Sistemas Integrales de Automatización de Bibliotecas: una descripción suscinta

Dr. Óscar Arriola Navarrete*
Lic. Evangelina Montes de Oca Aguilar**

Resumen

En este artículo se proporciona una descripción general de los Sistemas Integrales de Automatización de Bibliotecas, tanto propietarios como de Open Source, así como también conceptos, antecedentes, ventajas y desventajas de éstos.

Palabras clave: Sistemas Integrales de Automatización de Bibliotecas, Automatización, Software Propietario, Open Source

Abstract

This article provides an overview of the Integrated Library Systems both proprietary and Open Source, as well as concepts, history, advantages and disadvantages of these is provided.

Keywords: Integrated Library Systems, Automation, Software Owners, Open Source

Introducción

Las Tecnologías de Información y Comunicación (TIC) están inmersas en muchas actividades humanas, su uso y aprovechamiento son cada vez más comunes. Se busca aprovechar los beneficios que brindan las TIC para mejorar el entorno en el que vivimos, ya que con su buen uso se mejoran muchos procesos y servicios.

En la actualidad los desarrollos tecnológicos han sido explotados por instituciones, empresas, organizaciones, etc., con la finalidad de estar en continua actualización. Las bibliotecas también tienen que estar en constante evolución, aprovechando las TIC que permiten estar a la vanguardia y de esta forma brindar mejores servicios. Es por eso que el mercado de software para bibliotecas ha crecido en los últimos años de manera exponencial, y existe actualmente una gran diversidad de Sistemas Integrales de Automatización de Bibliotecas (SIAB).

Al existir tal variedad de software especializado para bibliotecas, arroja la pregunta, ¿cuál es el más adecuado para mi biblioteca?, los bibliotecarios tienen que tomar en cuenta las necesidades y objetivos de la biblioteca, así como también el presupuesto para

* Profesor Titular "C" de Tiempo Completo de la Escuela Nacional de Biblioteconomía y Archivonomía
** Estudiante de la Maestría en Bibliotecología y Estudios de la Información de la UNAM

determinar cuál es el sistema más adecuado. No siempre el software más avanzado y caro es el mejor, lo más importante es que cubra la mayor parte de las necesidades de la biblioteca y por consiguiente, ayude a mejorar los procesos y servicios, es importante también que cuente con posibilidad de actualizarse cada vez que se requiera y que tenga un soporte técnico eficiente. La demanda de información es cada vez mayor, los usuarios requieren de servicios de información ágiles y de calidad; para que las bibliotecas puedan brindar estos servicios necesitan explotar lo más posible los beneficios que brinda un SIAB.

Concepto

De manera general un sistema es un conjunto de elementos con relaciones de interacción e interdependencia que forman un todo unificado. Un sistema integral de automatización de bibliotecas está compuesto por varios elementos llamados módulos, lo cuales se encuentran diseñados con el objetivo de ayudar a desempeñar de mejor forma las diferentes actividades que realiza una biblioteca.

Ya dentro de la conceptualización que existe en la literatura bibliotecológica usada para definir un Sistema Integral de Automatización de Bibliotecas se tiene en primer lugar lo dicho por García Melero: "un sistema automatizado de bibliotecas es un conjunto organizado de recursos humanos que utilizan dispositivos y programas informáticos, adecuados a la naturaleza de los datos que deben procesar, para realizar procesos y facilitar los servicios que permiten alcanzar los objetivos de la biblioteca: almacenar de forma organizada el conocimiento humano contenido en todo tipo de materiales bibliográficos para satisfacer la necesidades informativas, recreativas y de investigación de los usuarios" (García Melero. 1999. p. 24).

Se considera que la definición anterior es confusa, ya que inicia definiendo lo referente a recursos humanos; el personal bibliotecario es el que utiliza los sistemas de automatización pero no son parte de ellos. Esta definición también describe la "utilización de programas informáticos para realizar los procesos y facilitar los servicios", estos elementos son importantes para describir un SIAB.

Por otro lado, tenemos lo que comenta Félix Moya, lo define como "*sistemas para el proceso automatizado o informático, de información estructurada y no estructurada, sobre actividades y documentos, adaptable a la estructura organizativa de la biblioteca*" (Moya Anegón. 1995. p. 25).

Esta definición comenta que un sistema esta "estructurado a las actividades y documentos de una biblioteca" y también que es "adaptable a la organización de bibliotecas", con esto se acerca un poco más al concepto de SIAB, pero se considera que todavía le faltan aspectos importantes que puntualizar.

Lourdes David dice que un sistema integral de bibliotecas "*es aquel que tiene una base de datos en común para realizar todas las funciones básicas de una biblioteca. Un sistema integrado de bibliotecas permite a la biblioteca vincular las actividades, por ejemplo la circulación con la catalogación, gestión de publicaciones seriadas, etc., en un momento dado. Hace uso de un servidor de archivos y clientes en una red de área local. La mayoría de los sistemas de gestión de bibliotecas tienen los siguientes módulos: catalogación y OPAC, circulación, adquisiciones, gestión de publicaciones seriadas y el módulo de préstamo interbibliotecario*" (David. 2001).

Oscar Arriola indica que un sistema integral de bibliotecas es un *"conjunto de módulos de aplicación integrados en un solo programa y que comparten una base de datos bibliográfica en común y que ayuda a la gestión de procesos y servicios de las unidades de información"* (Arriola Navarrete y Butrón Yáñez. 2008).

Las definiciones anteriores son más completas y manejan cosas en común, ya que las dos mencionan que un SIAB es un "conjunto de módulos que comparten una base de datos en común", pero se explica con mayor detalle las relaciones de integración en la definición de Lourdes David.

Cabe señalar que al hablar de un sistema de automatización de bibliotecas y de un sistema integral de automatización de bibliotecas, este último queda inmerso en el primero y la diferencia que puede haber entre ambos conceptos es que el sistema integral es cuando se tiene un conjunto de módulos integrados y en un sistema de automatización en general entran los sistemas integrados y no integrados, esto porque los primeros sistemas de automatización que surgieron no eran integrales, aunque en la actualidad esto ha cambiado.

Tomando en cuenta los elementos que aportan cada una de las definiciones anteriores, se entenderá que un sistema de automatización de bibliotecas es aquel que posee un conjunto de módulos que abarcan las actividades bibliotecarias más importantes, los cuales están relacionadas entre sí, ya que comparten una misma base de datos, aunque dichos módulos tienen funciones distintas están unificados para facilitar su control, y de ésta manera ayudar a mejorar la eficiencia y eficacia de los procesos, servicios y de la gestión general de la biblioteca.

Un elemento importante dentro del concepto de un SIAB, son los módulos, un módulo es un subprograma dentro de un programa que permite la aplicación específica sobre un área determinada para gestionarla dentro de un programa integral.

En el siguiente esquema se trata de clarificar este concepto:

Esquema 1. Estructura del SIAB

Antecedentes

Los sistemas integrales son una evolución de sus antecesores, los sistemas monofuncionales que se utilizaron a finales de los años setentas. Los SIAB como su nombre lo dice, integran en un solo programa en módulos todas las áreas necesarias para la gestión de una biblioteca como son: circulación, adquisición, catalogación, etc. En este tenor Arriola Navarrete menciona que el término integración indica multifuncionalidad, un sistema que recoge todas las funciones necesarias para la gestión de cualquier biblioteca. Por otra parte, un sistema de esta clase también se caracteriza porque en él existe una integración a nivel de datos, de manera que la información se almacena para el uso compartido y específico de cada módulo funcional.

Los sistemas integrales de automatización de bibliotecas han tenido un gran avance, siendo ahora más fáciles de usar tanto para el usuario como para el bibliotecario. La historia que hay sobre los sistemas integrales se remonta a los años ochenta, ya que es en esta década donde se consolidan.

Para lograr la integración de las tareas bibliotecarias en un solo sistema compartiendo una misma base de datos, se desarrollaron varios intentos. Uno de los primeros esfuerzos sistemáticos en la automatización de bibliotecas comenzó en la University of Chicago en 1965, cuando el Dr. Herman H. Fussler, director de la Biblioteca de la University of Chicago, presentó una propuesta a la Fundación Nacional de Ciencias para desarrollar un enfoque integrado, basado en bases de datos bibliográficas en computadora (Goldstein. 1983).

A mediados de la década de los setentas se desarrollaron algunos proyectos de sistemas integrales como: la University of California, San Diego, con el control automatizado para publicaciones periódicas; Southern Illinois University, Carbondale, con el sistema de circulación automatizada, y la Ontario New University Library, con el catálogo de libros en computadora. En Inglaterra, la biblioteca de la University of Newcastle con File Handling System (NFHS), que se utilizó para la adquisición, y la University of Southampton, que desarrolló un sistema automatizado para el control de la circulación de los fondos. Dichos proyectos fueron entorpecidos ante la falta de comunicación entre el personal especializado, la deficiencia de los sistemas de cómputo y una inapropiada comunicación entre bibliotecarios y el personal de cómputo (Arriola Navarrete y Butrón Yañez. 2008. p.4). La OIT (Organización Internacional del Trabajo), a finales de 1960, desarrolló un juego de programas informáticos llamados ISIS (Integrated Set for Information System), diseñados para una computadora mainframe, en una IBM 360-30, que corría bajo el sistema operativo DOS. Con dicho programa se gestionaba el complejo archivo de la OIT, así como los extensos fondos documentales de la misma. La OIT cedió dicho paquete informático a cuantas instituciones relacionadas con su actividad se lo solicitaron.

Al mismo tiempo que se creó ISIS, la UNESCO desarrolló otro sistema de gestión bibliográfica llamado CDS (Computarized Documentation System), que instaló en su Mainframe, una ICL de la serie 1900. La falta de normalización característica de los equipos informáticos de la época, hacían incompatibles ambos sistemas, que fueron utilizados por ambas organizaciones de forma independiente, hasta que en 1975 la UNESCO decidió renovar sus equipos informáticos y pasar de ICL a IBM.

En dicha adaptación se produjo una unión entre ambos sistemas, CDS de la UNESCO e ISIS de la OIT dando lugar a

un nuevo producto que se denominó CDS/ISIS, diseñado para *mainframes* con sistema operativo MVS.

La primera versión de MICROISIS (nombre más extendido entre los países de habla hispana de la versión CDS/ISIS para computadoras personales) corrió por primera vez en un equipo IBM PC-XT de 150 Kb de memoria y 10 Mb de disco duro y fue presentada en una reunión de usuarios de la versión Mainframe celebrada en 1985 en Buenos Aires. MICROISIS fue en realidad un conjunto de seis programas independientes que manejaban archivos comunes. Así para pasar de una función a otra, había que cerrar un módulo y abrir otro. Además, el programa se servía "tal como era", es decir, el programa hacia lo que hacía y no permitía que el usuario pudiese diseñar aplicaciones particulares. Era lo que se conocía como un paquete cerrado (MicroISIS).

Dos hechos ocurrieron, que sin duda, dieron el más grande efecto en la automatización de las funciones de las bibliotecas en la década de 1970, el establecimiento de Ohio College Library Center (OCLC), nombrado inicialmente así por sus creadores y posteriormente denominado Online Computer Library Center en 1967, seguida de la adopción del formato MARC II y la distribución de servicios de la Library of Congress en el año 1968. Estos dos acontecimientos dieron alternativas viables de automatización local a las bibliotecas. Ésta alternativa de tener acceso a los recursos en red, permitió que el costo de automatización fuera compartido por muchos. El compartir la catalogación en OCLC, esta actividad iba a ser, de hecho, la primera parte de un sistema integrado. Todo el sistema, incluyendo catálogo compartido, recuperación de información bibliográfica, control de circulación, control de las publicaciones seriadas y procesamiento técnico, se basó en un archivo, consiguiendo de ésta manera un sistema verdaderamente global (Goldstein. 1983).

Uno de los proyectos que pueden aprovecharse a través de los SIAB es el WorldCat, tiene una de las bases bibliográficas más grandes del mundo. Las decenas de millones de registros de WorldCat representan los recursos físicos y digitales que poseen las bibliotecas y que trascienden cualquier tema, idioma y cultura. A partir de ésta base de datos se pueden intercambiar registros bibliográficos o apoyar servicios como el préstamo interbibliotecario, entre otros.

Los sistemas integrales de automatización de bibliotecas actuales están diseñados para facilitar la gestión de las bibliotecas a través de sus módulos, que abarcan las diferentes actividades que se llevaban a cabo dentro de ellas. Como se mencionó anteriormente, los SIAB ayudan a que las bibliotecas puedan compartir recursos a nivel mundial. Existen numerosos sistemas integrales de automatización extranjeros, pero también hay sistemas locales que compiten en el mercado. Algunos de los sistemas propietarios tienen costos elevados y a veces dificulta su adquisición para ciertas bibliotecas. Por esta y otras razones es que ha surgido el software libre, que es otra opción con que cuentan las bibliotecas para establecer un sistema integral cuando no hay mucho presupuesto para ello. En los siguientes apartados de este artículo, se profundizará más sobre las características de éstas dos opciones que son el software propietario y el software libre.

Características

Independientemente de la opción que se utilice para automatizar una biblioteca, ya sea software propietario o software libre,

éste debe tener ciertas características que garanticen que es confiable su uso. Algunas de las características que debe tener el software son (Open Source Software. 2005):

- *Fiabilidad*: se define como el tiempo que un sistema puede permanecer en operación sin intervención del usuario.
- *Calidad:* comúnmente se define como el número de errores en un número fijo de líneas de código.
- *Seguridad:* lo resistente que el software es para no autorizar acciones fuera de protocolo (por ejemplo, virus).
- *Flexibilidad:* la facilidad con que el software puede ser personalizado para satisfacer las necesidades específicas y que se pueden ejecutar en diferentes tipos de dispositivo.
- *Gestión de proyectos*: La facilidad de organizar los proyectos en desarrollo.
- *Estándares abiertos*: los documentos creados con un tipo de software deben de ser leídos y trabajados en cualquier software.
- *Los costos de cambio*: el costo de pasar de un sistema a otro.
- *Costo Total de Propiedad*: la totalidad de los gastos durante la vida útil del software.
- *Facilidad de uso:* lo fácil y amigable que es usar el software.

Software propietario

Es de gran importancia conocer el software propietario ya que es una opción disponible en el mercado de la industria de la información; el cual se conoce también como software comercial. A continuación se brindarán definiciones y una descripción de las características generales que lo representan.

Definición

En la actualidad, la tecnología se ha encargado de transformar y mejorar las bibliotecas o unidades de información, desde la forma que manejan y organizan el conocimiento hasta los servicios que prestan, ya que ahora tienen mucho mayor alcance gracias al aprovechamiento de Internet. Estos recursos tecnológicos disponibles deben de ser aprovechados por las bibliotecas. Para esto es necesario que se cuente con un sistema integral de automatización que ayude a este objetivo. Dentro del mercado actual existe una opción para la gestión de la información, es el llamado software propietario. A continuación se verán algunas definiciones así como también las raíces del término, que aportarán elementos para tener un concepto más claro de lo que es el software propietario.

El software propietario se usa como sinónimo de software no libre, también es llamado software privativo, software privado o software con propietario. *"Se refiere a cualquier programa informático en el que los usuarios tienen limitadas las posibilidades de usarlo, modificarlo o redistribuirlo (con o sin modificaciones), o que su código fuente no está disponible o el acceso a este se encuentra restringido. En el software no libre una persona física o jurídica (por nombrar algunos: compañía, corporación, fundación) posee los derechos de autor sobre un software negando o no otorgando, al mismo tiempo, los derechos de usar el programa con cualquier propósito; de estudiar cómo funciona el programa y adaptarlo a las propias necesidades (donde el acceso al código fuente es una condición previa); de distribuir copias; o de mejorar el programa y hacer públicas las mejoras (para esto el acceso al código fuente es un requisito previo). De esta manera, un software sigue siendo no libre aun si el*

código fuente es hecho público, cuando se mantiene la reserva de derechos sobre el uso, modificación o distribución (por ejemplo, el programa de licencias shared source de Microsoft). No existe consenso sobre el término a utilizar para referirse al opuesto del software libre" (Culebro Juárez. 2006. p. 17).

Introduciéndonos a las raíces del término "software propietario" proviene de la expresión en inglés "proprietary software". El significado que tiene dentro de la lengua anglosajona, "proprietary" es (poseído o controlado privadamente) (privately owned and controlled), esto significa que enfatiza la reserva de derechos sobre el uso, modificación, redistribución del software. Aunque se sabe de antemano que la expresión "software propietario" es de una traducción literal del inglés, no corresponde su uso como adjetivo en el español, de manera que puede ser considerado como un barbarismo. A pesar de esto, es el término más usado para conceptualizar este tipo de software.

La Free Software Foundation define al software propietario como aquel software que no es libre. Su uso, redistribución o modificación está prohibida, o requiere que usted solicite autorización; está tan restringida que no puede hacerla libre de un modo efectivo. Es de dominio privado, porque una determinada persona tiene la titularidad de los derechos de autor y goza de un derecho exclusivo respecto de su utilización. Niega a otras personas el acceso al código fuente del software y derecho a copiar, modificar y estudiar el software (Carranza Torres. 2004. p. 103).

Al respecto Stella Rodríguez menciona que el software propietario es aquel software que está siendo desarrollado por una entidad que tiene la intención de hacer dinero del uso del software. El software propietario es aquel software que es imposible de utilizar en otro hardware o terminal, modificar y transferir sin pagar derechos a su creador o desarrollador (Rodríguez. 2008).

Retomando lo dicho anteriormente, se puede concluir que el software propietario es aquel que es de uso restringido, solo para aquellos que paguen una licencia, aunque ésta no les da derecho a modificarlo, estudiarlo o explotarlo económicamente por cualquier medio, ya que está protegido por copyright llevando la titularidad del dueño o creador ya sea una persona física o una organización o corporación.

Es importante mencionar que el sistema de copyright funciona mediante la concesión de privilegios, y por lo tanto de beneficios, a los editores y a los autores de una obra, llámese libro, disco musical, software, etc. El derecho de autor y copyright constituyen dos conceptos sobre la propiedad literaria y artística. La protección del copyright se limita estrictamente a la obra, sin considerar atributos morales del autor en relación con su obra, excepto la invención; no lo considera como un autor propiamente, pero tiene derechos que determinan las modalidades de utilización de una obra, a diferencia del derecho de (Culebro Juárez. 2006).

Dentro del ámbito bibliotecario se puede decir que al adquirir un SIAB que pertenezca a la categoría de software propietario, es una buena opción que posee muchas ventajas como es el soporte técnico, aunque también representa un costo importante que no todas las bibliotecas pueden cubrir.

Antecedentes

Uno de los más grandes avances dentro de la automatización de bibliotecas fue sin duda, la catalogación legible por máquina, es decir la incorporación de

MARC a los sistemas de automatización, posteriormente, al lograr la integración de las actividades en un solo sistema, surgen los SIAB y con ellos una gran gama de posibilidades para estar en continua evolución.

La utilización de sistemas de automatización dentro de las bibliotecas o centros de información se ha convertido en una necesidad, ya que es una forma de mejorar y expandir los servicios que se prestan, por esta razón es que se ha dado un avance tecnológico en cuanto a sistemas para bibliotecas, lo cuales han tenido un desarrollo considerable a lo largo de su historia, principalmente en los años ochenta.

Para poder tener una visión general sobre lo que ha sido la historia del software propietario, se presenta el siguiente cuadro que contiene los aspectos más relevantes que han sucedido desde la década de los sesentas hasta la actualidad:

Cuadro 1. Historia del software propietario

PERIODO	ANTECEDENTES RELEVANTES
DÉCADA 60's	❖ Se inicia con los antecedentes en la década de los sesentas, En 1961, P. Luhn de IBM desarrolló un programa para producir índice de palabras clave de títulos de los artículos que aparecerían en Chemical Abstract, y Douglas Aircraft Corporation el cuál comenzó a producir fichas catalográficas por computadora. ❖ En el año de 1966, la Biblioteca de la University of Chicago comenzó el desarrollo de un sistema totalmente integrado, fue diseñado para incluir control de adquisiciones, publicaciones periódicas y catalogación comenzó a funcionar en 1968, y fue hasta el año de 1975 que opero con éxito. ❖ También se desarrollaron sistemas cooperativos en Norteamérica, OCLC que se constituyó en 1967 para desarrollar e implementar un sistema automatizado que sirviera a las bibliotecas de Ohio y, a comienzos de los ochenta se extendió a Europa.
DÉCADA 70's	❖ Para la década de los años setenta, se presenciaron grandes cambios y un gran crecimiento de los servicios cooperativos y de recursos compartidos de las bibliotecas. ❖ Para mediados de los años setenta, diversos organismos como la Biblioteca Nacional de Medicina estadounidense, la Lockheed Missiles Corporation y Systems Development Corporation (SDC) comenzaron a ofrecer servicios de consulta en línea desde termínales remotas, sobre una gran variedad de revistas de abstracts e índices legibles por computadoras (Tedd. 1988). ❖ En el año de 1975, uno de los grandes cambios en cuanto a la concepción del software se presenta, la aparición de una nueva computadora diseñada por la empresa Micro Instrumentation Telemetry Systems (MITS), el nuevo modelo llevó el nombre de Altair 8800, su creación marca la presencia de dos personajes importantes en la historia del software propietario, Paul Allen y William Gates.
DÉCADA 80's	❖ La década de los ochentas es caracterizada por la industria del software propietario, ya que este no contaba con una competencia hasta ese momento, pero llego en cuanto a los desarrolladores de plataformas de hardware, se diseñan entonces, las computadoras personales (personal computer) creadas por la compañía Apple Computers (Da Costa Carballo).

	❖ En 1981, IBM introduce en el mercado su PC que "combinó un razonable nivel de poder computacional y un sistema operativo homogéneo que facilitaría el desarrollo de aplicaciones" (Sampedro).
DÉCADA 90's	❖ Esta década inicia en 1990, empresas e instituciones públicas crearon redes internas basadas en la PC, llamadas intranets. ❖ A mediados de esa década, las empresas utilizaban Internet para enlazar las redes internas (Intranet) y las redes privadas entre sí (Sampedro). ❖ En 1994, Netscape Navigator surge como la herramienta idónea para las computadoras personales (PC), un año después cotiza en NASDAQ batiendo records de beneficios (Sampedro). ❖ En el año de 1995 se lanza Windows 95, ❖ Para el año de 1997, Microsoft Internet Explorer supera en utilización a Netscape Comunicator, un año más tarde, se lanza Windows 98 con Internet Explorer 4.0 integrado, supone la derrota de Netscape en la llamada "guerra de los navegadores".
SIGLO XXI	❖ En el año 2001, se presenta el lanzamiento de Windows XP. ❖ Ya en el 2002, existía un total de 38 millones y medio de servidores funcionando en el mundo, de éstos, el 1,82% utilizaban alguna variante de Netscape-Enterprise, el 26,14% software Microsoft y el 65,18% estaban basados en Apache Web Server (Gómez Sánchez. 2004. p. 125-126). Después de dos años, se ven obligados a lanzar nuevas versiones de Windows XP. ❖ En el 2004 después de haber enfrentado demandas por "abuso dominante de mercado" se lanza en enero del 2007 Windows Vista al cual se le hicieron muchas versiones, estos sistemas generaron una serie de problemáticas como la seguridad la cual repercutió en las bajas ventas y que dio pie a la fácil entrada de sistemas operativos libres, finalmente es lanzado al mercado, Windows 7 en octubre del 2009.

Por lo que se puede observar, se ha tenido un avance considerable respecto a los últimos años, donde se aprovechan los beneficios de las TIC para mantener a la vanguardia las bibliotecas por medio de los SIAB, en este caso el software propietario. En el siguiente apartado se describirán las características que debe tener dicho software.

Características

Existen grandes diferencias entre el software libre y el propietario. Este último tiene ciertas particularidades que lo representan y que deben tomarse en cuenta al pretender implementarlo, algunas de sus características son:

- *Limitación de uso*: en el cual se debe tener licencias staff, usuarios u OPAC; estas licencias tienen que ser concurrentes (liberadas), puesto que poseen una vigencia y por tanto si no se pagan no hay actualizaciones.
- *Modificación:* al no conocer y no poder cambiar el código fuente del sistema adquirido, no se puede realizar una parametrización de acuerdo a las necesidades de la unidad de información, resaltando que esto sólo se puede realizar hasta donde lo permita el software.
- *Distribución:* por ser un software propietario perteneciente a una entidad, se deben de adquirir

licencias para su uso, lo cual significa que no puede ser distribuido en diferentes equipos si no va acompañado de su licencia correspondiente.
Por ser propietario y encontrarse protegido por copyright, el software no viene acompañado del código fuente, al momento de adquirirlo.

- *Costos:* en el caso de los costos se debe tomar en cuenta el pago por adquisición inicial del software, mantenimiento anual, parametrización (cada que se requiera alguna adecuación), capacitación sobre el uso y manejo al personal de la unidad de información, habrá que contemplarlo en el presupuesto que se posee.
- *Adjudicación directa:* la mejor forma de adquirir el software es por un proceso de licitación.
- *Espacio en su servidor*: verificar que tipo de servidor se posee, cuál es el espacio que se tiene para instalarlo, si se cuenta con un servidor espejo, que es una forma de almacenar o respaldar la información.
- *Revisar si el proveedor es el desarrollador*:
- *Ver el tamaño de la colección*: para adquirir un software se necesita comprobar si es adecuado al tamaño de la colección que se tiene.
- *Proceso-licitación*

Saber qué es lo que se quiere:

- Linux
- Unix
- Postgrest
- Apache

Qué módulos se necesitan y cuáles ofrece el software:

- Adquisición
- Circulación
- Publicaciones periódicas
- Catalogación
- Administración

Ventajas y desventajas

Ventajas:

Como ya se ha dicho, existe una gran diversidad de SIAB en el mercado actual, al tomar la decisión de implementar un software propietario a pesar del costo que representa, también es importante destacar todas las ventajas que se obtendrán al elegir esta opción para automatizar la unidad de información.

Algunas de las ventajas del software propietario mencionadas por Montserrat Culebro son:

Control de calidad. Las compañías productoras de software propietario por lo general tienen departamentos de control de calidad que llevan a cabo muchas pruebas sobre el software que producen.

Recursos a la investigación. Se destina una parte importante de los recursos a la investigación sobre los usos del producto.

Personal altamente capacitado. Se tienen contratados algunos programadores muy capaces y con mucha experiencia.

Uso común por los usuarios. El software propietario de marca conocida ha sido usado por muchas personas y es relativamente fácil encontrar a alguien que lo sepa utilizar.

Software para aplicaciones muy específicas. Existe software propietario diseñado para aplicaciones muy específicas que no existe en ningún otro lado más que con la compañía que lo produce.

Amplio campo de expansión de uso en universidades. Los planes de estudio de la mayoría de las universidades de México, tienen tradicionalmente un marcado enfoque al uso de herramientas propietarias y las compañías fabricantes ofrecen a las universidades planes educativos de descuento muy atractivos.

Difusión de publicaciones acerca del uso y aplicación del software. Existe gran cantidad de publicaciones, ampliamente difundidas, que documentan y facilitan el uso de las tecnologías proveídas por compañías de software propietario, aunque el número de publicaciones orientadas al software libre va en aumento.

Desventajas:

Es importante revisar también la contraparte de los beneficios, es decir las desventajas que puede tener el software propietario. Algunas contras que tienen el software propietario son:

Cursos de aprendizaje costosos. Es difícil aprender a utilizar eficientemente el software propietario sin haber asistido a costosos cursos de capacitación.

Secreto del código fuente. El funcionamiento del software propietario es un secreto que guarda celosamente la compañía que lo produce. En muchos casos resulta riesgosa la utilización de un componente que es como una caja negra, cuyo funcionamiento se desconoce y cuyos resultados son impredecibles. En otros casos es imposible encontrar la causa de un resultado erróneo, producido por un componente cuyo funcionamiento se desconoce.

Soporte técnico ineficiente. En la mayoría de los casos el soporte técnico es insuficiente o tarda demasiado tiempo en ofrecer una respuesta satisfactoria, puede ser legal o costosa la adaptación de un módulo del software a necesidades particulares.

Es ilegal extender una pieza de software propietario para adaptarla a las necesidades particulares de un problema específico. En caso de que sea vitalmente necesaria tal modificación, es necesario pagar una elevada suma de dinero a la compañía desarrolladora, para que sea ésta quien lleve a cabo la modificación a su propio ritmo de trabajo y sujeto a su calendario de proyectos.

Derecho exclusivo de innovación. La innovación es derecho exclusivo de la compañía desarrolladora. Si alguien tiene una idea innovadora con respecto a una aplicación propietaria, tiene que elegir entre venderle la idea a la compañía dueña de la aplicación o escribir desde cero su propia versión de una aplicación equivalente, para una vez logrado esto poder aplicar su idea innovadora.

Ilegalidad de copias sin licencia para el efecto. Es ilegal hacer copias del software propietario sin antes haber contratado las licencias necesarias.

Imposibilidad de compartir. Si una dependencia de gobierno tiene funcionando exitosamente un sistema dependiente de tecnología propietaria no lo puede compartir con otras

Quedar sin soporte técnico. Si la compañía desarrolladora del software propietario se va a banca rota el soporte técnico desaparece, la posibilidad en un futuro de tener versiones mejoradas de dicho software desaparece y la posibilidad de corregir los errores de dicho software también desaparecen. Los clientes que contrataron licencias para el uso de ese software quedan completamente abandonados a su propia suerte.

Descontinuación de una línea de software. Si una compañía desarrolladora de software es comprada por otra más poderosa, es probable que esa línea de software quede descontinuada y nunca más en la vida vuelva a tener una modificación.

Dependencia a proveedores. En la mayoría de los casos el gobierno se hace dependiente de un solo proveedor.

Nulificación de desarrollo tecnológico de la industria nacional. Nulidad de desarrollo tecnológico de la industria nacional, respecto de la extranjera (las aplicaciones de consumo masivo se desarrollan en otros países).

Después de un atisbo general a los sistemas de automatización de bibliotecas de tipo propietario o comercial más instalados en México, se encuentra una alternativa más que está rompiendo los paradigmas con respecto a la adquisición de un software para las diferentes unidades de información, independientemente del uso y las necesidades que éstas requieran. Este nuevo concepto tiene ya un tiempo considerable a nivel internacional, primordialmente en los países desarrollados o con una amplia capacidad de adquisición de nuevas tecnologías de información. Dentro de la forma contemporánea de gestión de bibliotecas o de cualquier organismo encargado de preservar y difundir información se encuentra con esta nueva tendencia a la cual se le conoce como “software libre” o de “Open Source” que se explicará a detalle en el siguiente apartado.

Software libre (free software) o de código fuente abierto (Open Source)

Un software libre es todo aquel programa informático en el cual los que lo adquieren tienen la posibilidad de modificarlo y mejorarlo de la manera que más convenga, es decir, una vez obtenido el programa este puede ser ejecutado, cambiado, copiado, mejorado, modificado, usado, estudiado y distribuido libremente.

El software gratuito (denominado usualmente Freeware) incluye en algunas ocasiones el código fuente; sin embargo, este tipo de software no es libre en el mismo sentido que el software libre, al menos que se garanticen los derechos de modificación y redistribución de dichas versiones modificadas del programa.

Con el software libre no es necesario solicitar ninguna licencia y cuyos derechos de explotación son para toda la humanidad porque pertenece a todos por igual. Cualquiera puede hacer uso de él, siempre con fines legales y consignando su autoría original.

Los dos anteriores conceptos proporcionan un claro ejemplo de la complejidad del término utilizado en inglés para el “free software” ya que free suele traducirse como libre o gratuito y es aquí donde se desprenden las dos corrientes antes mencionadas, entre el software libre y software gratuito, quedando esto claro y para sustentarlo en una sola línea de investigación, se considerará únicamente como término genérico e incluyente software libre para efectos de este trabajo.

Definición

El término "software libre y open source" comparten modelos de desarrollo similares, sus principales diferencias se encuentran en sus aspectos filosóficos. El software libre se enfoca en las libertades filosóficas que les otorga a los usuarios mientras que el Open Source se enfoca en las ventajas de su modelo de desarrollo.

En este trabajo los tomaremos como sinónimos a ambos, aunque más bien sean complementarios.

Una definición que mejor describe al Software Libre, la aportan da Rosa y Heinz: El software libre se define por su tipo de licenciamiento. Por lo que se puede entonces llamar "software licenciado bajo condiciones libres". Simplificando al máximo, se debe entender que software libre es un software o programa de computación cuya licencia permite ejercer una serie de libertades (Rosa y Heinz. 2007).

Así pues, el software libre es una fuente abierta que permite trabajar bajo cuatro libertades esenciales, esto dicho y comprobado por Richard Stallman, programador estadounidense y fundador del movimiento del software libre, quién en 1985, acuñara dicho término y lo describe bajo estas condiciones:

- ***Libertad de ejecutar*** el programa sea cual sea el propósito (libertad 0).
- ***Libertad de estudiar*** cómo funciona el programa para ajustarlo a tus necesidades (libertad 1). (Es indispensable tener acceso al código fuente).
- ***Libertad de redistribuir*** copias, colaborando con otras personas (libertad 2).
- ***Libertad de modificar***, de tal forma que la comunidad pueda aprovechar las mejoras (libertad 3). (Es indispensable tener acceso al código fuente). (GNU Operating System. Estados Unidos: Free Software Foundation, 2009).

Wayner, afirma que Stallman numeró las libertades empezando por el cero, porque así era como lo hacían los informáticos. Alguien calculó que era más sencillo empezar a numerar las bases de datos con el cero porque no se tiene que restar 1 tan a menudo (Wayner. 2001. p. 129).

La Open Source Initiative utiliza la definición de Open Source para determinar si una licencia de software de computadora puede o no considerarse software abierto. La definición se basó en las Directrices de software libre de Debian, fue escrita y adaptada primeramente por Bruce Perens. Es similar pero no igual a la definición de licencia de software libre.

Las licencias Open Source deben cumplir diez premisas para ser consideradas como tal:

- **Libre redistribución**: el software debe poder ser regalado o vendido libremente.
- **Código fuente**: el código fuente debe estar incluido u obtenerse libremente.
- **Trabajos derivados**: la redistribución de modificaciones debe estar permitida.
- **Integridad del código fuente del autor**: las licencias pueden requerir que las modificaciones sean redistribuidas solo como parches.
- **Sin discriminación de personas o grupos**: nadie puede dejarse fuera.
- **Sin discriminación de áreas de iniciativa**: los usuarios comerciales no pueden ser excluidos.

- **Distribución de la licencia**: deben aplicarse los mismos derechos a todo el que reciba el programa.
- **La licencia no debe ser específica de un producto**: el programa no puede licenciarse solo como parte de una distribución mayor.
- **La licencia no debe restringir otro software**: la licencia no puede obligar a que algún otro software que sea distribuido con el software abierto deba también ser de código abierto.
- **La licencia debe ser tecnológicamente neutral**: no debe requerirse la aceptación de la licencia por medio de un acceso por clic de ratón o de otra forma específica del medio de soporte del software.

Están son características ineludibles en el contenido de las propiedades que hacen del software libre y del open source un elemento de estudio para su incorporación en cualquier sector de la sociedad; gracias a la acción cooperativa que posibilita el manejo de estos tipos de software, es que ha sido posible generar productos finales de gran envergadura y excelentes cualidades técnicas.

Los movimientos de software libre y del open source son de los más exitosos a nivel mundial en los últimos 25 años, impulsado por una comunidad internacional de programadores, con ética dedicada a la causa de la libertad y la cooperación.

El software libre presenta la ventaja de la independencia frente a vicisitudes y arbitrariedades en cuanto a las estrategias comerciales y a la continuidad de diversas herramientas y formatos que se utilicen para el tratamiento de la información en soporte electrónico.

La utilización de un software impacta en tres aspectos:

a) En el acceso a los servicios que ofrece una biblioteca o unidad de información,
b) En los documentos disponibles en soporte electrónico y,
c) A los programas y aplicaciones usados por la unidad de información para sus fines y mejoramiento de sus servicios.

Algunas de las categorías que se relacionan con el software libre y que se considera necesario hacer mención con la finalidad de presentar las diferencias que existen entre un software y otro, esto para que no cause confusión entre sí, son (Porcel Iturralde y Rodríguez Mederos. 2005):

- Software libre (Free Software).- El software libre es un software que posee una autorización para que cualquiera pueda usarlo, copiarlo y distribuirlo, sea en forma literal o con modificaciones, gratis o mediante una gratificación. En particular, esto significa que el código fuente debe estar disponible.
- Software de código fuente abierto (Open Source).- El término software de “código fuente abierto’’ se emplea por algunas personas para dar a entender que es software libre.
- Software de dominio público.- El software de dominio público es aquel software que no está protegido con copyright. Dominio

público, es un término legal que quiere decir precisamente “sin copyright.

- Software con copyleft.- El software protegido con copyleft es un software libre cuyos términos de distribución no permiten a los redistribuidores agregar ninguna restricción adicional cuando éstos redistribuyen o modifican el software. Esto significa que cada copia del software, aun si se ha modificado, debe ser software libre.
- Software libre no protegido con copyleft.- El software libre no protegido con copyleft viene desde el autor con autorización para redistribuir y modificar, así como para añadirle restricciones adicionales. Si un programa es libre pero no está protegido con su copyleft, entonces algunas copias o versiones modificadas pueden no ser libres completamente. Una compañía de software puede compilar el programa, con o sin modificaciones, y distribuir el archivo ejecutable como un producto propietario de software.
- Software cubierto por la GPL. - La GNU GPL (Licencia Pública General), es un conjunto específico de términos de distribución para proteger con copyleft a un programa. El Proyecto GNU la utiliza como los términos de distribución para la mayoría del software GNU.
- El sistema GNU.- El sistema GNU es un sistema operativo libre completo similar a Unix. Debido a que el propósito de GNU es ser libre, cada componente individual en el sistema GNU tiene que ser software libre. No todos tienen que estar protegidos con copyleft, sin embargo; cualquier tipo de software libre es legalmente apto de incluirse si ayuda a alcanzar metas técnicas.
- Software GNU.- Software GNU es software que se libera bajo el auspicio del Proyecto GNU. La mayoría del software GNU está protegido con copyleft, pero no todos; sin embargo, todo el software GNU debe ser software libre.
- Software semilibre.- El software semilibre es software que no es libre, pero viene con autorización para particulares de uso, copia, distribución y modificación - incluye la distribución de versiones modificadas) sin fines de lucro. Pero también incluye otras restricciones.
- Software propietario.- El software propietario es software que no es libre ni semilibre. Su uso, redistribución o modificación está prohibida, o requiere que usted solicite autorización que es tan restringida que no pueda hacerse libre de un modo efectivo.
- Freeware.- El término ``freeware'' no tiene una definición clara aceptada, pero se utiliza frecuentemente para paquetes que permiten la redistribución pero no la modificación, y su código fuente no está disponible. Estos paquetes no son software libre.
- Shareware.-El shareware es software que viene con autorización para redistribuir copias, pero establece que quien continúe el uso de una copia deberá pagar un cargo por licencia. El shareware no es software libre, ni siquiera semilíbre. Existen dos razones por las que no lo es:

1. Para la mayoría del shareware, el código fuente no está disponible; de esta manera, no puede modificarse el programa en absoluto.

2. El shareware no viene con autorización para hacer una copía e instalarlo sin pagar una cantidad por la licencia, ni aún para particulares involucrados en actividades sin ánimo de lucro. En la práctica, la gente a menudo hace caso omiso a los términos de distribución y lo hace de todas formas, pero los términos no lo permiten.

3. Como conclusión "shareware" no es contemplado como un software libre debido a que no cumple con los 4 principios que son característicos y cada uno indispensables para ser considerado como tal.

Software comercial. - El software comercial es software que se desarrolla por una entidad que tiene la intención de obtener utilidades con el uso del software. Como se dijo, ``comercial'' y ``propietario'' ¡no son la misma cosa! La mayoría del software comercial es propietario, pero existe software libre comercial y software no libre no comercial".

Al igual que el software propietario, el software libre se encuentra protegido por el *copyleft* lo cual se refiere a que se tiene que informar a los colegas de las modificaciones que se le hagan debido a las libertades que posee es por eso que es libre.

Ante los matices mencionados anteriormente, que existen dentro del software libre, es necesario tomar en cuenta, que el software libre protegido con *copyleft* impide a los redistribuidores incluir algún tipo de restricción a las libertades propias del software así concebido, es decir, garantiza que las modificaciones seguirán siendo software libre (Carranza Torres. 2004. p. 103).

El *copyleft* se da cuando, en vez de conservar y ejercer el monopolio de explotación, el titular de los derechos de autor renuncia a esa exclusividad, pero lo hace bajo la condición de que las futuras distribuciones de su software, en su versión original o en versiones modificadas, se concedan las mismas facultades de utilización que el titular otorgó.

Es decir, la idea es que se da autorización a cualquiera ejecutar el programa, copiar el programa, modificar el programa y redistribuir versiones modificadas, aclarando que no para agregar restricciones propias. De esta manera, las libertades que caracterizan al software libre quedan garantizadas para cualquiera que tenga una copia. Para que el *copyleft* sea efectivo, las versiones modificadas deben también ser libres.

Es por lo anterior, que la FSF termina afirmando que *copyleft* es la forma general de hacer un programa software libre y requiere que todas las modificaciones y versiones extendidas del programa sean también software libre (Carranza Torres. 2004. p. 103).

Antecedentes

En este apartado se abordarán los sucesos más relevantes de cómo surge el software libre o llamado también open source o de código abierto, así como los grandes cambios que ha ido presentado a lo largo de su historia. El cual se presenta en orden cronológico en el siguiente cuadro:

Cuadro 2. Antecedentes del software libre.

AÑOS	DATOS HISTÓRICOS DEL SOFTWARE LIBRE
1960	El software no era considerado un producto sino un añadido que los vendedores de las grandes computadoras de la época (los mainframes) aportaban a sus clientes para que éstos pudieran usarlos. En dicha cultura, era común que los programadores y desarrolladores de software compartieran libremente sus programas unos con otros. Este comportamiento era particularmente habitual en algunos de los mayores grupos de usuarios de la época, como DECUS (grupo de usuarios de computadoras DEC).
1970	A finales de este año las compañías iniciaron el hábito de imponer restricciones a los usuarios, con el uso de acuerdos de licencia.
1980	Richard Stallman comenzó a trabajar en el proyecto GNU, y un año más tarde fundó la Free Software Foundation (FSF). Stallman introdujo una definición para free software y el concepto de “copyleft”, el cual desarrolló para dar a los usuarios libertad y para restringir las posibilidades de apropiación del software. Algunas pequeñas comunidades comienzan a concebir el software libre. La utopía.
1990	GNU, BSD, X11. Primeros resultados. Linux, sistemas completos libres, primeras empresas. LAMP, Apache, software libre en los medios.
2000-2010	GNOME, KDE, OpenOffice, software libre para todos. Una solución que se debe considerar.
2012-	En nuestros días podemos encontrar muchos software de este tipo. Como sistemas operativos (Suse, Mandrake, Ubuntu, Guadalinex), software ofimático (Open Office, Google docs), sistemas gestores de contenidos (WordPress, Joomla) o reproductores mp3 para Web (Neolao, Xspf, Drewplayer).

La iniciación del software libre surgió debido a la fusión de varios movimientos liberales entorno al ámbito de la informática que bien, esto ya se venía desarrollando años atrás y fue hasta el período de 1970 cuando logró su mayor desarrollo con la creación de sistemas de código abierto o también conocido como software libre, estos factores actuaron de manera decisiva para el perfeccionamiento de dicho sistema entre ellos se encuentran los hackers, el Internet, la filosofía de uso así como también el flujo masivo de la información entre computadoras.

Usar Internet fue vital para la evolución de los sistemas de código abierto, ya que en cuanto uno de sus sistemas es publicado en la red, es posible que alguien inmediatamente lo vea, lo instale y lo mejore. Existen diversos factores por los que sucede lo anterior, pero el que destaca un poco más es el de los hackers (principales promotores y desarrolladores de este tipo de software), los hackers son los que promueven y definen la libertad del acceso a la información, sin embargo, la sociedad los tacha duramente, teniendo una idea falsa de que su primordial actividad radica en ingresar a las bases de datos de cualquier parte del mundo alterando sus códigos de acceso, con el propósito de robar y falsificar la información que contienen.

Por otra parte, el documento de Martín Carranza llamado problemática jurídica del software libre, presenta una reseña

positiva referente a este software, manifestando que el software libre es más antiguo que el propietario, ya que como se explica anteriormente en los años sesenta y setenta no se contemplaba como un producto sino un añadido que los vendedores de grandes computadoras auxiliaban a sus clientes para que éstos pudieran usarlos. En dicha cultura, era habitual que los programadores y desarrolladores de software compartieran libremente sus programas unos con otros. Este procedimiento era exclusivamente tradicional en algunos de los mayores grupos de usuarios de la época, como DECUS (grupo de usuarios de computadoras DEC).

En un inicio al software libre no lo llamaban como tal simplemente lo conocían como software, y con el paso del tiempo y con los grandes avances tecnológicos, este contexto tuvo cambios drásticos en los cuales las computadoras se modernizaron y utilizaban sus propios sistemas operativos propietarios por lo que decidieron crear un software libre.

El software libre ha tenido un avance fundamental, lo cual ha logrado tener un papel muy relevante en el crecimiento y extensión de la red, esto es debido a que la mayor parte de la infraestructura del Internet se basa en protocolos abiertos.

A finales de los setenta las compañías iniciaron la moda de aplicar limitaciones a los usuarios, con el uso de convenios de licencia.

Se cree que con la libertad de expresión se crea el software libre y se relaciona con el proyecto conocido como el proyecto GNU (Stallman. 1983) que fue fundado por Free Software Foundation (FSF) y escribe el acta fundacional de la comunidad: El Manifiesto GNU. Dicho proyecto tenía el objetivo de hacer un sistema operativo de forma que nadie tuviera que costear por el software y organizar una comunidad a partir del software.

Richard Stallman en el año de 1971, mientras estudiaba en el primer año de la carrera de física en la Universidad de Harvard, ingreso al Laboratorio de Inteligencia Artificial del Massachussets Institute of Technology (MIT), lo cual provocó que se convirtiera en un hacker del mismo laboratorio, en este laboratorio enseñaba a compartir el código fuente del software que se manejaba.

Stallman disconforme con la idea de que el software tuviera propietarios, en ese año decide retirarse de su trabajo en el MIT con la finalidad de erigir y ampliar un sistema operativo perfecto.

Con el paso del tiempo Stallman decidió implantar un sistema simultáneo con UNIX pero con las características de un software libre, y como consecuencia obtuvo GNU (GNU is Not Unix), acrónimo que significa “GNU no es Unix”. Con este acontecimiento nació una trayectoria y disputa por el acceso libre a la información, y una libre expresión, estos esfuerzos se vieron traducidos en la creación en 1985 de la Free Software Foundation (FSF o Fundación para el Software Libre) con el propósito de brindar un soporte logístico, así como legal y financiero al proyecto GNU. La FSF estipuló programadores para contribuir a GNU, no obstante la gran parte del desarrollo ha sido causa del trabajo continuo de los voluntarios. A medida que GNU ganaba renombre, negocios, muchos interesados comenzaron a contribuir al desarrollo o comercialización de productos GNU y el correspondiente soporte técnico.

En esa época, Stallman y un conjunto de desarrolladores que colaboraban con él, implantaron dos instrumentos fundamentales para un sistema operativo

estilo UNIX: el compilador “GCC” para lenguaje C y el editor de texto “EMACS” (editor de texto) con “Lips” (Interferencias Lógicas por segundo) para cifrar comandos de edición.

Con este acontecimiento, Stallman expidió una licencia que consintiera para que los usuarios continuaran usando, copiando, estudiando, modificando o redistribuyendo el software instaurado y desarrollado por la FSF, pero que les impidiera adecuarse a las modificaciones que en el futuro ellos mismos ejecutarán, o combinaran el software GNU con otro tipo de software.

Otro de los hechos relevantes fue el de Roy Tennant quien en 2007 publica “*el manifiesto del software para bibliotecas”,* el cual tiene como finalidad ofrecer un intento por razonar cuál es la relación entre las bibliotecas y los vendedores de sistemas. Este manifiesto es un apoyo de vital importancia para los bibliotecarios y maneja un orden y una serie de derechos y compromisos que se deben de ejercer por parte de los usuarios del software:

Derechos como consumidor:

- ☞ Tengo derecho a saber lo que existe ahora y cuál es su potencial funcionalidad futura.
- ☞ Tengo derecho a usar lo que compro.
- ☞ Tengo derecho al API (Application Programming Interface) si he comprado el producto.
- ☞ Tengo derecho a documentación completa y actualizada.
- ☞ Tengo derecho a mis datos.
- ☞ Tengo derecho a tener acceso de sólo lectura a la base de datos.
- ☞ Tengo derecho a no hacer las cosas sencillas innecesariamente complicadas.
- ☞ Tengo derecho a conocer las líneas de desarrollo y la estimación de tiempo de desarrollo del producto que he comprado.
- ☞ Tengo derecho a hacer preguntas técnicas a un equipo capaz de comprenderlas y responderlas.
- ☞ Tengo derecho a no ser un probador involuntario.
- ☞ Tengo derecho a que se conserven mis personalizaciones y configuraciones en futuras actualizaciones.

Responsabilidades como consumidor:

- ☞ Tengo la responsabilidad de conocer las necesidades de mis usuarios.
- ☞ Tengo la responsabilidad de poner las necesidades de mis usuarios por delante de las mías.
- ☞ Tengo la responsabilidad de comunicar mis necesidades clara y específicamente.
- ☞ Tengo la responsabilidad de que las mejoras que pido sean realmente lo que quiero.
- ☞ Tengo la responsabilidad de asignar honestamente las prioridades de las mejoras.
- ☞ Tengo la responsabilidad de darme cuenta de que no soy especial.
- ☞ Tengo la responsabilidad de elegir software usando un procedimiento limpio y razonable.
- ☞ Tengo la responsabilidad de informar de los errores reproducibles de forma que puedan reproducirse.
- ☞ Tengo la responsabilidad de informar de los errores

irreproducibles con todos los detalles que pueda.

- ☞ Tengo la responsabilidad de ver críticamente cualquier ajuste a las configuraciones predefinidas.

Responsabilidades compartidas:

- ☞ Tenemos la responsabilidad de comenzar desde una posición de respeto mutuo.
- ☞ Tenemos la responsabilidad de comunicarnos correctamente.
- ☞ Tenemos la responsabilidad de establecer y mantener un proceso de mejora racional.
- ☞ Tenemos la responsabilidad de mantener las necesidades del usuario final como primordiales.
- ☞ Tenemos la responsabilidad de relajarnos y divertirnos (Tennant. 2007).

Con estos puntos se expresa que cualquier usuario de software tiene que cumplir mínimamente con lo expuesto anteriormente.

Características

Después de los antecedentes, ahora se señalarán las características más representativas de este tipo de software y mencionar que las empresas de software propietario no venden el software sino licencias de uso que restringen libertades. Es como si se comprará una bicicleta pero el que la vendió determinará adonde pueden ir con ella o adonde no.

En ese caso no sé es el dueño de la bicicleta sino un usuario cautivo. Además al no poder analizar el código fuente no se sabe qué procesos corre el software propietario en la máquina y estos pueden ser dañinos violando la privacidad.

Stallman ofreció una conferencia en donde explica que muchos de los programas propietarios poseen características maliciosas, algunos para espiar a los usuarios, otros para atacar a los usuarios a través de backdoors (es una secuencia especial dentro del código de programación, mediante la cual se pueden evitar los sistemas de seguridad del algoritmo para acceder al sistema).

Además al evitar copiar y compartir quiebra los lazos de solidaridad social entre las personas. Por último, como no permiten modificar el software comercial que viene empaquetado puede ser que no se adapte a las necesidades específicas y mandar a hacer software a medida es muy caro.

Por otra parte, cabe mencionar que Stallman menciona que la idea del software libre es que los usuarios de las computadoras se merecen ciertas libertades en el uso del software, merecen tener el control del software que usan, explicó que:

"*Un programa es libre cuando respeta la libertad del usuario, y un programa que no es libre, denominado propietario, mantiene a los usuarios divididos y sin ayuda, como un sistema colonial. Como no tienen el código fuente, no pueden cambiarlo, y no pueden verificar qué hace*" (Stallman. 2002).

Si se tienen las 4 libertades antes mencionadas, eso significa que el sistema social del programa es ético. Y si una de estas libertades es insuficiente o no existe, se trata de software propietario, porque impone un sistema social no ético a los usuarios. Entonces desde este enfoque la diferencia entre software libre y propietario para nada es

una cuestión técnica. Es una pregunta ética, social y política.

El software libre tiene sus bases en una ideología que dice que el software no debe tener dueños: es un asunto de libertad. La gente debería ser libre de usarlo en todas las formas que sean socialmente útiles.

De esta manera, el movimiento del software libre pone lo que es beneficioso para la sociedad por encima de los intereses económicos o políticos.

Entre los beneficios que percibe la sociedad se pueden mencionar los siguientes:

- Tecnologías transparentes, confiables y seguras.
- Tecnologías como bien público.
- Promoción del espíritu cooperativo, en el que el principal objetivo es ayudar a su vecino.
- Precios justos.

El Software libre ofrece a las personas la posibilidad de utilizar, estudiar, modificar, copiar y redistribuir el software (como se ha mencionado de manera recurrente a lo largo del artículo). Pero para hacer efectivas estas libertades, el código fuente de los programas debe estar disponible.

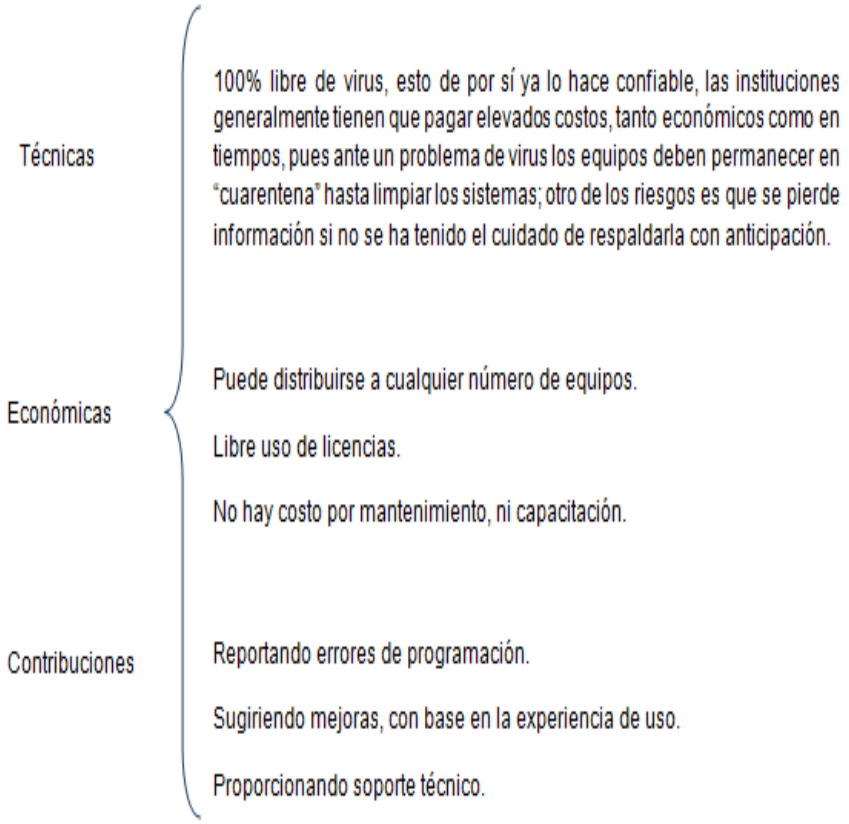
Gracias a estas libertades se obtienen muchos beneficios prácticos como son los siguientes:

- Podemos ejecutar el software cuando queramos y para lo que queramos.
- Podemos aprender de los programas existentes.
- Podemos mejorarlos.
- Podemos adaptarlos para que se ajusten a nuestras necesidades.
- Podemos basarnos en ellos, de forma que evitamos los costos adicionales de empezar un programa desde 0.
- Podemos formar negocios alrededor de la creación, distribución, soporte y capacitación de programas libres.

Existen diferentes aplicaciones del software libre entre ellos destacan algunos más usuales que son:

- El sistema operativo Linux
- El servidor de Web Apache
- El manejador de bases de datos objeto-relacional PostgreSQL
- El navegador Mozilla
- La suite de aplicaciones de escritorio OpenOffice
- El servidor de correo Sendmai (Arriola Navarrete y Butrón Yáñez. 2008. p.45).

Esquema 2. A favor del software libre y del open source



Fuente: Arriola Navarrete, Oscar. *2011.* "Open Access y software libre: un área de oportunidad para las bibliotecas". *En: Biblioteca universitaria: revista de la Dirección General de Bibliotecas de la UNAM.* Vol. 14, no. 1 (enero-junio). p.37

Ventajas y desventajas

Ventajas

En países como Estados Unidos, se le está dando cierto auge al software libre como se puede observar en la revista Library Journal del mes de abril de 2010 (Breeding. 2010), ya que cada vez está siendo utilizado por más bibliotecas. Usar software libre tiene varias ventajas, Montserrat Culebro menciona las siguientes:

Bajo costo de adquisición y libre uso. El software, como mercadería, por lo general no está a la venta. Lo que el usuario adquiere, a través de una erogación monetaria o sin ella, es una licencia respecto de los usos que puede dar a los programas en cuestión. El usuario que adquiere software libre lo hace sin ninguna erogación monetaria o a muy bajo costo y ofrece un conjunto de recursos muy amplios. Cualquier persona con una computadora y una conexión a Internet puede utilizar un software libre. Para la mayoría de usuarios individuales el software libre es una opción atractiva por las libertades que garantiza sin necesidad de verse agobiados por el precio.

Innovación tecnológica. El software libre, tiene como objetivo principal compartir la información, trabajando de manera cooperativa. Este es principalmente el modelo sobre el que la humanidad ha

innovado y avanzado. La ideología de los defensores del software libre, es que el conocimiento le pertenece a la humanidad, sin hacer distingos. Por lo tanto, los usuarios tienen un destacado papel al influir decisivamente en la dirección hacia donde evolucionan los programas: votando los errores que quieren que sean corregidos, proponiendo nueva funcionalidad al programa, o contribuyendo ellos mismos en el desarrollo del software (a finales del año 2004 se publicó una lista de las innovaciones más importantes en software del año 2004. Se consideró como innovación número uno el navegador libre FireFox y de los diez programas mencionados también se encontraba OpenOffice.org).

Requisitos de hardware menores y durabilidad de las soluciones. Aunque resulta imposible generalizar, si existen casos documentados que demuestran que las soluciones de software libre tienen unos requisitos de hardware menores, y por lo tanto, son más baratas de implementar. Por ejemplo, los sistemas Linux que actúan de servidores pueden ser utilizados sin la interfaz gráfica, con la consecuente reducción de requisitos de hardware necesarios.

También, es importante destacar que en el software propietario el autor puede decidir en un momento dado no continuar el proyecto para una cierta plataforma, para un hardware que considera antiguo, o descontinuar el soporte para una versión de su software. En las aplicaciones de software libre, estas decisiones no pueden ser tomadas por una empresa o individuo sino por toda una comunidad, con diferentes intereses. Lo que se traduce en un mejor soporte de manera general para las versiones antiguas de software y de plataformas de hardware o software minoritarias.

Escrutinio público. El modelo de desarrollo de software libre sigue un método a través del cual trabajan de forma cooperativa los programadores que en gran parte son voluntarios y trabajan coordinadamente en Internet. Lógicamente, el código fuente del programa está a la vista de todo el mundo, y son frecuentes los casos en que se reportan errores que alguien ha descubierto leyendo o trabajando con ese código.

El proceso de revisión pública al que está sometido el desarrollo del software libre imprime un gran dinamismo al proceso de corrección de errores. Los usuarios del programa de todo del mundo, gracias a que disponen del código fuente de dicho programa, pueden detectar sus posibles errores, corregirlos y contribuir a su desarrollo con sus mejoras. Son comunes los casos en que un error de seguridad en Linux se hace público y con él la solución al mismo. Con el software propietario la solución de los errores no llega hasta que el fabricante del programa puede asignar los recursos necesarios para solventar el problema y publicar la solución.

Independencia del proveedor. El software libre garantiza una independencia con respecto al proveedor, gracias a la disponibilidad del código fuente. Cualquier empresa o profesional, con los conocimientos adecuados, puede seguir ofreciendo desarrollo o servicios para nuestra aplicación. En el mundo del software propietario, solo el desarrollador de la aplicación puede ofrecer todos los servicios, con el software libre, como su denominación lo indica, su uso es libre: todo aquel que lo tiene en su poder puede usarlo cuantas veces quiera, en cuantas máquinas quiera, para los fines que quiera. De esta manera, utilizándolo, el usuario se libera de toda dependencia de un proveedor único, y puede administrar su crecimiento y operación

con total autonomía, sin temor de costos ocultos ni extorsiones. Uno de los grandes problemas en la industria del software propietario es la dependencia que se crea entre el fabricante y el cliente.

Industria local. Si bien es cierto que no existen aún soluciones libres para todas las necesidades de los usuarios; tampoco existen soluciones propietarias para todas las necesidades. En aquellos casos en que la solución libre no existe, hay que desarrollarla, lo que significa esperar a que alguien más tropiece con la necesidad y lo desarrolle, o desarrollarlo uno mismo (o lo que es igual, pagar para que alguien lo desarrolle). La diferencia está en que en aquellos casos en que si hay una solución libre disponible, el usuario puede utilizarla inmediatamente y sin reparos de ningún tipo, mientras que con las soluciones propietarias siempre tiene que pagar, y lo que obtiene a cambio es una "solución" cerrada y secreta, en vez de una herramienta que le permita crecer y operar con seguridad y libertad.

En México, es casi nula la industria de software y las aplicaciones de consumo masivo se desarrollan en otros países. Un gran porcentaje de la industria se basa en distribuir y dar apoyo e información de productos realizados fuera de nuestras fronteras, por lo tanto, la parte de creación y desarrollo de software es realmente la parte de la industria que requiere de excelentes ingenieros y programadores que sin duda los hay en México, lo que generaría que nuestra industria local creciera generando valor y conocimiento y transcender tecnológicamente.

Datos personales, privacidad y seguridad. Seguridad nacional. Para cumplir con sus funciones, el Estado debe almacenar y procesar información relativa a los ciudadanos. La relación entre el individuo y el Estado depende de la privacidad e integridad de estos datos, que por consiguiente deben ser adecuadamente resguardados contra tres riesgos específicos:

Riesgo de filtración: Los datos confidenciales deben ser tratados de tal manera que el acceso a ellos sea posible exclusivamente para las personas e instituciones autorizadas.

Riesgo de imposibilidad de acceso: Los datos deben ser almacenados de tal forma que el acceso a ellos por parte de las personas e instituciones autorizadas este garantizado toda la vida útil de la información.

Riesgo de manipulación: La modificación de los datos debe estar restringida, nuevamente, a las personas e instituciones autorizadas.

Adaptación del software. El software propietario habitualmente se vende en forma de paquete estándar, que muchas veces no se adapta a las necesidades específicas de empresas y administraciones. Una gran parte de la industria del software se basa en desarrollar proyectos donde se requiere software personalizado. El software libre permite personalizar, gracias al hecho de que se dispone del código fuente, los programas tanto como sea necesario hasta que cubran exactamente las necesidades. La personalización es un área muy importante en que el software libre puede responder mucho mejor que el software de propiedad a unos costos mucho más razonables. Un gran porcentaje de uso de software en los países es de uso interno para empresas y las dependencias de la administración pública, que requiere un alto grado de personalización y donde el software puede proporcionar desarrollos más económicos.

Lenguas minoritarias, traducción, uso e impulso de difusión. Las lenguas minoritarias existentes en México, como el náhuatl, zapoteco, mixteco, mazahua, purépecha, entre otros de nuestras comunidades indígenas tienen pocas posibilidades de desarrollarse en el mundo del software propietario y para aquellas personas que no dominan el castellano y solo la lengua original de la comunidad no tendría acceso al uso y manejo de las computadoras, además de que iría perdiendo cada vez mas estos idiomas aunque muchos quizás opinen que este no es una ventaja importante o un tema relevante, pero pensamos que podría servir como un medio para impulsar la difusión de estas lenguas y que no queden en el olvido y se pierda parte de esta cultura y por lo tanto de nuestro patrimonio nacional.

Uno de los proyectos para disminuir la brecha digital es el llamado OLPC (One Laptop per Child) o una laptop por niño, "*John Davies, Vicepresidente del programa World Ahead de Intel, manifestó al diario "El Universal" de México, que Intel brindará a muchos salones de clase de ese país laptops de muy bajo costo, resistentes y adecuadas para el trabajo rudo de los niños"* (Guglielmetti. 2007).

Software y Estado. El Estado, por su envergadura y por su papel de administrador de los bienes comunes, es particularmente vulnerable a los riesgos del software propietario, a la vez que está en una posición particularmente estratégica para beneficiarse con las ventajas del software libre, y también para contribuir a su desarrollo. Si para el sector privado adquirir software libre puede resultar conveniente, pensamos que para el Estado se debe volver una necesidad. El Estado administra información pública y privada acerca de los ciudadanos, y simultáneamente propiedad de los ciudadanos. La inseguridad intrínseca en la operación "secreta" del software propietario implicaría exponer estos datos a un riesgo injustificable de sustracción y alteración.

Desventajas

El uso adecuado del software libre puede representar grandes ahorros y beneficios a una biblioteca, aunque también es relevante marcar ciertas desventajas que puede representar como (Culebro Juárez. 2006):

El aprendizaje de los usuarios es menor. Si se pone a dos señoras que nunca han tocado una computadora, probablemente tardarán lo mismo en aprender a usar software propietario por ejemplo de Microsoft, que software libre como Gnome o KDE; pero si antes los usuarios ya usaron software propietario generalmente tarda más en aprender a usar un software libre.

El software libre no tiene garantía proveniente del autor. Se puede hacer uso libre del paquete e incluso modificarlo, pero el autor no se hace responsable por cualquier tipo de falla o inconveniente que pueda surgir.

Los contratos de software propietario no se hacen responsables por daños económicos, y de otros tipos por el uso de sus programas. El software libre se adquiere, se vende "AS IS" (tal cual) sin garantías explicitas del fabricante, sin embargo, puede haber garantías específicas para situaciones muy específicas.

Se necesita dedicar recursos a la reparación de errores. Sin embargo, en el software propietario es imposible reparar errores, hay que esperar a que saquen a la venta otra versión.

No existen compañías únicas que respalden toda la tecnología. Para el desarrollo de un software se hacen trabajos cooperativos donde muchas personas están involucradas, dicho grupo de personas pueden tener cambios constantes de personal, incluso los usuarios u otras organizaciones pueden aportar algo al desarrollo del software, por esta razón no hay una compañía u organización única que respalden por completo el software.

Las interfaces gráficas de usuario (GUI) y la multimedia apenas se están estabilizando. Aunque hay un número cada vez mayor de usuarios que aseguran que las interfaces gráficas más populares en el software libre (KDE, GNOME y el manejador de ventanas WindowMaker) son ya lo suficientemente estables para el uso cotidiano y lo suficientemente amigables para los neófitos de la informática.

La mayoría de la configuración de hardware no es intuitiva. Se requieren conocimientos previos acerca del funcionamiento del sistema operativo y fundamentos del equipo a conectar para lograr un funcionamiento adecuado. Sin embargo, la documentación referente a la configuración del hardware es tan explícita y detallada que permite al usuario neófito profundizar en el conocimiento de su hardware en muy pocas horas y una vez teniendo ese conocimiento la configuración se vuelve trivial.

Únicamente los proyectos importantes y de trayectoria tienen buen soporte, tanto de los desarrolladores como de los usuarios. Sin embargo, existen muchos proyectos más pequeños y recientes que carecen del compromiso necesario por parte de sus usuarios o desarrolladores para que sean implementados de manera confiable. Estos proyectos importantes que tienen un excelente soporte cubren más del 90% de las necesidades de cómputo del usuario promedio.

El usuario debe tener nociones de programación. La administración del sistema recae mucho en la automatización de tareas y esto se logra utilizando, en muchas ocasiones, lenguajes de guiones (perl, python, shell, etc). Sin embargo, existen en la actualidad muchas herramientas visuales que permiten al usuario no técnico llevar a cabo tareas de configuración del sistema de una manera gráfica muy sencilla sin la necesidad de conocimientos de programación.

En sistemas con acceso a Internet, se deben de monitorear constantemente las correcciones de errores de todos los programas que contengan dichos sistemas, ya que son fuentes potenciales de intrusión. En el software propietario también se deben de monitorear constantemente las correcciones de errores de todos los programas y además es imposible reparar las vulnerabilidades (que en su mayoría son reparaciones triviales) por uno mismo sino que hay que esperar a que la compañía fabricante libere la actualización y en algunos casos hay que pagar dinero extra por obtener esta.

La diversidad de distribuciones, métodos de empaquetamiento, licencias de uso, herramientas con un mismo fin, etc., pueden crear confusión en cierto número de personas. Hay quienes ven esto como una fortaleza porque se pueden encontrar desde distribuciones especializadas en sistemas embebidos con muchas limitantes de almacenamiento y dispositivos periféricos de uso especializado hasta distribuciones optimizadas para su uso en servidores de alto rendimiento con varios procesadores y gran capacidad de almacenamiento; pasando por las distribuciones diseñadas para su uso en

computadoras de escritorio y entre las cuales se encuentran las diseñadas para el usuario neófito que son muy fáciles de instalar y utilizar y las diseñadas para el usuario avanzado con todas las herramientas necesarias para explotar el software libre en todo su potencial. Cabe notar que la posibilidad de crear distribuciones completamente a la medida para atacar situaciones muy específicas es una ventaja que muy pocas marcas de software propietario pueden ofrecer y que Microsoft ha sido completamente incapaz de hacer.

Conclusiones

Con las nuevas tendencias en cuanto a normatividad para la organización de la información, acceso a los recursos electrónicos y la gestión de colecciones digitales, la oferta actual de sistemas de automatización se ha ido convirtiendo en un obstáculo para el progreso. Está emergiendo una nueva generación de plataformas de servicios digitales para bibliotecas, diseñadas para proporcionar un apoyo integral a la gestión y al acceso de todo tipo de materiales de la biblioteca: impresos, electrónicos y digitales. Estos nuevos sistemas implicarán una modernización de las arquitecturas orientadas a servicios, con un mayor desarrollo del concepto de "software como servicio" y de otros modelos basados en la "nube".

Los grandes desarrolladores de software para bibliotecas se han dado cuenta, tras un período de investigación, que deben ofrecer opciones alternativas a las bibliotecas, tanto en la automatización del back-end (operaciones internas de los bibliotecarios) como en el descubrimiento (búsqueda y localización de información por parte del usuario final). Así, han surgido varias soluciones nuevas, que a menudo representan modelos conceptuales muy diferentes que van más allá del formato MARC21, los bibliotecarios siempre buscan hallar soluciones que mejoren las experiencias de sus usuarios de forma inmediata, especialmente a través de productos de descubrimiento.

Estos nuevos software de automatización surgirán en el contexto de hechos y circunstancias que moldean el trabajo de las bibliotecas universitarias, especialmente en relación con la proporción cada vez mayor de materiales electrónicos y digitales que componen las colecciones.

El modelo tradicional de SIAB no es muy adecuado para la gestión de recursos electrónicos y ello ha llevado a la proliferación de productos complementarios tanto de software libre como de propietario, para abordar este aspecto cada vez más estratégico para el funcionamiento de las bibliotecas, en donde además de su SIAB implementan servicios basados en Open URL para la resolución de enlaces, sistemas especializados de gestión de recursos electrónicos, interfaces de descubrimiento, plataformas de gestión de activos digitales (*digital asset management*), repositorios institucionales, servidores proxy y otros componentes que abordan un aspecto u otro de las operaciones digitales.
Cada una de estas aplicaciones suele requerir la implementación de procesos de gestión y plataformas de hardware separadas, de lo cual resulta un entorno muy complejo de gestión, que supera la capacidad de muchas bibliotecas. Estos sistemas a menudo no logran interactuar entre sí con efectividad, debido a que usan modelos de datos aislados y a la falta de APIs robustas y bien desarrolladas.

Para el año 2016 se espera la implementación de plataformas de servicios bibliotecarios más completas y adecuadas. Hay que tener en cuenta que

actualmente muchas bibliotecas todavía usan SIAB diseñados hace más de una década, Esto en el contexto internacional, en el ámbito mexicano el tema es mucho más grave, porque estamos hablando en su mayoría de bibliotecas con recursos limitados las cuales no tienen más remedio que tratar de sacar el mejor partido posible de su sistema de automatización obsoleto. Históricamente, la plena realización del ciclo de vida de los productos es de una década.

Breeding comenta que mirando el calendario de 2026 se puede aventurar que las plataformas de servicios bibliotecarios habrán alcanzado la plena madurez y su despliegue será casi universal. De la misma manera que casi cualquier biblioteca universitaria de hoy ha puesto en marcha un sistema integrado de biblioteca de algún tipo, para el año 2026 se puede esperar que sea usual disponer de plataformas capaces de manejar exhaustivamente recursos impresos, electrónicos y digitales.

Muchos software de automatización utilizados en las bibliotecas universitarias actuales se habrán extinguido. De todas formas, es de esperar que también sobrevivan algunos de ellos pero la década de evolución los hará casi irreconocibles con respecto a su forma actual, puesto que habrán adquirido muchas de las características de las plataformas de servicios bibliotecarios puestas en marcha en 2012. El entorno de automatización de bibliotecas ha sido siempre favorable a un enfoque evolutivo para el desarrollo de productos, a pesar de que el ciclo actual presenta una serie de alternativas revolucionarias. Algunos proveedores han sido capaces de navegar con éxito a través de una larga serie de ciclos tecnológicos, como Innovative Interfaces, Inc., VTLS, SirsiDynix (y sus compañías predecesoras) y Ex Libris. Podemos anticipar que éstos, y otros, seguirán haciendo evolucionar los productos existentes o crearán otros nuevos a lo largo de la próxima era de la automatización de bibliotecas (Breeding. 2011. p. 9-15).

Fuentes consultadas

Arriola Navarrete, Oscar. 2011. *"Open Access y software libre: un área de oportunidad para las bibliotecas". En:* Biblioteca universitaria: revista de la Dirección General de Bibliotecas de la UNAM. Vol. 14, no. 1 (enero-junio).

Arriola Navarrete, Oscar y Butrón Yáñez, Katya. 2008. "Sistemas integrales para la automatización de bibliotecas basados en software libre". [en línea]. En: *ACIMED.* vol.18, no.6, diciembre. [Consulta 20 Agosto 2011]. Disponible en Internet: http://bvs.sld.cu/revistas/aci/vol18_6_08/aci091208.html

Breeding, Marshall. 2010. "Automation System Marketplace 2010: New Models, Core Systems. Discovery interfaces add a new facet to the marketplace". [en línea]. En: Library Journal. Vol. 135, No. 6, April 1. [Consulta 27 Mayo 2010]. Disponible en Internet: http://www.libraryjournal.com/article/CA6723662.html

Breeding, Marshall. 2011. "Current and future trends in information technologies for information units". En: *El profesional de la información.* v. 21, no. 1, p. 9-15.

Carranza Torres, Martín. 2004. *Problemática jurídica del software libre.* Buenos Aires: Lexis Nexis.

Culebro Juárez, Montserrat. 2006. *Software libre vs. Software propietario: ventajas y desventajas*. [en línea]. [Consulta 18 Agosto 2010]. Disponible en Internet: http://www.softwarelibre.cl/drupal//files/32693.pdf

Da Costa Carballo, Carlos Manuel. *Los orígenes de la informática*. [en línea]. Universidad Complutense de Madrid [Consulta 11 Agosto 2010] Disponible en Internet: http://revistas.ucm.es/byd/11321873/articulos/RGID9898120215A.PDF

David, Lourdes T. 2001. "Introduction to Integrated Library Systems". [en línea] En: *ICT for Library and Information Professionals: A Training Package for Developing Countries*. [Consulta 4 Agosto 2010] Disponible en Internet: http://arizona.openrepository.com/arizona/bitstream/10150/106374/1/125105e.pdf

García Melero, Luis Ángel y García Camarero, Ernesto. 1999. *Automatización de bibliotecas*. Madrid: Arcod/libros.

GNU Operating System. 2009. Estados Unidos: Free Software Foundation. [Consulta 10 Febrero 2012]. Disponible en Internet: http://www.gnu.org/philosophy/free-sw.html

Goldstein, Charles M. 1983. *Integrated Library Systems*. [en línea]. [Consulta 5 Agosto 2010]. Disponible en: Internet: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC227197/pdf/mlab00067-0079.pdf

Gómez Sánchez, Rafael. 2004. "Software libre vs. Software propietario: programando nuestro futuro". [en línea]. En: *HAOL*. Núm. 2, Otoño p. 125-126. [Consulta 25 Julio 2010]. Disponible en Internet: http://www.historia-actual.com/HAO/Volumes/Volume1/Issue2/esp/v1i2c10.pdf.

Guglielmetti, Marcos. 2007. *La competencia de la OLPC llega a México por parte de Intel*. [en linea]. [Consulta 6 Septiembre 2010]. Disponible en Internet: http://www.mastermagazine.info/articulo/11186.php.

MicroISIS - *Diecinueve años de historia: Un sistema gratuito multilingüe, que facilita el intercambio de información y la cooperación a nivel Internacional*. [en línea]. Instituto de Estudios Documentales sobre Ciencia y Tecnología (IEDCYT). [Consulta 4 Agosto 2010]. Disponible en Internet: http://www.cindoc.csic.es/isis/historia.htm

Moya Anegón, Félix de. 1995. *Los sistemas integrados de gestión bibliotecaria*. Madrid: ANABAD.

Open Source Software. [en línea]. En: Postnote June 2005 Number 242, 2005. [Consulta 20 Agosto 2010]. Disponible en Internet: http://www.parliament.uk/documents/post/postpn242.pdf

Perens, Bruce. *Open standards, principles and practice*. [Consulta 23 Mayo 2012]. Disponible en Internet: http://perens.com/OpenStandards/Definition.html

Porcel Iturralde, María Laura y Rodríguez Mederos, Mabel. 2005. "Software libre: una alternativa para las bibliotecas". [en línea]. En: *ACIMED*. Vol. 13, no. 6. [Consulta 30 Agosto 2010]. Disponible en Internet: http://www.bvs.sld.cu/revistas/aci/vol13_6_05/aci090605.htm

Rodríguez, Gladys Stella. 2008. "El software libre y sus implicaciones jurídicas". [en línea]. En: *Revista de derecho*. No. 30. Barranquilla:

Universidad del Norte. [Consulta 19 Agosto 2010]. Disponible en Internet: http://redalyc.uaemex.mx/pdf/851/85112306007.pdf

Rosa, Fernando da y Heinz, Federico. 2007. *Guía práctica sobre software libre su selección y aplicación local en América Latina y el Caribe*. Organización de las Naciones Unidas para la Educación la Ciencia y la Cultura. [Consulta 5 Febrero 2012]. Disponible en Internet: http://unesdoc.unesco.org/images/0015/001560/156096s.pdf

Sampedro, José Luis. 2006. "Construcción de capacidades de innovación en la industria de software a través de la creación de interfaces: el caso de empresas mexicanas". [en línea] En: *1er Congreso Iberoamericano de Ciencia, Tecnología, Sociedad e Innovaciones CTS+I. [*Consulta 11 Agosto 2010] Disponible en Internet: http://www.oei.es/memoriasctsi/mesa8/m08p19.pdf

Stallman, Richard. 1983. *El manifiesto de GNU*. [en línea]. [Consulta 10 Agosto 2010]. Disponible en Internet: http://www.gnu.org/gnu/manifesto.es.html

Stallman, Richard. 2002. *Software libre para una sociedad libre.* [en línea]. GNU Press. [Consulta 30 Julio 2010]. Disponible en Internet: http://biblioweb.sindominio.net/pensamiento/softlibre/softlibre.pdf.

Tedd, Lucy A. 1988. *Introducción a los sistemas automatizados de bibliotecas*. Tr. Isabel Quintana. Madrid: Díaz de Santos.

Tennant, Roy. 2007. *Library software manifiesto.* [en línea]. [Consulta 26 Agosto 2010]. Disponible en Internet: http://techessence.info/manifesto/

Wayner, Peter. 2001. *La ofensiva del software libre: como Linux y el movimiento del software libre se impusieron a los titanes de la alta tecnología*. Barcelona: Garnica.