestra la fiabilidad del mismo, pero que debe ser mejorada en muchos aspectos que se analizan en el siguiente capítulo. En este último capítulo, además, se realiza una evaluación de la mejora en los tiempos de búsqueda que representa utilizar la herramienta, respecto al método tradicional que se suele utilizar a la hora de realizar búsquedas en la Web.

Capítulo 6: Análisis y posibles mejoras

6.1 Introducción

Este capítulo final comienza con un análisis que valida cómo el enfoque mejora concretamente la experiencia del usuario final al momento de realizar las búsquedas Web mediante una evaluación cuantitativa basada en GOMS-KLM, aplicada al ejemplo presentado en el Capítulo 5 del actual trabajo.

Luego a partir de los los avances ya presentados en la herramienta, se analizan cuáles son los aspectos que restan ser desarrollados para aprovechar la totalidad de las ventajas que ofrece el enfoque y que las mismas puedan dar soporte para la mayor cantidad de motores de búsqueda posibles. Entre estos aspectos se incluye la expansión para adaptarse a la mayor cantidad posible de buscadores, aspectos como paginación y el filtrado así como también mejoras relacionadas con las visualizaciones..

6.2 Evaluación cuantitativa

A partir del ejemplo utilizado en la presentación de la herramienta a continuación se realiza una evaluación cuantitativa de la tarea de búsqueda basada en el modelo GOMS-Keystroke (KLM). Este método permite medir la eficiencia en la interacción con cualquier software, a partir de un escenario puntual. Cada acción tiene un tiempo preestablecido que permite predecir la performance.

En este caso se compara el ejemplo presentado en el Capitulo 5 contra la búsqueda tradicional, como se realizaría sin usar la herramienta propuesta. La tarea de búsqueda se divide en los siguientes pasos:

1. Acceder al artículo de Wikipedia (en este caso del escritor Eduardo Galeano)
2. Seleccionar el nombre del escritor
3. Realizar la 1ra búsqueda auxiliar (nypl)
4. Seleccionar el nombre de un libro (en el ej “Las venas abiertas de américa latina”)
5. Realizar la 2da búsqueda auxiliar (Amazon)

Los pasos que cambian al utilizar los servicios de búsqueda son al momento de realizar ambas búsquedas auxiliares (paso 3 y 5). Por lo tanto se comparan solo estos, contra el método tradicional que implica acceder a los respectivos sitios Web (amazon y nypl) para realizar las respectivas búsquedas. A partir de esta comparación se llega a la conclusión que cada búsqueda auxiliar utilizando los servicios de búsqueda, requiere 1.5 segundos desglosados en las siguientes acciones:

- Click mouse button para abrir menú contextual del navegador: 0.2 segundos.
- Apuntar con el mouse un objetivo en la pantalla (seleccionar el servicio de búsqueda de la lista): 1.1 segundos
- Click para disparar la búsqueda: 0.2 segundos

En cambio con la forma tradicional una vez que el usuario tenga seleccionado el texto que desea buscar, debe abrir una nueva ventana o pestaña, ingresar la URL, pegar el texto en el input y disparar la búsqueda. Todo esto puede involucrar en promedio casi 16 segundos. Si el sitio Web en cuestión se encuentra en marcadores se puede ahorrar un tiempo prudencial. Se puede notar que es mucha la variación, pero incluso considerando el mejor escenario, como mínimo el usuario, una vez que selecciono el texto que desea buscar (ej. Eduardo Galeano) debe:

- Pasaje mouse a teclado: 0.4 segundos.
- Copiar texto (Ctrl C): 0.2 segundos
- Pasaje teclado a mouse: 0.4 segundos.

- Apuntar con el mouse a un objetivo (suponiendo que la página Web esta en marcadores del navegador): 1.2 segundos
- Click para abrir la página : 0.2 segundos
- Apuntar con el mouse a un objetivo (input UI del motor de busqueda): 1.2 segundos
- Pasaje mouse a teclado: 0.4 segundos.
- Pegar texto (Ctrl V): 0.2 segundos
- Pasaje teclado a mouse: 0.4 segundos.
- Apuntar con el mouse a un objetivo (ventana o pestaña para volver al sitio de origen): 1.2 segundos

Este caso lleva la tarea a 5.8 segundos (casi 4 veces más que con el enfoque de servicios de búsqueda), es decir considerando el mejor caso, la búsqueda auxiliar implica mayor tiempo, lo cual lógicamente se incrementa si el usuario tiene que abrir una pestaña y tipear la url del sitio Web u otras consideraciones que pueden tardar más según la persona.

Además a mayor cantidad de búsquedas auxiliares requeridas mayor será el tiempo de deferencia. En la tarea puesta como ejemplo se realizan 2 búsquedas, que con el enfoque actual demoran un total de 3 segundos, mientras que de la manera tradicional (en el mejor de los casos) casi 12 segundos. Otro aspecto que debe ser considerado es si, eventualmente, el usuario desea comparar resultados entre sí obtenidos a partir de diversas búsquedas auxiliares o incluso entre los resultados y la información del contexto. En este caso el proceso resulta aún más ágil con el nuevo enfoque ya que el usuario evita moverse entre ventanas o pestañas para realizar dicha comparación.

Este análisis evidencia que una vez definidos los servicios de búsqueda por parte del usuario, puede resultar muy beneficiosa su frecuente utilización.

6.3 Posibles mejoras

Existen varios aspectos en los que se puede ahondar a la hora de buscar posibles mejoras del enfoque, asi como tambien de la herramienta que brinda soporte al mismo.

En primera instancia se debe considerar que las interfaces de los motores de búsqueda están construidas con diferentes componentes UI y pueden requerir diferentes tipos de interacciones. En la demostración de la herramienta se da soporte a los motores de búsqueda de Amazon y de nypl. Aunque también se comprobó que la definición de servicios de búsqueda muestra una buena adaptación en sitios e-commerce de los mas populares, ademas de Amazon, como por ejemplo e-bay. El desafío está en cubrir la mayor cantidad de motores de búsqueda, y que estos sean de diversos dominios (por ejemplo se pueden incluir redes sociales). Además que la herramienta a partir de los resultados en estos sitios debiera poder adaptarse a otros tipos de propiedades que no sean texto o numéricos, como por ejemplo imágenes. Esto último también abre la posibilidad de ofrecer nuevas formas de visualizaciones. A continuación vemos un ejemplo de una persona navegando en un sitio Web de un congreso y que decide ver imagenes de la ciudad donde se realiza el mismo. Para ello dispara una búsqueda a Pinterest y obtiene las imágenes para visualizarlas como se observa en la Fig.6.1.

Además respecto a este último aspecto, no solo se debe buscar posibles nuevas formas de visualización que mejoren la experiencia del usuario, sino que al momento de la definición del servicio, se debe proveer diversos mecanismos para lograr un mayor grado de personalización. Es decir además de que el usuario seleccione las propiedades que luego se recuperan de un objeto, también debe tener mayor injerencia en cómo se va a renderizar esa información. Por ejemplo, ofrecer diferentes tipos de gráficos estadísticos y que pueda elegir con qué propiedades generarlos, o lo mismo con las imágenes, que pueda seleccionar entre diversas formas de visualizar imágenes.

Figura 6.1: Visualización de imágenes

Otro aspecto que se puede agregarse, es ofrecer al usuario final, en el paso de definición del servicio, agregar determinados filtros o mecanismos de ordenamiento. Es decir que los filtros que ofrece el motor de búsqueda puedan agregarse por defecto en las futuras búsquedas al nuevo servicio, de manera que retorne los resultados filtrados según el criterio que el usuario determina al momento de la definición. La misma idea aplica a los mecanismos de ordenamiento.

Por último, en relación a la UI, resulta determinante que la herramienta se adapte a diferentes tipos de paginación que poseen las interfaces UI a la hora mostrar los resultados. De esta forma se podrá obtener todo el conjunto de resultados y no solo la primer página.

Otro aspecto relevante es el de la colaboración, resultaría útil mejorar las técnicas y encontrar nuevos mecanismos que permitan compartir los servicios definidos por otros usuarios, logrando una herramienta más ágil y fomentando su uso.

Como se puede observar son múltiples los aspectos del enfoque actual que aún restan por ser explotados, pero las bases están sentadas, e incluso se puede analizar la idea de aplicar el mismo enfoque en plataformas móviles, con todo el desafío que ello implicaría.

Bibliografía

[1] Cava R, Freitas CMDS, Barboni E, Palanque P, Winckler M (2014) Inside-In Search: An Alternative for Performing Ancillary Search Tasks on the Web.

[2] Anne Aula, Natalie Jhaveri, Mika Käki. (2005). Information search and re-access strategies of experienced web users. In Proceedings of the 14th international conference on World Wide Web.

[3] Daniel Rose, Danny Levinson (2004). Understanding Goals On Web Search.

[4] G. Marchionini (2006). Exploratory Search: From Finding To Understanding.

[5] Vicki L. O'Day and Robin Jeffries (1993). Orienteering in an information landscape: how information seekers get from here to there.

[6] Yates, R.B., Neto, B. R. (2011). Modern Information Retrieval: The Concepts and Technology behind Search (2nd Edition).

[7] Winckler M, Cava R, Barboni E, Palanque P, Freitas C (2015) Usability Aspects of the Inside-in Approach for Ancillary Search Tasks on the Web.

[8] McCallum A, Nigam K, Rennie J, Seymore K (1999). A Machine Learning Approach to Building Domain-Specific Search Engines.

[9] Morris D, Morris MR, Venolia G (2008) SearchBar: a search-centric web history for task resumption and information re-finding.

[10] Firmenich S, Bosetti G, Rossi G, Winckler M, Barbieri T (2016) Abstracting and Structuring Web Contents for Supporting Personal Web Experiences.

[11] Price, M., Crumley-Branyon, J., Leidheiser, W., Pak, R (2016). Effects of Information Visualization on Older Adults' Decistudy.

[12] Gabriela Bosetti, Sergio Firmenich, Marco Winckler, Gustavo Rossi, Ulises Cornejo Fandos, Egyed-Zsigmond (2018). An End-User Pipeline for Scraping and Visualizing Semi-Structured Data over the Web

[13] Pantazos, K., Kuhail, M., Lauesen, S., Xu, S (2013). uVis Studio: an integrated development environment for visualization.

[14] Emilio Ferrara, Pasquale De Meob, Giacomo Fiumarac, Robert Baumgartnerd (2014). Web Data Extraction, Applications and Techniques: A Survey

[15] N. Dalvi, P. Bohannon, and F. Sha (2009). Robust Web extraction: an approach based on a probabilistic tree-edit model.

[16] N. Dalvi, R. Kumar, and M. Soliman (2011). Automatic wrappers for large scaleWeb extraction.

[17] K. Dave, S. Lawrence, and D. Pennock (2003). Mining the peanut gallery: opinion extraction and semantic classification of product reviews.

[18] J. Myllymaki, and J. Jackson (2002). Robust Web Data Extraction with XML Path Expressions

[19] N. Dalvi, R. Kumar, and M. Soliman (2011). Automatic wrappers for large scale web extraction.

[20] Teixeira, J., Barata, G., Gon¸calves, D (2012). Meatbrain: Web information extraction and visualization.

[26] 15. Díaz, O., Arellano, C (2015). The augmented web: rationales, opportunities, and challenges on browser-side transcoding. ACM Transactions on the Web

[27] Lydia B. Chilton, Robert C. Miller, Greg Little, and Chen-Hsiang Yu. (2010). Why we customize the web. In No Code Required: Giving Users Tools to Transform the Web.

[28] Niels Olof Bouvin (2002). Augmenting the web through open hypermedia. The New Review of Hypermedia and Multimedia

[29] Lance Whitney (2009). Average net user now online 13 hours per week.

[30] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design patterns: elements of reusable object-oriented software. Pearson Education.

[31] Paul Dourish (2003). The appropriation of interactive technologies: Some lessons from placeless documents.

[32] T. V. Raman (2009). Toward 2W, beyond web 2.0. Communications of the ACM.

[33] Card SK, Mackinlay JD, Shneiderman B (1999). Focus+ context. In

[34] Shneiderman B (1996). The Eyes Have It:A Task by Data Type Taxonomy for Information Visualizations.

[35] Wilson TD (1999). Models in information behaviour research.