Programas de código libre (GNU) para la recuperación de información

Carlos G. Figuerola
José Luis Alonso Berrocal
Ángel F. Zazo
Emilio Rodríguez
Grupo REINA. Universidad de Salamanca (España)

Resumen

Los programas y aplicaciones informáticas de código libre pretenden superar los problemas derivados de la dependencia tecnológica que producen los programas llamados "propietarios". Frente a la concepción de programa perteneciente a una determinada empresa, de copia restringida, cuya utilización está sujeta a limitaciones (número de usuarios, ubicación física, etcétera), surgieron hace ya algún tiempo las ideas de *código abierto* y *software libre*. En contra de determinados augurios, la posibilidad de modificación del código fuente ha producido, además, una continua mejora y un perfeccionamiento de los programas, así como un aumento de prestaciones y complementos. El hecho es que actualmente los programas de código abierto tienen una solidez técnica importante, que ya quisieran para sí muchas aplicaciones propietarias.

En el campo documental son numerosos los programas de código abierto existentes, especializados en diversas tareas, además de gran cantidad de utilidades de propósito general aplicables en el quehacer documental. Este trabajo se centra en los programas y aplicaciones dedicados a la recuperación de información. El objetivo es hacer una revisión de las aplicaciones de código abierto dedicadas a la recuperación documental; esta revisión no busca la exhaustividad, sino un análisis con cierta profundidad de los programas más importantes en este campo.

Para ello, tras efectuar una descripción pormenorizada del concepto de código abierto, se ha llevado a cabo una selección de programas, fundamentalmente con base en su estabilidad técnica y en el grado de uso y difusión.

Palabras clave: Recuperación de información. Código abierto. Código libre. GNU.

Abstract

Free software tries to overcome the problems caused by the technological dependency that results from the use of "proprietary" software. The ideas of *free software* and *open source* arose some time ago in opposition to the conception of software as owned by a company, whose use is subjected to various restrictions affecting to copies, number of users, physical location, etc. On the contrary, the basic characteristics of open code are free use, free copy and distribution, free access to the source code; and, consequently, freedom of modification, improvement and adjustment. Free access to the source code has made possible the portability of this type of software to different physical architectures and different operating systems; so, although this is not formally a requirement for a programme to be considered open source, the fact is that most of open source programmes are characterized, among other things, by their portability among platforms. In the information field there are lots of free software programs suited for several specific tasks. This paper deals with free software specialized in information retrieval. Our target is to analyze the most frequently used programs in this field.

Keywords: Information retrieval. Open source. Free software. GNU.

1. El software libre

En los años sesenta y setenta era frecuente compartir programas entre grupos de programadores. El concepto de usuario final era radicalmente distinto del que ahora tenemos. Los usuarios de ordenadores eran informáticos muy especializados, y el trabajo en este ámbito consistía básicamente en programar.

En los primeros ochenta se asienta la idea de programa como algo distinto del propio ordenador y, junto a ella, la tendencia a restringir la copia y distribución de programas. Los programas se venden separadamente de la máquina y adquieren valor comercial propio. No es ajena a esta tendencia la aparición y rápida difusión de los ordenadores personales y de los programas llave en mano, dirigidos a auténticos usuarios finales. Las empresas blindan el código fuente de los programas distribuyendo solo los binarios ya compilados, imponiendo licencias restrictivas e incluso muy restrictivas (por ejemplo, en algunos casos se prohíbe la mera copia de seguridad del programa).

En 1984, Richard Stallman, que entonces trabajaba en el célebre MIT, fue el primero en formalizar la idea de software libre, en contraposición a esa tendencia restrictiva comentada antes (Williams, 2002). Formuló lo que se conoce como *las cuatro libertades del software* y emprendió una gran actividad es este campo. Fruto de ella fue la Free Software Foundation (http://www.fsf.org), un organismo destinado a promover el software libre. En la actualidad la FSF tiene una presencia

muy activa, con innumerables campañas y actividades, y posee filiales en muchos países.

Stallman inició también el proyecto GNU (http://www.gnu.org). GNU es un acrónimo recursivo (Gnu No es Unix), una pequeña broma que intenta reafirmar la idea de interoperabilidad. Su principal finalidad era la creación de un sistema operativo libre que debería llevar el nombre de HURD. Aunque este objetivo no está cumplido, ni parece que lo vaya a estar a corto plazo, el proyecto GNU ha dado y sigue dando muchos otros resultados, como la elaboración de distintas clases de licencias para el software (y también para otro tipo de creaciones intelectuales). Entre ellas se encuentra la General Public License o licencia GPL, que es desde hace tiempo un verdadero punto de referencia en este campo, y cuyo objetivo es especificar y proteger las libertades del software.

De otro lado, el proyecto GNU ha producido literalmente centenares de programas de todo tipo y para realizar toda clase de tareas. Algunos son programas completamente nuevos y otros son reescrituras de programas preexistentes. Todos, por ser software libre, se distribuyen como código fuente, compilable y ejecutable en cualquier plataforma hardware. En su gran mayoría son programas que, al llevar tiempo siendo utilizados, estudiados, corregidos y vueltos a utilizar, tienen una solidez técnica envidiable.

El proyecto de sistema operativo HURD, por otra parte, pasó por diversas fases de estancamiento, aunque actualmente se sigue trabajando en él. Sin embargo, el surgimiento de Linux vino a resolver en cierta medida este problema. En 1991, Linus Torvald, entonces estudiante, escribió el núcleo o *kernel* de un sistema operativo y lo publicó en la Red. Inmediatamente muchos programadores se dieron cuenta del potencial de dicho *kernel* y se dedicaron a escribir un sinfín de programas que completaran y complementaran ese núcleo. Así surgió lo que hoy conocemos como *Linux*. El proyecto GNU adoptó Linux como sistema operativo, si bien de forma temporal, puesto que se sigue trabajando en HURD.

En 1994, E. S. Raymond, B. Perens y otros fundaron la OSI (Open Software Initiative, http://www.opensource.org), cuya finalidad es impulsar el desarrollo de programas y aplicaciones de código abierto, es decir, que sea posible estudiarlos, modificarlos, corregirlos, etcétera (Raymond, 2001). La perspectiva de la OSI es menos filosófica que las de la FSF y la GNU, en el sentido de que no se enmarca en una determinada visión de la sociedad. Simplemente, la OSI entiende que la disponibilidad y libre copia, estudio y modificación del código promueve o facilita la excelencia técnica de los programas, puesto que los somete a un ciclo continuo de uso, análisis y corrección, lo que los libera de errores y los mejora notablemente.

Las diferencias ideológicas entre ambos planteamientos hacen que unos prefieran la expresión *código abierto* y otros *software libre*. Sea con una perspectiva

o con otra, el hecho es que desde hace años los programas de este tipo son abundantes y gozan de una solidez técnica notable, y constituyen una opción perfectamente válida frente a soluciones comerciales.

En el campo de los sistemas de recuperación de información existen abundantes programas de código abierto o libre disponibles (Morgan, 2004). Las características son muy variadas, y algunos son muy conocidos y usados, mientras que otros son realmente exóticos. En este trabajo hemos seleccionado cuatro de tales programas atendiendo en primer lugar a sus capacidades, pero también a nuestro conocimiento directo de ellos, es decir, al hecho de haber trabajado con ellos de una forma u otra. No son, ni lo pretendemos, necesariamente los mejores, pero en todo caso se trata de programas potentes, sólidos y estables, que pueden ser utilizados perfectamente para implementar un sistema de recuperación documental. Al tratarse de software libre, todos ellos pueden descargarse, utilizarse y copiarse sin problemas.

2. DataPark (http://www.dataparksearch.com)

Se trata de un programa poco conocido pero notable en cuanto a sus capacidades. Está diseñado para indizar sitios web, aunque es posible utilizarlo también para indizar otro tipo de documentos, no necesariamente situados en la Red. En consecuencia, el programa consta de un *spider* o robot (Figuerola et álii, 2005), para recolectar las páginas web de los sitios a indizar, un módulo de indización y otro de búsqueda.

Como es habitual con los programas de código libre, se suministra el código fuente, así como instrucciones y *scripts* para configurar, compilar e instalar de forma automática. Para ello es necesario disponer de las herramientas habituales (*gunzip*, *tar*, *autoconf*, *make*, *gcc*...). Además de estas, DataPark requiere una base de datos SQL, donde índices y documentos son almacenados. Una larga lista de bases de datos es compatible con DataPark, entre ellas PostGres, MySQL, SQLite y otras, incluidos también algunos productos comerciales (Oracle, por ejemplo).

Adicionalmente necesitaremos tener instalado un servidor web, ya que las búsquedas se efectúan a través de una interfaz web.

2.1. Indización

El *spider* o robot es extraordinariamente flexible y configurable en cuanto a extremos como los URL que ha de seguir y los que ha de indizar, ya que no son necesariamente coincidentes; también permite ajustar parámetros como tiempos de espera, gestión de errores, respeto al estándar de exclusión, etcétera.

Admite también la indización de documentos locales, directamente a través del sistema de ficheros, y soporta documentos con una gran variedad de códigos de caracteres, entre ellos UTF-8, por lo que resulta adecuado para colecciones de documentos multilingües.

Internamente está preparado para indizar documentos HTML y en texto plano (además de algunos otros un poco más exóticos, como MP3), pero también lo está para conectar de forma automática con *parsers* o conversores externos. Muchos de estos *parsers* son también software libre, por lo que no debería haber problemas para conseguirlos. Esto permite indizar casi cualquier formato de documento convertible a texto.

Uno de los puntos fuertes de este programa es el abundante conocimiento lingüístico que incorpora. Admite el uso de listas de palabras vacías y aporta repertorios para una gran cantidad de lenguas. También es posible definir listados de acrónimos, sinónimos y abreviaturas. Si se desea se pueden utilizar las reglas y diccionarios de *ispell* (http://ficus-www.cs.ucla.edu/geoff/ispell.html) para lematizar las palabras; *ispell* es también software libre y dispone de información lingüística para muchos idiomas.

2.2. Búsquedas

DataPark permite buscar por palabras o términos pero también admite el uso de operadores booleanos. De otro lado, puede analizar la estructura de un documento HTML y pesar o dar más importancia a unas partes que a otras. Por lo que se refiere a los resultados de una búsqueda, es posible ordenarlos en función de varios criterios; igualmente, el programa es capaz de producir un pequeño resumen o *snippet* de cada página o documento encontrado.

Las búsquedas se efectúan a través de una página web que llama o ejecuta un programa CGI al cual le pasa la ecuación de búsqueda. El programa viene con dos páginas de búsqueda (una para búsquedas sencillas y otra para avanzadas), que pueden ser modificadas y personalizadas fácilmente.

2.3. Documentación

El programa tiene su propio manual, de unas cincuenta páginas, en inglés (hay también una versión en ruso). El manual está bien organizado y es completo, aunque en algunos aspectos se echan en falta más ejemplos de uso. Además, los autores del programa mantienen un foro de mediana actividad, así como un wiki muy poco activo.

El principal problema de este programa es su escasa base de usuarios. Se utiliza en pocos sitios, a pesar de su potencia, por lo que todavía no se tiene mucha experiencia en cuanto a su uso; a su vez, esto hace que haya poca información disponible, y la que existe se limita prácticamente a la documentación oficial del programa.

3. Lucene y Nutch (http://lucene.apache.org/nutch)

Lucene no es exactamente un programa, sino una librería, es decir, una serie de herramientas y componentes para construir motores de búsqueda. Forma parte

del proyecto Apache, al igual que el conocido servidor web, y es la base de otros muchos proyectos actualmente en marcha, por lo que es previsible que en el futuro veamos bastantes sistemas de recuperación basados en Lucene.

3.1. Indización

La librería, diseñada originalmente en Java, ha sido portada a varios otros lenguajes. El diseño original se hizo pensando en economizar costes computacionales; esto significa que consume poca memoria, pero al mismo tiempo indiza con rapidez, y produce índices que ocupan poco espacio en el disco. Es posible, por otra parte, hacer indización incremental: no es necesario reconstruir la totalidad de los índices cada vez que se añade un nuevo documento.

Por lo que se refiere al análisis léxico, Lucene incorpora rutinas para extraer términos de diversos formatos de documento (HTML, PDF, MS Word...), pero puede ser conectado también con conversores externos. Maneja los juegos de caracteres más habituales, entre ellos Unicode, y admite el uso de listas de palabras vacías y de sinónimos. Tiene rutinas específicas para conectar con lematizadores basados en SnowBall (http://snowball.tartarus.org) y con redes terminológicas que se apoyan en WordNet (http://wordnet.princeton.edu), como EuroWordNet (http://nlp.shef.ac.uk/eurowordnet/eurowordnet.html), lo que abre la puerta a recuperaciones multilingües. Por otro lado, puede indizar por campos o partes de documentos, así como distinguir valores numéricos o fechas.

3.2. Búsquedas

Las búsquedas pueden efectuarse de diversas formas; así, están disponibles las sencillas, por términos, pero también por frases, por operadores booleanos o por rangos de fechas. Además, es posible limitarlas a uno o varios campos determinados, así como buscar por términos parecidos. Existen también varios algoritmos para la ordenación de los documentos encontrados tras una búsqueda.

3.3. Documentación

La documentación de Lucene es, sencillamente, excelente. Lo es la de tipo oficial, abundante y precisa, pero lo es también la abundantísima documentación extraoficial existente. Así, sobre Lucene hay varios libros disponibles (Hatcher y Gospodneti, 2004), así como numerosos artículos en revistas (White, 2006; Ferguson Smart, 2006). En la Red se pueden encontrar abundantes tutoriales, *howtos*, recetas, trucos, etcétera. Los creadores y mantenedores de Lucene mantienen un wiki bastante activo, así como un FAQ importante. El foro para desarrolladores está también bastante concurrido. En cualquier caso, la gran cantidad de usuarios y expertos en Lucene hace que sea fácil encontrar información sobre su uso en Internet, ya se trate de documentación de tipo general o de respuestas concretas a aspectos muy determinados o específicos.

La lista de sitios, centros de documentación, etcétera, que utilizan Lucene es, por otra parte, amplísima.

4. SWISH-E (http://swish-e.org)

Este es un programa histórico, en el sentido de que existe desde hace años y ha pasado por multitud de versiones. En muchas distribuciones Linux se incluye como paquete estándar, y puede decirse que es un clásico en este campo. Es también un todoterreno, puesto que ha sido ideado para indizar colecciones de documentos de muy diverso tipo y en diversas circunstancias (Rabinowitz, 2004).

El programa está diseñado para economizar recursos, tanto de memoria como de capacidad de proceso. Puede producir índices incrementales, es decir, que no es necesario reconstruirlos completamente cuando se añaden documentos nuevos, y también es capaz de operar con índices simultáneos en diferentes plataformas (Ruiz, 2002). De otro lado, ha sido probado y utilizado con éxito en numerosos sistemas operativos y arquitecturas diversas, incluido Windows.

4.1. Indización

En este sentido, recibe e indiza documentos procedentes de tres fuentes: documentos recibidos a través de la Web, generalmente recolectados por un *spider* o robot; documentos residentes en el sistema de ficheros local; y documentos entregados por un tercer programa (por ejemplo, aquellos que han sido previamente almacenados en una base de datos SQL o los recibidos por otros protocolos de red distintos de HTTP).

Como otros programas vistos, puede conectarse con conversores externos e indizar así gran cantidad de formatos (PDF, MsWord, etcétera), pero lo que hace realmente bien es indizar documentos en HTML y XML. En ambos casos es capaz de comprender la estructura del documento y tratar de forma diferente cada uno de los elementos, entre ellos las etiquetas META. Este tratamiento incluye pesar de manera distinta los diferentes términos de diversas partes del documento.

En lo que se refiere al análisis léxico, Swish-E maneja varios juegos de caracteres, entre ellos UTF-8, y permite establecer cuáles son los caracteres constitutivos de palabra y cuáles deben ser separadores. También ofrece la posibilidad de especificar caracteres equivalentes (por ejemplo, vocales con y sin acento) y de usar listas de palabras vacías. Además puede conectarse con lematizadores construidos con SnowBall. Permite aplicar algoritmos Soundex y Metaphone.

4.2. Búsquedas

Además de las consabidas búsquedas sencillas por términos, podemos usar frases, truncamientos, operadores booleanos y también campos (cuando trabajamos

con documentos HTML o XML), así como combinar todo esto entre sí. La sintaxis de Swish-E es intuitiva y potente.

La lista de documentos recuperados tras una búsqueda puede ordenarse en función de diversos criterios, e incluso mostrarse por rangos de posiciones en dicha lista, no necesariamente de forma correlativa (por ejemplo, es posible mostrar los documentos entre el puesto 10 y el 25). Esto permite paginar la lista si es especialmente larga. La propia lista es altamente configurable, no solo en lo referente al formato, sino también en lo que tiene que ver con qué parte(s) o qué información se ofrece para cada documento recuperado.

Aunque la interfaz básica del programa, tanto para indización como para búsquedas, es la línea de comandos en modo texto, su sencillez y su flexibilidad permiten empotrarlo fácilmente en interfaces o programas más amigables; lo más sencillo es utilizar como interfaz amigable una página web que, mediante un formulario, ejecuta el comando de búsqueda adecuado. Dado que el listado de resultados, como se ha comentado, es muy configurable, es fácil hacer que dicho listado tome la forma de página web.

Este es el uso más frecuente; de hecho, el programa viene acompañado de unos formularios web de ejemplo, que pueden utilizarse directamente o tomarse como modelos para construir una interfaz personalizada.

4.3. Documentación

La documentación oficial consiste en un amplio y completo manual, ayudado por un foro de correo, una lista regular de FAQ y un wiki moderadamente activo. Sin embargo, el verdadero punto fuerte de este programa (en lo que a documentación se refiere) es el gran número de usuarios que, a lo largo de los muchos años que lleva en funcionamiento, lo han aplicado en circunstancias muy diversas. Es realmente fácil encontrar en la Red gente que ha trabajado con él, y algunos son auténticos expertos.

5. HTDIG (http://htdig.org)

Sin tener tanta antigüedad como el anterior, se ha convertido también en un clásico, presente en muchas distribuciones Linux, en varias de las cuales se instala por defecto cuando se aplica una configuración estándar. Está pensado para indizar sitios web (por eso se instala por defecto cuando se configura una máquina como servidor web en muchas versiones de Linux), aunque no está limitado necesariamente a ese uso (Morgan, 2004).

5.1. Indización

Este programa se caracteriza porque obtiene los documentos que indiza a través de la Web. Mediante un potente *spider*, se conecta a un servidor web, obtiene

los documentos de este y los indiza. Naturalmente, los servidores y los documentos que se deben obtener e indizar son cosas configurables. Nada impide que el servidor web sea o esté en la misma máquina que HTDIG; pero el acceso debe realizarse a través del servidor HTTP y no a través del sistema local de ficheros.

Por lo demás, el programa, al estar pensado para indizar sitios web, se encuentra especialmente preparado para tratar documentos HTML, pero también otros formatos, como PDF. Igual que otros muchos, puede conectar con conversores externos, lo que facilita la indización de cualesquiera otros formatos. Permite aplicar las técnicas habituales de análisis léxico: configuración de caracteres constitutivos de palabra, tratamiento de acentos, listas de palabras vacías, sinónimos, lematización a través de las reglas y diccionarios *ispell*, etcétera.

5.2. Búsquedas

Las búsquedas se efectúan a través de un programa CGI que debe ser llamado por un formulario en una página web. Este programa produce un listado de documentos encontrados que es devuelto también como página web. Admite el uso de operadores booleanos y la aplicación de algoritmos como *soundex* o *metaphone*. Los resultados pueden ser ordenados en función de diversos criterios. El formato de este listado es totalmente personalizable.

5.3. Documentación

La documentación oficial es muy abundante, tanto la de tipo general como la correspondiente a cada una de las muchas directivas que el programa admite. Para cada una de ellas, además de la explicación pertinente, se ofrecen ejemplos. Este podría ser el principal problema de la documentación oficial del programa: su abundancia y fragmentación por directivas puede hacer algo dificultoso su manejo, a pesar de su navegabilidad.

Se proporciona un abundante FAQ con buenas explicaciones y se mantienen varias listas de correo bastante activas que pueden resultar de gran ayuda. De otro lado, la lista de usuarios es amplísima, lo que hace que sea fácil encontrar información en la Red tanto sobre el uso general como sobre aspectos específicos. Existen también varios artículos relativos a este programa (Brockmeier, 2002).

Se trata de un programa muy sólido que, pese a la gran cantidad de opciones y elementos configurables, es fácil de poner en funcionamiento. En efecto, viene con un modelo de fichero de configuración que puede usarse prácticamente tal cual; únicamente hay que poner el URL del sitio que se desea indizar. Solo es necesario modificar otras directivas cuando se desea un funcionamiento no estándar. Además, viene con páginas web de búsqueda y de resultados de ejemplo que pueden utilizarse directamente. De ahí la sencillez de su utilización, y probablemente también la gran difusión de la que goza.

6. Conclusiones

El software libre ha alcanzado un grado de madurez y solidez técnica notables, de manera que constituye hoy día una alternativa razonable al software comercial en muchos campos. Uno de estos campos es el de los sistemas de recuperación documental. Numerosos programas libres están disponibles para implementar estos sistemas; algunos de ellos lo están desde hace años y han pasado por sucesivas versiones. Muchos organismos públicos y privados, instituciones, fundaciones, bibliotecas, museos, universidades, centros de documentación, etcétera, han elegido programas libres para sus sistemas de recuperación. Aunque la lista de esta clase de programas es amplia, se han seleccionado algunos de los más conocidos y se han revisado sus características más señaladas.

Referencias

Brockmeier, Joe (2002). The Open Road: using htDig. // UnixReview. (Abril de 2002). http://www.unixreview.com/documents/s=2426/uni1019674163632/0204k.htm (2006-07-05).

Ferguson Smart, J. (2006). Integrate advanced search functionalities into your apps: implement powerful multi-criteria search criteria and filters with Lucene. // Java World. http://www.javaworld.com/javaworld/jw-09-2006/jw-0925-lucene.html (2006-07-05).

Figuerola, Carlos G.; Alonso Berrocal, José Luis; Zazo, Ángel; Rodríguez, Emilio (2006). Diseño de *spiders*. Informe técnico DPTOIA-IT-2006-002, Universidad de Salamanca, marzo de 2006.

Hatcher, Erik; Gospodneti, Otis (2004). Lucene in Action. Nueva York: Manning, 2004.

Morgan, Eric Lease (2004). Comparing Open Source indexers. // Infomotions Musings (Noviembre de 2004). http://www.infomotions.com/musings/opensource-indexers/ (2006-07-05). Creado en 2001. Actualizado en 2004.

Rabinowitz, Josh (2004). Indexing arbitrary data with SWISH-E. // USENIX 2004. Annual Technical Conference. Berkeley: Usenix.og, 2004. 199-205.

Raymond, Eric S. (2001). The cathedral & the bazaar: musings of Linux and Open Source by an accidental revolutionary. Sebastopol: O'Reilly & Associates, 2001.

Ruiz, José Manuel (2002). SWISH-E: un ejemplo de colaboración en el desarrollo del software libre. // VII Jornadas sobre Tecnologías de la Información en la Modernización de las Administraciones. A Coruña, 2002. http://www.csi.map.es/csi/tecnimap/tecnimap2002/pdfs/c1.3-26.pdf (2006-07-05).

White, Tom (2006). Introduction to Nucth (parts 1 & 2). // java.net: The source for the Java techonolgy collaboration. http://today.java.net/pub/a/today/2006/01/10/introduction-to-nutch-1.html (2006-05-05).

Williams, Sam (2002). Free as in freedom: Richard Stallman's crusade for free software. Sebastopol: O'Reilly & Associates, 2002.