UNIVERSIDAD MICHOACANA DE SAN NICOLAS DE HIDALGO

FACULTAD DE CONTADURIA Y CIENCIAS ADMINISTRATIVAS

FACTIBILIDAD DEL USO DEL SOFTWARE LIBRE EN LA ADMINISTRACIÓN DE LA FACULTAD DE CONTADURÍA Y CIENCIAS ADMINISTRATIVAS DE LA UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO

TESINA

QUE PARA OBTENER EL TITULO DE

LICENCIADO EN INFORMATICA ADMINISTRATIVA

PRESENTA:

RAFAEL AYALA AHUJA

ASESOR DE TESINA

M.A. MA. HILDA RODALES TRUJILLO

MORELIA MICHOACAN SEPTIEMBRE 2012

AGRADECIMIENTOS

Doy gracias a Dios por darme todos los días la oportunidad de vivir y seguir adelante.

A mi papá Eleazar, mi mamá Delfina y mi hermana, por creer en mi, por darme el apoyo, el cariño, la paciencia por estar en todo momento conmigo y brindarme los consejos, la orientación necesarios para seguir adelante cada día y el coraje de creer que se pueden realizar esos propósitos en mente y luchar por ellos, a estos seres a quienes sus vidas las fueron dedicando para darme lo mejor, no tengo con que pagar lo que han hecho por mi, los quiero mucho estoy y estaré por siempre agradecido.

Agradezco a Álvaro por ayudarme indirectamente con sus comentarios y enseñanzas que me hicieron esclarecer muchas cosas que no entendía.

Agradezco a todos y cada uno de las personas que a través de sus conocimientos me ayudaron ampliar un poco más mis conocimientos.

Agradezco a mi asesora M.A Ma. Hilda Rodales Trujillo por haberme dado la oportunidad y asesoría para realizar esta tesina.

ÍNDICE

INTRODUCCIÓN

EL tiempo en el que utilizar un software de paga por el simple hecho que genero un costo era atractivo utilizarlo, hoy en día ya no lo es tanto en las organizaciones porque entre menor sea el costo y teniendo el mismo beneficio es mejor. Por varios años se ha utilizado software propietario, por ser el de mayor difusión y uso que se tiene, por lo que es importante saber qué es software libre y que gira en torno a lo que se conoce como GNU/Linux.

Contamos hoy con miles de aplicaciones, teniendo un gran repertorio de software libre equivalente al software propietario, y pueden ser utilizados independientemente de los conocimientos en informática de los usuarios. A todo esto la globalización y la necesidad de conocer y utilizar estas tecnologías en los ambientes de trabajo hacen que el software libre tenga más aceptación por no tener ningún costo, obteniendo así los mismos y mejores resultados.

En el presente trabajo se da la tarea de dar a conocer los conceptos generales del software, los antecedentes del software libre, ventajas y desventajas contra el software propietario, quienes fueron los fundadores, casos de éxito en su uso e implementación y el porqué el gobiernos deberían optar por políticas en uso al software libre.

Todo esto nos permite conocer con mayor amplitud el software libre con la finalidad de dar a conocer que es factible el uso del software libre en la facultad de Contaduría y Ciencias Administrativas en el ambiente administrativo, teniendo en cuenta que hay casos de éxito en otros países implementado estos sistemas.

PROBLEMÁTICA

En los últimos años se ha oído hablar mas del Software Libre y Open source, de como otros países han ido desarrollado políticas para promoverlo y usarlo en todos los ámbitos de la sociedad, especialmente en las administraciones publicas divulgando los beneficios que esto puede tener eficientando así el uso del recurso publico.

Pero para que esto conlleve a una realización, uno de los problemas o quizá el más fuerte es el desconocimiento, la mala información y el poco interés de quienes tomas las decisiones, aunándolo a los convenios que se tiene con empresas de software y que estas saben bien manejar su negocio realizando todo tipo de eventos, regalías diversas, por lo tanto los responsables técnicos prefieren no arriesgar y seguir contratando y comprando a las grandes marcas y consultoras que en caso de fracaso siempre justifican su decisión en que eligieron a la grande, alagándola de que es la mejor que hay en el mercado. Por otro lado aqueja el alto grado de utilización de sistemas privativos desde la primaria hasta el término de la carrera, desconociendo otras tecnologías como el software libre por lo tanto la mayoría de los trabajadores de la facultad de Contaduría Y Ciencias Administrativas hace más difícil la adopción de una alternativa para la elaboración de su trabajo diario.

OBJETIVOS

El objetivo general es:

Determinar el grado de impacto que se tendrá al implementar el uso de software libre en la administración de la facultad de Contaduría y Ciencias administrativas de la Universidad Michoacana de San Nicolás de Hidalgo. Con esto se intenta demostrar que se puede implementar software libre en unidades administrativas teniendo en cuenta el personal administrativo que es un factor a tomar en cuenta.

Para alcanzar el objetivo general se tienen los siguientes objetivos específicos.

- Recopilar información dentro de la facultad si se utilizan algún tipo de software libre en su administración, mediante la aplicación de entrevistas al encargado del centro de cómputo y a los empleados.
- Identificar los factores que inciden negativamente en la implementación del software libre.
- Dar a conocer los casos de éxito al implementar software libre en la administración.

JUSTIFICACIÓN

En la actualidad el uso de herramientas del software libre está siendo vista de buena manera en la administra pública de varios países, así como en organismos privados, implementando y obteniendo más que un ahorro al no pagar licencias de software propietario, beneficios para el desarrollo de estas organizaciones. Teniendo así menor costo de servicios, flexibilidad, innovación, y crecimiento.

Con el propósito de adquirir conocimiento sobre software libre por el gusto y necesidad de aprender decidí investigar y sobre la investigación dar las ventajas que se pueden tener en la administración de la Facultada de Contaduría y Ciencias Administrativas así como las desventajas que esto puede llevar. Elegí el tema porque el software libre hoy en día es muy usado tanto en organismos públicos y privados satisfaciendo sus necesidades.

Capítulo 1. Conceptos fundamentales de software.

1.1. Definición de software.

La definición más formal de software se le atribuye a la IEEE (Instituto de Ingenieros Eléctricos y Electrónicos), en su estándar 729: Es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de cómputo (IEEE, 1993). Software hace referencia a toda la parte inmaterial incorporada al equipo que permite su funcionamiento lo que lo convierte en una maquina de cálculo de uso general, capaz de realizar los mas complicados trabajos (Pardo, 1993). Es decir, abarca todo lo intangible, todo lo "no físico". La teoría que forma la base de la mayor parte del software moderno fue propuesta por vez primera por Alan Turing, "Los números computables", con una aplicación al problema de decisión (Turing, 1936).

1.2. Definición de software libre.

La definición de la Free Software Fundation (FSF), significa que el software respeta la libertad de los usuarios y la comunidad. En términos generales, los usuarios tienen la libertad de copiar, distribuir, estudiar, modificar y mejorar el software. Con estas libertades, los usuarios (tanto individualmente como en forma colectiva) controlan el programa y lo que hace (FSF, 2007).

Software libre es una cuestión de libertad, no de precio. Para comprender este concepto, se debe pensar en la acepción de libre como en “libertad de expresión”. Software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. Se refiere especialmente a cuatro clases de libertad para los usuarios de software:

- Libertad 0: la libertad para ejecutar el programa sea cual sea el propósito.
- Libertad 1: la libertad para estudiar el funcionamiento del programa y adaptarlo a las necesidades. El acceso al código fuente es condición indispensable para esto.

- Libertad 2: la libertad para redistribuir copias.
- Libertad 3: la libertad para mejorar el programa y de distribuir el programa modificado. También exige el código fuente.

Por lo tanto un programa puede definirse como software libre solamente si los usuarios tienen todas estas libertades. Cuando hablamos de software libre, se debe evitar utilizar expresiones como “regalar” o “gratis”, ya que se puede caer en el error de interpretarlo como una mera cuestión de precio y no de libertad. (Stallman, 2004).

1.2.1. Open source.

Open source (Código abierto). Este término es utilizado a menudo para referirse al software libre. Sin embargo, sus criterios son algo menos determinantes. El código abierto acepta algunas clases de restricciones que el software libre ha rechazado. El significado de código abierto se refiere a la accesibilidad del código fuete, aunque tiene licencias restrictivas sobre ese código. Este término comenzó a utilizarse en 1998 y rápidamente fue asociado a una filosofía, unos valores e incluso unos criterios para los cuales las licencias eran aceptables. El movimiento del software y el movimiento de código abierto son hoy movimientos separados con distintas visiones y metas, aunque trabajan juntos en algunos proyectos prácticos. Para el movimiento del código abierto la pregunta de si el software libre debe tener código abierto es una pregunta práctica, no ética. Según este movimiento el software no libre es una solución. En cambio, el movimiento del software libre es un movimiento social, para el cual el software libre es la solución. La definición oficial de código abierto está muy cerca de la definición de software libre, sin embargo es un poco mas débil en algunos aspectos y han aceptado algunas licencias restrictivas para los usuarios que el software libre no acepta (Andoni y Carmen, 2004).

En 1998 se creó la Open Source Iniciative (OSI). La OSI nació con el objetivo de crear y fomentar el uso de programas informáticos de código abierto. La razón por la que la OSI defiende el acceso al código fuente de los programas no es la libertad, sino la posibilidad de crear mejor software, adaptándolo a necesidades reales de los usuarios.

El resultado final es que según la OSI, el software de código abierto a de ser de mejor calidad que el privativo (Andoni y Carmen, 2004).

Para que un programa pueda ser considerado de código abierto, la OSI establece un decálogo de condiciones que ha de cumplir. En su versión 1.9 (OSI, 2012).

- 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 sólo 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.

Esto significa que los usuarios pueden combinar software libre y open source según sus necesidades. Este decálogo es compatible con las cuatro libertades del software libre.

1.3. Definición de Software propietario.

El software no libre también es llamado software propietario, 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 aún si el código fuente es hecho público, cuando se mantiene la reserva de derechos sobre el uso, modificación o distribución (Culebro, et al, 2006).

Capítulo 2.- Antecedentes generales del software libre.

2.1. Orígenes del software libre.

La primera generación de computadoras aparece a finales de la década de 1940. Eran de enormes dimensiones y muy costosas. El poder computacional era muy pobre comparado con las computadoras de la actualidad. La relación entre el hardware y el software era demasiado estrecha, los programas se escribían de una manera bastante especializada (lenguaje de máquina) y por lo tanto, el concepto de software como una parte "independiente" del hardware se veía todavía muy lejano. Debido precisamente a la relación entre hardware-software, las personas que operaban las computadoras debían de poseer cierto nivel de conocimientos sobre el funcionamiento de las mismas, así como de los programas que necesitaban para hacerlas funcionar. En ese entonces no existían los usuarios convencionales, todos eran usuarios especializados, en su gran mayoría científicos o ingenieros (Franco y Martínez, 2007).

Entre esos usuarios expertos, era muy común que se diera el intercambio de programas así como el compartir mejoras hechas a los mismos. A estas mejoras en el software se les conoce como "hacks" y a estos primeros expertos o "gurús" de la programación se les empezó a llamar "hackers". Término que en la actualidad se ha ido desvirtuando, confundiéndolos con delincuentes informáticos. En general a los hackers les interesa conocer el funcionamiento detallado de los sistemas informáticos y de su seguridad, manteniendo una actitud ética. Algunos traspasan esta línea y se convierten en lo que la comunidad "hacker" ha denominado "cracker" (Franco y Martínez, 2007).

En los inicios de la computación, 40s y 50s, no existían las licencias de software. En los laboratorios de investigación, los programas circulaban como las ideas: libremente. Era absolutamente normal que un programa desarrollado por un equipo de programadores o investigadores se distribuyera a otros equipos de otras universidades y a cualquier otro lugar donde hiciera falta. Y nada había de raro en que este programa fuera modificado por otro equipo, y así sucesivamente. A día de hoy, cuando un ilustre matemático demuestra un teorema difícil, publica el resultado de sus investigaciones en

obras especializadas con el fin de ayudar al progreso de la ciencia. Todo el mundo tiene acceso a ello (Rohaut, 2010).

Pero el universo de la informática ha seguido otros derroteros. Pese a ser una ciencia, el fruto de las investigaciones en informática no se circunscribe al mundo de los universitarios. Rápidamente, las empresas vieron el inmenso interés de automatizar algunas de sus tareas, como la contabilidad, los pagos, etc. Con la compra de las primeras grandes computadoras de gestión, se necesitaron programas. Estos programas tuvieron que ser protegidos como secretos industriales: había nacido una nueva industria: la creación de programas. Con su entrada en la dinámica de las grandes empresas, la informática perdió rápidamente la inocencia y se hizo mucho menos libre. Se empezó a hablar de licencias, impuestos y tasas, derechos de autor (lo que no impide autorizar la copia según el caso), limitación de los derechos, prohibición de copiar, y mucho mas (Rohaut, 2010). Las grandes compañías, plantearon la necesidad de fijar una línea divisora entre el software libre y el software propietario.

Con el surgimiento del ARPANET (Advanced Research Projects Agency Network - Red de la Agencia de Proyectos de Investigación Avanzada, precursor del Internet) a finales de la década de los 60s la cual permitía la interconexión entre redes de computadoras de las diversas universidades empezó el surgimiento de la primera comunidad global que se alzaba sobre los valores y principios del software libre. Los grupos hasta entonces dispersos de “hackers”, pudieron a través de la red, sumar esfuerzos, intercambiar conocimientos y colaborar entre sí. Los proyectos involucraban cada vez más desarrolladores de software, quienes estaban dispersos geográficamente alrededor del mundo y utilizaban el correo electrónico como medio de comunicación para hacer llegar sus aportaciones (Franco y Martínez, 2007).

2.2. La aportación de Unix al software libre.

Actualmente. UNIX es un sistema operativo multiusuario, multitarea, portable, con distintos interpretes de comandos, multiprocesador, multicore, con compiladores propios, con entorno gráfico, etc. entre otras muchas características que lo definen, según la organización The Open Group. Esta institución está formada por un grupo de

empresas punteras en tecnología UNIX como HP, IBM o SUN entre otras, que se encargan de otorgar estándares en distintas ramas de la informática (Martínez, 2009).

UNIX, es un derivado de lo que inicialmente se denomino UNICS *(Uniplexed Information and Computing System).* Este proyecto tenia como objetivo la creación de un sistema operativo simple, cuyo fruto fue la creación de UNICS, que finalmente acabo derivando en UNIX. Este sistema fue inventado en los años 60 por los laboratorios BELL de AT&T. a través de un equipo formado por Ken Thompson y Denis Ritchie entre otros. La primera versión del sistema fue escrita en lenguaje ensamblador, pero las necesidades de portabilidad a otras arquitecturas de hardware hicieron cambiar este lenguaje por C (Martínez, 2009).

Unix es el perfecto ejemplo del trabajo que se puede efectuar cuando se encauzan todas las energías a la búsqueda de un ideal tecnológico. Cuando AT&T distribuye casi libremente en 1974 el código fuente del sistema operativo a las universidades porque, entre otras razones, no ve ningún futuro económico a su producto, no parece dudar del entusiasmo de los estudiantes, profesores e investigadores en informática. Esta primera comunidad pasará mucho tiempo modificando y mejorando el producto, subiendo todas las novedades a AT&T para que se integren al producto oficial. Tras el cambio de licencia en 1978, la energía de la comunidad se encauzó hacia el proyecto universitario BSD, dejando el Unix comercial de AT&T. Señalando que los más grandes progresos se hicieron con el Unix de Berkeley (Rohaut, 2010).

Actualmente existen dos vertientes de sistemas UNIX:

- La rama de AT&T, que se convertiría en System V.
- La rama de Berkeley o BSD, desarrollado por la universidad de california.

2.3. El proyecto GNU y Linux.

El proyecto GNU fue iniciado por Richard Stallman (Richard Matthew Stallman programador estadounidense y fundador del movimiento por el software libre en el mundo.) con el objetivo de crear un sistema operativo completamente libre: el sistema

GNU. El 27 de septiembre de 1983 se anunció públicamente el proyecto por primera vez en el grupo de noticias net.unix-wizards. Al anuncio original, siguieron otros ensayos escritos por Richard Stallman como el "Manifiesto GNU", que establecieron sus motivaciones para realizar el proyecto GNU, entre las que destaca "volver al espíritu de cooperación que prevaleció en los tiempos iniciales de la comunidad de usuarios de computadoras" (Franco y Martínez, 2007).

En oposición a la comercialización del software y más particularmente a la falta de disponibilidad del código fuente, Richard Stallman, entonces investigador en el MIT (Massachusetts Institute of Technology). Este movimiento se tradujo en la creación de un proyecto consistente en rescribir completamente un sistema operativo libre. El modelo seguido era Unix, y Richard Stallman llamó a su proyecto GNU que significa GNU's Not Unix. El proyecto GNU conoció rápidamente un gran éxito y muchas herramientas y aplicaciones de Unix se desarrollaron a partir de cero ("from scratch"). Sin embargo, el núcleo libre de este sistema, llamado "Hurd", no se desarrolló tan rápidamente. El proyecto GNU se limitó durante cierto tiempo a ser una gama de herramientas completa de Unix sin núcleo (Pons, 2005).

En 1990 el sistema operativo GNU estaba casi completo, el único componente que faltaba era el núcleo (kernel). Un año después, en 1991, Linus Torvalds, un estudiante finlandés frustrado por tener que usar MS-DOS y queriendo evitar las limitaciones de MINIX (pequeño UNIX de código abierto escrito con fines educativos por el profesor Andrew S. Tanenbaum, de la Universidad de Ámsterdam clon de que primero se publicó en enero de 1987), envió un mensaje por Internet al grupo de noticias comp.os.minix en el cual mencionaba que estaba trabajando en un versión libre similar a MINIX. Éste era un sistema operativo bastante reducido creado por Andrew Tanenbaum con fines didácticos, el sistema era bastante simple y con pocas funcionalidades (Franco y Martínez, 2007).

Torvalds ponía a disposición (en la red Internet) esta versión de Minix, para quien la quisiera usar y a su vez invitaba a realizar aportaciones que sirvieran para mejorarla. Por otro lado, en 1991, Linus Torvalds (un estudiante de sistemas en la Universidad de Helsinki, Finlandia) publica el código fuente preliminar del núcleo de Linux. Muchos

programadores comenzaron a contribuir con él. Aportando piezas de código con nuevas funciones y mejoras para el sistema. Ya para 1994 aparece la primera versión estable del núcleo. Se puede decir entonces que la dupla GNU/Linux se considera como "un sistema operativo para computadores que facilitan su uso y operación" (Bermúdez Silva, 2008).

2.4. Libertades del software libre.

De acuerdo con tal definición, el software es "libre" si garantiza las siguientes libertades:

Libertad	Descripción
0	La libertad de ejecutar el programa sea cual sea el propósito.
1	La libertad para modificar el programa para ajustarlo a tus necesidades. (Para que se trate de una libertad efectiva en la práctica, deberás tener acceso al código fuente, dado que sin él la tarea de incorporar cambios en un programa es extremadamente difícil.)
2	La libertad de redistribuir copias, ya sea de forma gratuita, ya sea a cambio del Pago de un precio.
3	La libertad de distribuir versiones modificadas del programa, de tal forma que la comunidad pueda aprovechar las mejora introducidas.

Tabla 1: Libertades del Software Libre.

Software libre es cualquier programa cuyos usuarios gocen de estas libertades. De modo que se obtiene la libertad de redistribuir copias con o sin modificaciones, de forma gratuita o cobrando por su distribución, a cualquiera y en cualquier lugar. Gozar de esta libertad significa, entre otras cosas, no tener que pedir permiso ni pagar para ello. Asimismo, se obtiene la libertad para introducir modificaciones y utilizarlas de forma privada, sin siquiera tener que mencionar su existencia. Si se decide publicar estos cambios, no se está obligado a notificárlo a ninguna persona ni de ninguna forma en particular. La libertad para utilizar un programa significa que cualquier individuo u organización podrán ejecutarlo desde cualquier sistema informático, con cualquier fin y sin la obligación de comunicárselo subsiguientemente ni al desarrollador ni a ninguna

entidad en concreto. La libertad para redistribuir copias supone incluir las formas binarias o ejecutables del programa y el código fuente tanto de las versiones modificadas, como de las originales, ya que se obtiene la libertad para redistribuir tales formas si se encuentra el modo de hacerlo, pues las libertades para hacer cambios y para publicar las versiones mejoradas requieren de la accesibilidad de código fuente, por supuesto de manera libre, condición necesaria del software libre. Cuando se habla de software libre, se debe evitar utilizar expresiones como “regalar” o “gratis”, ya que se puede caer en el error de interpretarlo como una mera cuestión de precio y no de libertad (Stallman, 2004).

2.4.1. La ambigüedad de “free”

Software Libre no siempre es gratis. El mercado de software, como producto de ingeniería que es, también cuenta con un amplio elenco de profesionales especializados. Posibilita cargar adaptaciones, mejoras personalizadas o bien adquirir soporte técnico. Servicios por los que hay que pagar, como diariamente se hace a cualquier otra empresa cuando requieres sus productos o servicios. El software gratuito se posible conseguirlo gratis, pero no es Software Libre, ya que no permite que nadie, salvo su creador, pueda ver cómo está hecho y mejorarlo.

Mientras que el software gratuito no ofrece ninguna garantía de estar libre de virus, el Software Libre permite que los expertos en seguridad puedan ver cómo está hecho y certificar que el programa es seguro. El Software Libre garantiza en todo momento sólo pagar por lo que consume. No se necesitas pagar veinte veces por el soporte CDROM, una caja, un manual, o quizás una licencia.

En muchos casos los servicios de valor añadido que podrían justificar la compra de una licencia de software no libre, como la puesta a punto o la asesoría telefónica, suelen ser más baratos por la libre competencia garantizada y el conocimiento profundo potencial que cualquier profesional puede alcanzar sobre cualquier aplicación de Software Libre (Perez i Noguer).

2.5. Comparación del software libre y Software de Fuente abierta.

Actualmente hay dos movimientos importantes referentes al software que se define como libre. Por un lado está el Movimiento de Software Libre (Free Software Movement) y por el otro está el Movimiento de Software de Fuente Abierta (Open Source Movement). Ambos tienen grandes similitudes, pero también tienen grandes diferencias. En breve se explicaran cada uno de estos movimientos (Cerón Estrada, 2008).

2.5.1 El Movimiento de Software Libre

El Movimiento de Software Libre fue el primero de todos, inició con Richard M. Stallman a mediados de la década de los 80. Stallman pensaba que las personas que utilizan software deberían tener 4 libertades: libertad de uso, libertad de estudio y modificación, libertad de redistribución y libertad de redistribuir las modificaciones. Entonces creó la Fundación para el Software Libre (Free Software Fundación) y el Proyecto de GNU los cuales se constituyeron en el pilar del Movimiento de Software Libre. El movimiento de Software Libre está basado en una filosofía y una ideología bien definida, que propende por valores como la libertad, el compartir, y el tener una comunidad colaborativa donde todos se beneficien. La libertad en el software se pone por encima de todo, es decir que se considera como el mayor benefició del software libre el hecho de su propia libertad, más allá de aspectos técnicos o prácticos. Los seguidores del Movimiento del Software Libre están en rotundo desacuerdo con el modelo de software propietario, e instan a las personas a que únicamente usen software libre. El movimiento materializó su definición de software libre con la Licencia Pública General de GNU (GNU General Public License, GPL). Y plantea una estrategia legal para, no solo desarrollar software libre, sino también para protegerlo de convertirse en propietario; esta estrategia se conoce como copyleft, y significa que el usuario del software obtiene las cuatro libertades mencionadas anteriormente, pero tiene una sola restricción: no pude de ninguna manera convertir el software en propietario, ni si mismo ni sus derivados. Para dicha protección se hace uso de los tratados internacionales de copyright. El término copyleft es solo un juego de palabras que significa que es contrario a lo que, por lo general, se trata de hacer un copyright (Cerón Estrada, 2008).

2.5.2 El Movimiento de Software de Fuente Abierta

Para 1998, muchas personas alrededor del mundo se habían unido al Movimiento del Software Libre y gracias a la aparición de Linux y su acoplamiento con el Proyecto GNU, por fin había un sistema operativo funcional completamente libre: GNU/Linux. Algunas personas empezaron a notar que GNU/Linux crecía a pasos agigantados, como una bola de nieve. Entonces se dieron cuenta que el sistema de desarrollo de software libre podría convertirse en una nueva estrategia de negocios que brindaría a las empresas una gran ventaja competitiva y además fortalecería a la comunidad. El problema era que las empresas estaban distantes del Movimiento del Software Libre y no confiaban en que pudieran hacer negocios con este sistema. Esto se debía a que el término en ingles para software libre, free software, tiene en realidad dos significados: software libre y software gratis. La palabra gratis sumada al gran contenido ideológico (que contrasta con las políticas expansionistas de las grandes empresas) del Movimiento de Software Libre asustaba a la industria. Fue entonces cuando nació el Movimiento de Software de Fuente Abierta, que trata de ser un "plan de mercadeo para el software libre" con el cual se pueda convencer a las empresas. Para ello han decidido tomar las características prácticas que plantea el software libre, que representan ventajas competitivas para la industria como por ejemplo su revolucionario modelo de desarrollo y de distribución y dejar a un lado los aspectos éticos e ideológicos del Movimiento del Software Libre. Este movimiento no ha creado una licencia especifica que materialice su definición de Software de Fuente Abierta, en cambio, ha creado unas serie de criterios a modo de meta-licencia, las cuales, según ellos, debería cumplir todo software se denomine como de Fuente Abierta. Los criterios son: disponibilidad del código fuente, libre distribución, posibilidad de modificación y trabajos derivados, respeto del autor y su integridad, no discriminación de uso para personas o grupos, no discriminación de ningún campo o disciplina, la licencia debe aplicarse a todo el software, licencia no debe estar atada a algún producto, no debe restringir otro software, y no debe estar atado a ninguna tecnología en especial. El movimiento realiza certificaciones a todas las licencias que cumplen con dichos criterios, así por ejemplo la licencia GPL del Movimiento de Software Libre tiene una certificación de este tipo. El Movimiento de Software de Fuente abierta, es además más

flexible que el Movimiento de Software Libre en algunos aspectos, por ejemplo sus seguidores no piensan que deben usar sólo software de fuente abierta, sino que también pueden usar software propietario, es más, piensan que el software propietario debe seguir existiendo en convivencia con el software libre. También es más flexible en el sentido de que dentro de sus criterios para definir una licencia como de fuente abierta, no requieren el copyleft y aceptan por ejemplo la licencia BSD que no requiere que sus derivados sean libres. Aunque tiene sus diferencias, en realidad ambos movimientos van hacia el mismo lado. El Movimiento Software Libre no considera al Movimiento de Fuente Abierta como un enemigo ni viceversa. Lo importante no es decidir de que lado se está, sino entender que son movimientos diferentes, que coinciden en gran parte y solo difieren en algunos principios básicos (Cerón Estrada, 2008).

2.6 Tipos de licencia del software libre.

El software libre dispone de licencias para su uso, una licencia se define como una forma de contrato mediante la que el titular de los derechos de autor establece las condiciones y términos bajo los que el usuario puede utilizar un determinado programa informático. Cuando se refiere a las licencias de los programas informáticos, el contrato se realiza entre el creador del programa y las personas que lo van a utilizar (Niño Camazón, 2011).

Se clasifica en dos grupos: Licencias permisivas y Licencias robustas.

2.6.1 Licencias robustas (Copyleft)

Este tipo de licencias de software libre, mantienen durante toda la cadena de redistribución lo que se conoce como Copyleft que trata de mantener los mismos derechos de la obra original a través de toda la cadena, incluyendo las derivaciones producidas sobre la misma. Es decir, este tipo de licencias obliga a los usuarios el mantenimiento de la licencia original que el autor original de la obra puso a su software, permitiendo así que el software derivado del original siga siendo tan libre y con los mismas características que pose la original, prohibiendo así que se produzca el hecho que el anterior tipo de licencias permitía, como es que el software derivado del original,

en algún punto de la cadena de redistribución se convierta en software propietario. Licencias de este tipo son por ejemplo GNU GPL (GNU General Public, Licencia pública general de GNU), LGPL (GNU Lesser General Public License, Licencia pública general reducida de GNU), AGPL (GNU Affero General Public License, Licencia pública general de Affero de GNU) (Montes L., 2011).

2.6.1.1 Licencia Pública General (GPL)

Es la principal licencia de la FSF, es la más utilizada, se encuentra en su versión 3 (desde 2007). Su principal característica es la incorporación del concepto de Copyleft. El Copyleft es un método general para hacer libre a un programa, y obligar a que toda modificación o extensión de este programa sean también libre. Es decir, la intención del Copyleft es mantener las libertades del software libre, incluso en las modificaciones o trabajos derivados. Por esto se dice que este tipo de licencias son virales, ya que el Copyleft se va extendiendo por cada modificación o trabajo derivado (Montes L., 2011).

2.6.1.2 La LGPL (Licencia Pública General Reducida de GNU)

Este tipo de licencias posee un copyleft de grado menor, esto debido a que permite que el software se integre con otros módulos que son software no libre. Es especialmente utilizado en librerías y en los siguientes casos: Cuándo se quiera utilizar el software en programas privativos. (Cuando se quiera sacar un beneficio económico).Otro caso es cuándo ya existe software libre que ofrezca las mismas ventajas que ofrece el software que se creó. Compatible con todas las licencias GPL, se puede cambiar la licencia a GPL (Montes L., 2011).

2.6.1.3 Licencia Affero

La Licencia Pública General de Affero (Affero GPL o AGPL) es una licencia copyleft derivada de la Licencia Pública General de GNU diseñada específicamente para asegurar la cooperación con la comunidad en el caso de software que corra en servidores de red. La Affero GPL es íntegramente una GNU GPL con una cláusula nueva que añade la obligación de distribuir el software si este se ejecuta para ofrecer servicios a través de una red de computadoras. La Free Software Foundation

recomienda que el uso de la GNU AGPLv3 sea considerado para cualquier software que usualmente corra sobre una red (Montes L., 2011).

2.6.2 Licencias permisivas o (minimalistas)

Son un tipo de licencias de software libre que otorgan mucha libertad a la redistribución del software auspiciado con algún tipo de licencia de este tipo, pudiendo hacerlo en forma de software libre o en forma de software propietario, siendo software libre la licencia original con la que el autor licenció su obra. Es decir, tenemos una aplicación que nosotros como autores publicamos bajo una licencia de este tipo, bien, nuestro software es libre, ahora, este software es utilizado por ejemplo, por una migo nuestro, el cual le hace una serie de cambios, y decide, que ya que la licencia que tiene ese software permite hacerlo, publicarlo bajo una licencia de software privativo, cerrando el código fuente y eliminando cualquiera o todas de las libertades anteriormente dichas. Esto se puede hacer porque este tipo de licencias son muy permisivas en el aspecto de la redistribución, suelen basarse en textos muy sencillos, dando mucha libertad a quien recibe el código. Tipos de licencias de este tipo son por ejemplo MIT, BSD, Apache, MPL (Montes L., 2011).

2.6.2.1 Licencia MIT

La licencia MIT, también conocida como MIT/X11 debido a que el MIT (Instituto Tecnológico de Massachusetts) utiliza muchas otras licencias, permite casi una libertad total. En este aspecto es muy similar a la licencia copyleft con algunas modificaciones. La licencia MIT añade una serie de cláusulas que protegen en parte al autor, ésta cláusula se puede saltar fácilmente. Una característica importante es que esta licencia introduce un blindaje ante garantías implícitas y expresas, con lo que en principio el autor está legalmente protegido contra ellas. Además establece algunas cláusulas más, como que el software no tiene por adaptarse con ningún propósito en particular o el autor no tiene que hacerse cargo de ningún daño derivado del uso del software. Aunque añade un conjunto de cláusulas interesantes: Permite productos derivados completamente privativos. Se permite re-licenciar el código. Se puede usar el renombre de los autores para promocionar, por ejemplo, el software derivado. La documentación

tiene que tener la misma licencia, o al menos aparecer en ella. No obliga a distribuir el código fuente. Finalmente, se puede decir que esta licencia se considera una meta licencia. Las meta licencias son aquellas que sirven de plantilla para generar licencias similares siguiendo los mismo criterios pero que no son compartidas por distintos proyectos de software libre, ya que difieren ligeramente. Es decir, cada vez que utilizamos la licencia, lo que estamos haciendo realmente es crear una licencia basada en esta para licenciar nuestro software (Montes L., 2011).

2.6.2.2 BSD (Berkeley Software Distribution)

La licencia BSD nació en la Universidad de California. Esta licencia en sus orígenes contenía una cláusula que obligaba a que en todos los productos donde se utilizara se incluyera a la Universidad de California como autora de la aplicación. Las licencias BSD que incluyen esta cláusula son conocidas como la vieja licencia BSD. Es una licencia de software libre y permisivo que no contiene copyleft, permite el uso de código privativo dentro de él, esta está muy cerca del dominio público, el cual carece de protección además de ser totalmente permisiva y sin copyleft, no es actualizable; es también una meta licencia (Montes L., 2011).

2.6.2.3 Apache v2.0

Da permiso para usar el software para cualquier propósito, distribuirlo, codificarlo y distribuir las modificaciones. No tiene copyleft: No requiere que las versiones modificadas tengan que ser distribuidas como software libre (Montes L., 2011).

2.6.2.4 Mozilla Public License (MPLv1.1)

No es compatible con ninguna de las licencias GPL. Todas las modificaciones al código fuente han de ser redistribuidas bajo la misma licencia que el código fuente del que derivan. Tiene copyleft, aunque se permite la distribución de binarios bajo licencias diferentes. Permite licenciamiento múltiple del programa o partes de él. Si se usa GNU GPL como licencia alternativa, esa parte del programa será compatible con GPL. Clausulas de no garantía y limitación de responsabilidad. Incluye cláusulas explícitas

sobre el uso de patentes: Provisión de terminación de los derechos cedidos por la licencia en caso de litigación. Hay numerosas licencias basadas en MPL: Common Development and Distribution License, Sun Public License, gSOAP Public License, AROS PublicLicense, SugarCRM Public License, Terracotta Public License, Common Public AttributionLicense, Erlang Public License (Montes L., 2011).

2.6.3 Licencias Libres para Documentación

Existen varias Licencias para Documentación como por ejemplo: GFDL, Creative Commons.

2.6.3.1 GFDL(GNU Free Documentation Libre Licence)

Esta licencia fue creada por la FSF igual que la GPL en software, permite al usuario el derecho de copiar y redistribuir un determinado trabajo pero exige que todas las copias y derivados queden disponibles bajo la misma licencia. Las copias incluso pueden ser comercializadas, aunque si esto ocurre en grandes cantidades (más de 100) la documentación original habrá de estar disponible para el usuario final (Montes L., 2011).

2.6.3.2 Creative Commons

Creative Commons es una organización no gubernamental, cuyo objetivo es ayudar a promover la creatividad, reduciendo sus barreras legales, por medio de nuevas tecnologías y nueva legislación. Crearon las licencias Creative Commons inspirada en la GPL (General Public License) de GNU, su fin es proporcionar un modelo legal para facilitar la distribución y uso de contenidos para dominio publico (Montes L., 2011).

Rank	Licencia	%
1.	GNU General Public License (GPL) 2.0	38.23%
2.	MIT License	12.08%

3.	GNU General Public License (GPL) 3.0	8.60%
4.	Apache License 2.0	7.94%
5.	Artistic License (Perl)	7.24%
6.	BSD License 2.0	7.02%
7.	GNU Lesser General Public License (LGPL) 2.1	6.45%
8.	GNU Lesser General Public License (LGPL) 3.0	1.81%
9.	Code Project Open 1.02 License	1.74%
10.	Microsoft Public License (Ms-PL)	1.63%
11.	Mozilla Public License (MPL) 1.1	1.08%
12.	Eclipse Public License (EPL)	0.99%
13.	BSD Two Clause License	0.37%
14.	Common Public License (CPL)	0.34%
15.	Common Development and Distribution License (CDDL)	0.33%

Tabla 2: Estadística tipo de licencias. Base de Conocimiento incluye más de 570.000 proyectos de más de 5.400 sitios y se actualiza con miles de nuevos proyectos sobre una base regular (Black Duck Software, Inc., 2012).

2.7. Distribuciones GNU/Linux

Hay tres elementos de software principales que componen un sistema GNU/Linux:

1) El kernel Linux: es tan sólo la pieza central del sistema. Pero sin las aplicaciones de utilidad, shells, compiladores, editores, etc. no se podría tener un sistema entero.

2) Las aplicaciones GNU: en el desarrollo de Linux, éste se vio complementado con el software de la FSF existente del proyecto GNU, que le aportó editores (como emacs), compilador (gcc) y utilidades diversas.

3) Software de terceros: normalmente de tipo de código abierto en su mayor parte. Todo sistema GNU/Linux se integra además con software de terceros que permite añadir una serie de aplicaciones de amplio uso, ya sea el propio sistema gráfico de X Windows, servidores como el de Apache para web, navegadores, etc. Así mismo, puede ser habitual incluir algún software propietario, dependiendo del carácter libre que en mayor o menor grado quieran disponer los creadores de la distribución.

Al ser la mayoría del software de tipo de código abierto o libre, ya sea el kernel, software GNU o de terceros, normalmente hay una evolución más o menos rápida de versiones, ya sea por medio de corrección de errores o nuevas prestaciones. Esto obliga a que en el caso de querer crear un sistema GNU/Linux, se tenga que escoger qué software se quiera instalar en el sistema, y qué versiones concretas de este software (Jorba Esteve & Suppi Boldrito, 2004).

Una distribución es una variante del sistema GNU/Linux que se enfoca a satisfacer las necesidades de un grupo específico de usuarios, dando así origen a ediciones domésticas, empresariales y para servidores. La mayoría de las distribuciones están, en mayor o menor medida, desarrolladas y dirigidas por sus comunidades de desarrolladores y usuarios.

Actualmente existen más de 300 distribuciones de GNU/Linux y su número es creciente en la medida en que cada vez resulta más fácil hacer una distribución propia a partir de las existentes.

2.7.1 Distribuciones más conocidas.

Slackware: Fue fundada en 1993 y desde entonces ha sido mantenida activamente por Patrick J. Volkerding. Es una de las más antiguas. Incluso se entregaba en un disquete. Durante los primeros años de vida de Linux, la Slackware era la distribución de

referencia para aprender a utilizar Linux. Es extremamente austera: su instalador se reduce a la mínima expresión y casi todas las configuraciones deben establecerse a mano, sin asistente. No cuenta con un gestor de paquetes (se trata de simples archivos de ficheros comprimidos). Todo ello hace que sea ideal para los manitas y los apasionados de Unix, pero no tanto para los principiantes (Rohaut, 2010).

Debian: El proyecto Debían fue creado por Ian Murdock en 1993, en una época en la cual la idea misma de distribución Linux estaba todavía en pañales. El nombre de Debian proviene de Debra (la esposa de Murdock) e Ian. Durante mucho tiempo, Debian ha sido la única distribución entera y únicamente compuesta de programas libres y Open Source, lo que le sigue valiendo el nombre oficial de Debian GNU/Linux distribuida y mantenida por una red de desarrolladores voluntarios con un gran compromiso por los principios del software libre (Rohaut, 2010).

Red Hat Enterprise Linux: Si existe una empresa comercial en el mundo de Linux que influyó y sigue marcando época, es la empresa Red Hat. Fundada en 1995 por Robert Young y Marc Ewing, edita la famosa distribución epónima cuya primera versión oficial data de 1994 (la empresa se fundó tras el lanzamiento de la distribución). El sistema de paquete RPM apareció con la versión 2.0. Fue tal la buena acogida de Red Hat que lleva casi veinte años siendo la referencia. Cada versión era innovadora tanto en la integración de los programas como en su instalador (llamado «anaconda») y sus herramientas de configuración (Rohaut, 2010).

Las distribuciones se diferencian entre si según como hayan sido construidas las aplicaciones, sus métodos de administración y manejo del sistema. Cada una de ellas puede tener una enorme variedad de aplicaciones, entre ellos, entornos gráficos, suites ofimáticas, navegadores web, gestores de correo, clientes de mensajería instantánea; por nombrar algunos tipos de aplicaciones. La base de cada distribución incluye el núcleo Linux con los programas y librerías del proyecto GNU.
Como ya se comentó, existen cientos de distribuciones de Linux, creadas con distintos propósitos y fines. Entre las más destacadas se encuentran: (Bermúdez Silva, 2008)

Nombre	Empresa	Creador	Año	País
Slackware	Ninguna	Patrick Volkerding	Julio 1993	EEUU
Debían	El Proyecto Debian	Ian Murdock	Agosto 1993	Mundial
Redhat	Red Hat Software Inc.	Bob Young y Marc Ewing	Noviembre 1994	EEUU
Knoppix	Ninguna	Klaus Knopper	Enero 2003	Alemania
Mepis	Ninguna	Warren Woodford	Mayo 2003	EEUU
Fedora Core	El Proyecto Fedora	N/D	Noviembre 2003	EEUU
CentOS	El equipo de desarrollo de CentOS	N/D	Marzo 2004	Mundial
Ubuntu	Canonical Ltd.	Mark Shuttleworth	Julio 2004	Sudáfrica

Tabla 3: Cuadro cronológico de algunas distribuciones importantes

Algunas de estas distribuciones son pioneras en el mundo de Linux, mientras otras son derivadas de las primeras, o son subderivadas de las anteriores. Esto ocurre porque se generan nuevos proyectos a partir de otros, lo cual es perfectamente normal (Bermúdez Silva, 2008).

Capítulo 3. Copyright, Copyleft y Patentes.

3.1 Copyright.

El término “derechos de autor” (copyright), que en un principio se refería literalmente al “derecho de copiar” las obras literarias de un autor, pretendía estimular la publicación de obras escritas. Los derechos de autor eran puestos en vigor obligando a las casas editoriales a obtener el permiso de los autores para re-imprimir su obra. Estos derechos proporcionan un beneficio público a un bajo costo para los ciudadanos y funcionaron razonablemente bien hasta que los avances en la reproducción digital empezaron a permitir hacer copias de alta calidad a un costo muy bajo. Desde entonces, para proteger sus productos, los propietarios de derechos comenzaron a adoptar medidas drásticas, algunas de las cuales violan indiscutiblemente derechos ya establecidos como el de uso legal por ciertos usuarios. Como anota Richard Stallman (2000), fundador del proyecto de fuente abierta GNU: “lo que alguna vez fue una norma para regular la industria editorial, se ha convertido en restricción para el público al que se pretendía servir” (Henson-Apollonio, Longhorn, & White, 2002).

Derechos de autor (copyright). Los derechos de autor otorgan ciertos derechos a los creadores de obras artísticas y literarias; tales como libros, dibujos y pinturas. El concepto se ha ampliado para abarcar programas de computadora, mapas, imágenes y bases de datos. Por lo general, el software de computadora está protegido por derechos de autor o por acuerdos de licenciamiento. Aunque la Convención de Berna no menciona el tema, el Artículo 4 del Tratado WIPO (1996) sobre los Derechos de Autor establece que los programas de computadora se deben proteger como obras literarias, según el Artículo 2 de la Convención de Berna, “…cualquiera que sea la forma o modo de expresión”. El Artículo 10 (1) del Acuerdo WTO TRIPS también declara que “los programas de computadora, ya sean en código fuente o de objeto, deberán protegerse como obras literarias según la Convención de Berna (1971)”. Así, todas las naciones que ratificaron el Tratado WIPO o el Acuerdo WTO TRIPS tienen que permitir que los programas de computadora sean protegidos mediante los derechos

de autor. La directiva respecto a los derechos de autor para software de computadora de la Unión Europea (Comisión Europea, 1991), exige que los estados miembros protejan los programas de computadora con derechos de autor como obras literarias, según los términos de la Convención de Berna. El término "programas de computadora" abarca también el material de diseño, incluido el código fuente. La directiva estipula también que un programa de computadora se protege si "…es original en el sentido de que es una creación intelectual propia del autor" y que no se debe aplicar ningún otro criterio. La protección cubre la expresión de cualquier forma del programa de computadora, pero no las ideas ni los algoritmos en que éste se basa. (Henson-Apollonio *et al,* 2002).

3.1.1 Ventajas del Copyright.

- Si no se utilizara copyright muchos artistas podrían temer que su trabajo pudiera ser copiado y modificado sin reconocer el trabajo al artista inicial. Sin embargo, esto puede traer problemas: el trabajo del artista podría utilizarse de manera contraria a su voluntad, poniendo una fotografía estándar en un cartel racista. Si el artista es reconocido, será entonces asociado aparentemente con un grupo y una ideología que tal vez no comparta. Asimismo, tampoco hay garantía de que se le atribuya el mérito de su trabajo.
- El dueño o la persona que ha escrito una obra tiene derecho a cobrar por la misma, así como por sus reproducciones.
- Las compañías que distribuyen software propietario responden ante cualquier problema legal que se suscite respecto al posible reclamo de propiedad intelectual.
- Frente a problemas con los programas de los cuales se han comprado sus licencias propietarias, existe un responsable frente al cual se puede fincar alguna acción legal. (Rodríguez, 2008)

3.1.2 Desventajas del Copyright.

- Origina que en el mercado se generen monopolios, ejemplo de esto es el claro desarrollo de Microsoft. Los distribuidores de programas generan mercados

cautivos, ya que insertan problemas a sus propios programas a efecto de que se requiera alguna actualización.

- Debido al constante cambio tecnológico los programas se vuelven obsoletos sumamente rápido por lo que es necesario comprar las nuevas versiones de los programas, o bien, los nuevos programas que salgan al mercado; por lo que implica un gran gasto para los usuarios. Debido a las prácticas monopólicas que se ejercen actualmente respecto a los programas propietarios, las compañías se aprovechan de ello y fijan altos costos para sus productos, que en muchos casos son inaccesibles para muchas personas.
- Los altos costos derivados de las prácticas monopólicas de los software propietarios han contribuido a la proliferación de la piratería. (Rodríguez, 2008)

3.2 Copyleft.

El símbolo del copyleft es "ↄ", es decir, el símbolo del copyright invertido, viendo hacia la izquierda. Es utilizado como la contrapartida del símbolo del copyright, sin embargo no posee reconocimiento legal. El término copyleft describe un grupo de licencias que se aplican a una diversidad de trabajos tales como el software, la literatura, la música y el arte. Una licencia copyleft se basa en las normas sobre el derecho de autor, las cuales son vistas por los defensores del copyleft como una manera de restringir el derecho de hacer y redistribuir copias de un trabajo determinado, para garantizar que cada persona que recibe una copia o una versión derivada de un trabajo, pueda a su vez usar, modificar y redistribuir tanto el propio trabajo como las versiones derivadas del mismo. La licencia debe asegurar que el propietario del trabajo derivado lo distribuirá bajo el mismo tipo de licencia. Así, y en un entorno no legal, el copyleft puede considerarse como opuesto al copyright. Los vocablos ingleses "right" y "left" además significan "derecha" e "izquierda" respectivamente, lo que acentúa la diferencia entre ambos conceptos. Una posible traducción al español sería "izquierdos de autor", en contraste con los derechos de autor. En la práctica sin embargo el término se deja sin traducir. Los autores y desarrolladores usan el copyleft en sus creaciones como medio para que otros puedan continuar el proceso de ampliar y mejorar su trabajo (Rodríguez, 2008).

3.2.1 Ventajas del Copyleft.

- Cuando el copyleft rige un trabajo su eficiencia hace cumplir las condiciones de la licencia a todos los tipos de trabajos derivados.
- Este tipo de licencias es el que se utiliza generalmente para la creación de bibliotecas de software, con el fin de permitir que otros programas puedan enlazar con ellas y ser redistribuidos, sin el requerimiento legal de tener que hacerlo bajo la nueva licencia copyleft.
- El copyleft es aquel que permite que todas las partes de un trabajo (excepto la licencia) sean modificadas por sus sucesivos autores.
- El copyleft también ha inspirado a las artes, con movimientos emergentes como la "Free Society" y los sellos discográficos open-source. Por ejemplo, la Licencia Free Art es una licencia copyleft que puede ser aplicada a cualquier obra de arte.
- Han inspirado también la creación de las licencias Creative Commons "compartir igual" y la Licencia de Documentación Libre de GNU (Rodríguez, 2008).

3.2.2 Desventajas del Copyleft.

- El copyleft hace referencia a las licencias que no se heredan a todos los trabajos derivados, dependiendo a menudo de la manera en que éstos se hayan derivado.
- Se requiere distribuir los cambios sobre el software con "copyleft", pero no los cambios sobre el software que enlaza con él. Esto permite a programas con cualquier licencia ser compilados y enlazados con bibliotecas con copyleft tales como glibc (una biblioteca estándar requerida por muchos programas) y ser redistribuidos después sin necesidad de cambiar la licencia.
- El copyleft parcial implica que algunas partes de la propia creación no están expuestas a su modificación ilimitada, o visto de otro modo, que no están completamente sujetas a todos los principios del copyleft.
- El copyleft es más difícil de poner en práctica en aquellas artes que se caracterizan por la producción de objetos únicos, que no pueden ser copiados (a menos que no se tema por la integridad del trabajo original). Se puede ilustrar esta idea con el siguiente ejemplo: suponga que hay una exposición pública de

> algunos cuadros mundialmente famosos, algunas de las muchas copias y trabajos derivados que Andy Warhol hizo de sus propias obras de arte, y suponga que alguien que tiene acceso a esos cuadros (sin tener plena propiedad de los derechos de ´estos), decide "mejorarlos" con algunos efectos pictóricos de su gusto (sin olvidar la correspondiente firma con pintura de spray). Dada esta situación, no habría manera (legal) de detener a este tipo si le puede considerar el titular bajo copyleft de dichas obras (Rodríguez, 2008).

3.3 Patentes.

La OMPI (Organización Mundial de la Propiedad Intelectual) define la patente como: "Una patente es un derecho exclusivo concedido a una invención, es decir, un producto o procedimiento que aporta, en general, una nueva manera de hacer algo o una nueva solución técnica a un problema (Organización Mundial de la Propiedad Intelectual, 1967). En contraste con los derechos de autor, que se centran en las formas de expresión, las patentes protegen las invenciones útiles. Las patentes están reguladas bajo los términos de la Convención de París para la Protección de la Propiedad Industrial, 1883-1979 (WIPO, 2001b) y la Sección 5 del Acuerdo TRIPS (WTO, 1995). El TRIPS establece que las patentes deben tener un período de protección de al menos 20 años "a partir de la fecha de solicitud" (Artículo 33). (Henson-Apollonio *et al*, 2002).

Las patentes se solicitan no sólo para generar beneficios económicos y restringir el uso de los elementos patentados, sino también para asegurar el control de propiedad intelectual sobre técnicas, modelos y herramientas nuevas que, una vez protegidas, pueden ser proporcionadas de manera más manejable al público y a otros investigadores. Las patentes pueden proteger el marco conceptual que respalda una propiedad intelectual que ha sido reconocida como útil por la "reducción a la práctica". El peligro para las instituciones de investigación es que se le otorgue la patente de un elemento indispensable para su investigación, a una organización que permite el acceso sólo a quienes pagan los costos de la licencia correspondiente (Henson-Apollonio *et al*, 2002).

3.3.1 Patentes de programas de cómputo o software.

El software tiene dos componentes:

- Un componente escrito: el código.
- Un componente técnico: los algoritmos.

Las patentes de software protegen a ´este desde el componente técnico. Así, con las patentes se protegen las ideas, los algoritmos. Sobre la posibilidad de patentar los programas de computación la Organización Mundial de la Propiedad Industrial establece que: “Los requisitos sustantivos y de procedimiento para la concesión de patentes varían de un país a otro. En particular, varían significativamente las prácticas y la jurisprudencia vigentes en relación con la patentabilidad de las invenciones relativas a programas informáticos. Por ejemplo, en algunos países se entiende que las invenciones deben tener un carácter técnico con arreglo a la legislación de patentes, y no se consideran invenciones patentables los programas informáticos como tales, mientras que en otros no existe dicho requisito, por lo que los programas informáticos son por lo general materia patentable” (Culebro, et al, 2006).

Por otra parte, “los programas informáticos pueden protegerse en virtud del derecho de autor. Sin embargo, la protección del derecho de autor únicamente abarcara las expresiones, pero no las ideas, procedimientos, métodos de operación o conceptos matemáticos en sí.” (Organización Mundial de la Propiedad Intelectual, 1967).
“Permitir la patentabilidad de programas informáticos es tan absurdo como permitir patentar las fórmulas básicas de la matemática ya que es abrir la puerta a patentar algoritmos universales y básicos que pueden ser parte de cualquier programa.” (Masi Hernández, 2005).

Capítulo 4. Ventajas y desventajas del software libre y software propietario.

Una distinción técnica entre el software "propietario" y el software libre, es la apertura del código fuente de las aplicaciones junto con la distribución "binaria" (Zuniga, 2007).

En el software libre, se permite la distribución y modificación del software, pero manteniendo el espíritu original de las condiciones definidas por el autor, en términos de uso y distribución "libres" (sin restricción) (Zuniga, 2007).

En el caso del "software propietario", la distribución se restringe a quienes adquieren una "licencia de uso" (generalmente por la vía de un pago), para un "paquete de aplicación" o "producto". Pero la distribución se restringe al código "binario" (los programas ejecutables de las aplicaciones), quedando el código fuente en propiedad de los creadores del software. Y dependiendo de las condiciones de la licencia, existirán diversos "usos" permitidos a los "usuarios" (Zuniga, 2007).

4.1 Ventajas del software libre.

El software libre presenta una serie de ventajas sobre el software propietario por derechos que otorga a usuarios. Algunas de estas ventajas pueden ser más apreciadas por usuarios particulares, otras por empresas, y otras por administraciones publicas (Culebro, et al, 2006).

Principales ventajas.

Bajo costo, económico. 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.

El bajo o nulo costo de los productos libres permiten proporcionar a las PYMES servicios y ampliar sus infraestructuras sin que se vean mermados sus intentos de crecimiento por no poder hacer frente al pago de grandes cantidades en licencias.

La práctica totalidad de los concursos para desarrollo de software para la administración pública pasan por compatibilizar con productos de la factoría de Microsoft, por lo que garantiza la perpetuación e ingresos hacia Microsoft y no favorece a las empresas locales que pudieran ofrecer productos equivalentes.

Libertad de uso y redistribución. Las licencias de software libre existentes permiten la instalación del software tantas veces y en tantas máquinas como el usuario desee.

Independencia tecnológica. El acceso al código fuente permite el desarrollo de nuevos productos sin la necesidad de desarrollar todo el proceso partiendo de cero. El secretismo tecnológico es uno de los grandes frenos y desequilibrios existentes para el desarrollo en el modelo de propiedad intelectual.

Fomento de la libre competencia al basarse en servicios y no licencias. Uno de los modelos de negocio que genera el software libre es la contratación de servicios de atención al cliente. Este sistema permite que las compañías que den el servicio compitan en igualdad de condiciones al no poseer la propiedad del producto del cual dan el servicio. Esto, además, produce un cambio que redunda en una mayor atención al cliente y contratación de empleados, en contraposición a sistemas mayoritariamente sostenidos por la venta de licencias y desatención del cliente.

Soporte y compatibilidad a largo plazo. Este punto, más que una ventaja del software libre es una desventaja del software propietario, por lo que la elección de software libre evita este problema. Al vendedor, una vez ha alcanzado el máximo de ventas que puede realizar de un producto, no le interesa que sus clientes continúen con el. La opción es sacar un nuevo producto, producir software que emplee nuevas tecnologías solo para éste y no dar soporte para la resolución de fallos al anterior, tratando de hacerlo obsoleto por todos los medios, pese a que este pudiera cubrir perfectamente las necesidades de muchos de sus usuarios.

Formatos estándar. Los formatos estándar permiten una interoperabilidad más alta entre sistemas, evitando incompatibilidades. Los estándares de facto son válidos en ocasiones para lograr una alta interoperabilidad si se omite el hecho que estos exigen el pago de royalties a terceros y por razones de mercado expuestas en el anterior punto no interesa que se perpetúen mucho tiempo. Los formatos estándares afectan a todos los niveles. Un ejemplo lo estamos viendo en los documentos emitidos por las administraciones públicas en distintos formatos y versiones, que producen retrasos y dificultades en el acceso adecuado a la información para las mismas administraciones y para sus usuarios.

Sistemas sin puertas traseras y más seguros. El acceso al código fuente permite que tanto hackers como empresas de seguridad de todo el mundo puedan auditar los programas, por lo que la existencia de puertas traseras es ilógica ya que se pondría en evidencia y contraviene el interés de la comunidad que es la que lo genera.

Corrección más rápida y eficiente de fallos. El funcionamiento e interés conjunto de la comunidad ha demostrado solucionar más rápidamente los fallos de seguridad en el software libre, algo que desgraciadamente en el software propietario es más difícil y costoso. Cuando se notifica a las empresas propietarias del software, éstas niegan inicialmente la existencia de dichos fallos por cuestiones de imagen y cuando finalmente admiten la existencia de esos bugs tardan meses hasta proporcionar los parches de seguridad.

Métodos simples y unificados de gestión de software. Actualmente la mayoría de distribuciones de Linux incorporan alguno de los sistemas que unifican el método de instalación de programas, librerías, etc. por parte de los usuarios. Esto llega a simplificar hasta el grado de marcar o desmarcar una casilla para la gestión del software, y permiten el acceso a las miles de aplicaciones existentes de forma segura y gratuita a la par que evitan tener que recurrir a páginas web de dudosa ética desde las que los usuarios instalan sin saberlo spyware o virus informáticos en sus sistemas. Este sistema de acceso y gestión del software se hace prácticamente utópico si se extrapola al mercado propietario.

Sistema en expansión. Las ventajas especialmente económicas que aportan las soluciones libres a muchas empresas y las aportaciones de la comunidad han permitido un constante crecimiento del software libre, hasta superar en ocasiones como en el de los servidores web, al mercado propietario.

El software libre ya no es una promesa, es una realidad y se utiliza en sistemas de producción por algunas de las empresas tecnológicas más importantes como IBM, SUN Microsystems, Google, Hewlett-Packard, etc. Paradójicamente, incluso Microsoft, que posee sus propias herramientas, emplea GNU Linux en muchos de sus servidores.

4.2 Desventajas del software libre.

La curva de aprendizaje es mayor.

El software libre no tiene garantía proveniente del autor.

Se necesita dedicar recursos a la reparación de errores.

No existiría una compañía única que respaldará toda la tecnología.

Las interfaces amigables con el usuario (GUI) y la multimedia apenas se están estabilizando.

La diversidad de distribuciones, métodos de empaquetamiento, licencias de uso, herramientas con un mismo fin, etc., pueden crear confusión en cierto número de personas.

Algunas aplicaciones (bajo GNU/Linux) pueden llegar a ser algo complicadas de instalar.

Poca estabilidad y flexibilidad en el campo de multimedia y juegos.

4.3 Ventajas del Software propietario.

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 la usabilidad 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 usar.

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 estudios de la mayoría de las universidades del país 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. Existen 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 (Culebro, et al, 2006).

4.4 Desventajas del software propietario.

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.

Ilegal 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 fabricante, para que sea esta 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 fabricante. 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 dependencias a menos que cada una de ´estas contrate todas las licencias necesarias.

Quedar sin soporte técnico. Si la compañía fabricante del software propietario se va a la banca rota el soporte técnico desaparece, la posibilidad de en un futuro tener

versiones mejoradas de dicho software desaparece y la posibilidad de corregir los errores de dicho software también desaparece. 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 fabricante 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 (Culebro, et al, 2006).

Capítulo 5. El software libre y la administración pública.

La discusión en torno al software libre tiene uno de los puntos medulares en la administración pública. Si, el software libre debe ser tenido en cuenta, esto no solo es debido a los beneficios que recaen sobre los usuarios individuales. La administración pública también se verá beneficiada adoptando la plataforma Gnu/Linux y aquellos programas de código abierto que sean necesarios por parte de los sistemas informáticos de todas las dependencias (Gutiérrez, 2005). Hoy en día, cualquier empresa o institución utiliza numerosos programas informáticos para realizar la mayor parte de sus actividades; los datos de los ciudadanos están almacenados electrónicamente en diversos organismos públicos y privados que hacen uso de ellos según las diferentes leyes sobre privacidad y, básicamente, por los principios éticos que guían su actividad. La elección de un formato concreto para almacenar y gestionar estos datos supone de por sí una decisión que afecta a todas las personas cuyos datos se almacenan (Jiménez Romera, 2002).

En el caso de las administraciones públicas, los ciudadanos se ven obligados, bien por patriotismo o bien por temor a la ley, a ceder sus datos, por lo que debería ser lógico que la propia administración garantizase de forma adecuada la gestión de tan valiosa información. Este debate, que debería ser público y abierto, se encuentra, una vez más, alejado de la ciudadanía, en un ámbito en que las grandes empresas pueden ejercer sus presiones legales, o incluso ilegales, sin preocuparse por un público apático e indiferente a estos temas, en los que se `enorgullece' de su ignorancia (Jiménez Romera, 2002).

Existen argumentos para que un país adopte el software libre en la administración pública y se mencionan a continuación:

1) *Costo.* El gasto en sistemas operativos por parte del Estado es un gasto que sencillamente puede evitarse pasándose a GNU/Linux, que es un sistema operativo que es gratis. Lo mismo vale decir para las distintas aplicaciones finales, como los paquetes de oficina, bases de datos, navegadores, clientes de correo electrónico, entre otras.

El software abierto normalmente hace un uso más eficiente de hardware antiguo que el software propietario, cosa que permite recortar costes en hardware y en ocasiones evitar la adquisición de hardware nuevo.

2) *Seguridad*. Contrariamente a lo que puede pensarse, el hecho de hacer públicos los códigos de los programas no va en contra de la seguridad de los mismos sino que la favorece. Utilizando software libre se puede saber qué está haciendo realmente un programa, qué tipo de información maneja y cómo lo hace. Una buena seguridad debe basarse en la transparencia. El software propietario oculta estos aspectos y muchas veces no sabemos qué información está enviando a otras computadoras remotas. La transferencia de información reservada puede ser debida a fallas o errores contenidos en los programas o porque así lo hicieron intencionalmente sus fabricantes.

3) *Autonomía tecnológica*. Adoptando el software libre y con las posibilidades que éste ofrece de acceder al código fuente, muchos usuarios pasarán de ser consumidores a ser desarrolladores de software. Esto significa que se podrán adaptar los programas a las necesidades específicas de las distintas dependencias, y todas esas modificaciones deberán realizarse siguiendo los requisitos exigidos por el modelo de software libre. La autonomía tecnológica debe estar vinculada al concepto de estándares abiertos, que consisten en especificaciones técnicas que son publicadas por una organización y puestas a disposición de cualquier usuario para ser implementadas en aplicaciones específicas, lo cual favorece la interoperabilidad entre las distintas aplicaciones.

4) *Independencia de proveedores*. Adquiriendo un software propietario generamos una relación de dependencia con respecto a un fabricante. Una vez que instalamos dicho software dependeremos del fabricante para obtener actualizaciones, y en la mayoría de los casos esas actualizaciones exigirán invertir más dinero aparte del que ya pagamos. Con una política de software libre, si el Estado paga por el desarrollo de un software exigirá que se le entregue el código fuente del mismo, con lo cual si en el futuro desea efectuarle modificaciones podrá optar por proporcionarle el código a otros desarrolladores para que las realicen.

5) *Argumento democrático*. Las nuevas tecnologías de la información han pasado a ocupar un lugar central en la gran mayoría de las sociedades. Si bien cada vez son más los usuarios que acceden a dichas tecnologías, la “brecha tecnológica” es muy grande y

en medio del actual modelo instaurado es un factor más de exclusión social. El software libre favorece la democratización de la información permitiendo la utilización de protocolos y lenguajes que no son propiedad ni monopolio de nadie. En este mismo argumento se sitúan la posibilidad de traducir el software a lenguas para las que no esté disponible en su origen, así como adaptarlo a las características propias de quienes serán los usuarios finales, antes que el usuario se adapte a las características que le impone el software.

6) *Aprovechamiento más adecuado de los recursos.* Muchas aplicaciones utilizadas o promovidas por las administraciones públicas son también utilizadas por muchos otros sectores de la sociedad. Por ello, cualquier inversión pública en el desarrollo de un producto libre que le interesa redundará en beneficios no sólo en la propia administración, sino en todos los ciudadanos que podrán usar ese producto para sus tareas informáticas, probablemente con las mejoras aportadas por la administración.

7) *Fomento de la industria local.* Una de las mayores ventajas del software libre es la posibilidad de desarrollar industria local de software. Cuando se usa software propietario, todo lo gastado en licencias va directamente al fabricante del producto, y además esa compra redunda en el fortalecimiento de su posición. Lo cual no es necesariamente perjudicial, pero poco eficiente para la región vinculada a la administración si analizamos la alternativa de usar un programa libre.

En este caso, las empresas locales podrán competir proporcionando servicios (y el propio programa) a la administración, en condiciones muy similares a cualquier otra empresa. Digamos que, de alguna manera, la administración está allanando el campo de juego, y haciendo más fácil que cualquiera pueda competir en él. Y naturalmente, entre esos cualquiera estará las empresas locales, que tendrán la posibilidad de aprovechar sus ventajas competitivas (mejor conocimiento de las necesidades del cliente, cercanía geográfica, etc.).

8) *Disponibilidad a largo plazo.* Muchos datos que manejan las administraciones y los programas que sirven para calcularlos han de estar disponibles dentro de decenas de años. Es muy difícil asegurar que un programa propietario cualquiera estará disponible cuando hayan pasado esos periodos de tiempo, y más si lo que se desea es que funcione en la plataforma habitual en ese momento futuro. Por el contrario, es muy

posible que su productor haya perdido interés en el producto y no lo haya portado a nuevas plataformas, o que sólo esté dispuesto a hacerlo ante grandes contraprestaciones económicas. De nuevo, hay que recordar que sólo él puede hacer ese porte, y por lo tanto será difícil negociar con él. En el caso del software libre, por el contrario, la aplicación está disponible con seguridad para que cualquiera la porte y la deje funcionando según las necesidades de la administración. Si eso no sucede de forma espontánea, la administración siempre puede dirigirse a varias empresas buscando la mejor oferta para hacer el trabajo. De esta forma puede garantizarse que la aplicación y los datos que maneja estarán disponibles cuando haga falta (Gutiérrez, 2005).

La elección del software afecta y se ve afectada por las políticas gubernamentales en multitud de formas. Las acciones y decisiones de los gobiernos tienen un tremendo impacto en la forma en que la sociedad hace uso de las TIC. Los gobiernos establecen los límites económicos que permiten desarrollar las actividades económicas, y suelen ser los mayores clientes de productos y servicios en TIC. Los marcos legales y reguladores que implementan las políticas gubernamentales establecen los parámetros para el uso de las TIC y pueden fomentar o entorpecer la extensión del uso de las TIC.

En el sector público, Europa tiene una mayor penetración. Alemania, Francia y España lideran en Europa la adopción de SFA. El apoyo gubernamental a la adopción del SFA ha sido la clave para ello, si bien con diferentes instrumentos de implantación. El Gobierno Alemán ha lanzado políticas de promoción y apoyo del SFA, el Gobierno francés ha promovido de manera centralizada la implantación del SFA en la Administración y en las empresas públicas, mientras que en España la adopción de políticas de promoción del software de fuentes abiertas ha recaído mayoritariamente en iniciativas de las Comunidades Autónomas, si bien bajo un claro marco de políticas dictadas por el Ministerio de Industria, Turismo y Comunicaciones y por el Ministerio de la Presidencia.
Además hay que tener en cuenta el papel impulsor y armonizador que juegan las instituciones europeas y que están contribuyendo a la promoción del sector

de las Tecnologías de la Información y las Comunicaciones (TIC), y dentro de este ámbito, a la promoción del SFA, como uno de los motores claves del sector europeo. Así, al amparo del Séptimo Programa Marco para la Investigación y el Desarrollo Tecnológico, la Unión Europea financia proyectos tecnológicos que suponen desarrollos de SFA para los que involucran a varias empresas y universidades de diferentes países que aportan diferentes competencias para su ejecución (Centro Nacional de Referencia de Aplicación de las Tecnologías de Información y la Comunicación basadas en Fuentes Abiertas, 2010).

En la región del Pacífico, Australia destaca como uno de los países con mayor adopción del software de fuentes abiertas en el mundo, gracias a que cuenta con activas comunidades de desarrolladores de SFA que participan en proyectos internacionales. Además, la Universidad tiene un papel muy importante tanto en la formación de personal cualificado en TIC como e la participación en proyectos de SFA (Centro Nacional de Referencia de Aplicación de las Tecnologías de Información y la Comunicación basadas en Fuentes Abiertas, 2010).

En Asia encontramos cuatro países liderando el mundo del software de fuentes abiertas: India, China, Japón y Corea del Sur; aunque con niveles de avance en la Sociedad de la Información muy heterogéneos. India es el país más atípico, ya que ha conseguido un importante nivel de desarrollo de SFA gracias al nivel educativo de su población y su dedicación a la programación para compañías americanas y europeas, a pesar de un bajo nivel de la SI en el país. El desarrollo del SFA en China está absolutamente dirigido por el Gobierno, hasta el punto de que la principal empresa proveedora de SFA, Red Flag Linux, está participada por el Estado. En Japón y Corea del Sur, el sector de la electrónica ha sido uno de los motores para el desarrollo de aplicaciones de SFA, pero no el único. El Gobierno coreano ha promovido especialmente el SFA como un medio de impulsar y dinamizar el Sector de las TIC del país. La existencia de una empresa local líder en la distribución del software de fuentes abiertas ha favorecido la adopción del SFA por las empresas del sector privado, contribuyendo al crecimiento y madurez del sector TIC (Centro Nacional de

Referencia de Aplicación de las Tecnologías de Información y la Comunicación basadas en Fuentes Abiertas, 2010).

En Latinoamérica, Brasil se destaca del resto de países de la zona por su mayor adopción y desarrollo del SFA, comparable a países como India y China. Y ello a pesar de que su nivel de SI es similar a los mayores países latinoamericanos como Argentina, México y Chile. La razón está en el impulso que el Gobierno ha sabido conferir a todos los ámbitos del ecosistema del SFA: publicación de normativas, migraciones masivas en agencias y empresas del sector público, desarrollo de productos (bienes y servicios) de SFA desde la Universidad pública y creación de un portal colaborativo para los actores de la Comunidad. Una mayor concienciación de los usuarios acerca del uso de software 100% legal permitirá en el futuro mayores tasas de uso de software de fuentes abiertas. La falta de políticas de apoyo por parte de Gobiernos y distribuidoras que puedan proporcionar el soporte necesario, es la clave que explica el menor desarrollo del SFA en Latinoamérica (Centro Nacional de Referencia de Aplicación de las Tecnologías de Información y la Comunicación basadas en Fuentes Abiertas, 2010).

Finalmente, África se posiciona a la cola mundial en desarrollo de Software de Fuentes Abiertas y de Sociedad de la Información, careciendo de los mínimos medios para que efectivamente se pueda desarrollar el SFA. Si a ello sumamos la inexistencia de políticas públicas de promoción y la alta tasa de uso de software ilegítimo (Centro Nacional de Referencia de Aplicación de las Tecnologías de Información y la Comunicación basadas en Fuentes Abiertas, 2010).

Por tanto, que los países con mayor nivel de desarrollo y adopción del SFA como Estados Unidos, Australia, Alemania, Francia, España o Brasil, comparten altos niveles de desarrollo en todos los elementos del ecosistema: El Gobierno, las Universidades, las Empresas y la Comunidad de Desarrolladores (Centro Nacional de Referencia de Aplicación de las Tecnologías de Información y la Comunicación basadas en Fuentes Abiertas, 2010).

5.1. Software libre y las universidades.

La relación software libre y universidades tienen su máximo exponente en la historia del software y las redes de computadoras. Por ejemplo, la tecnología en que se basa el internet y el comercio electrónico (TCP/IP, Apache, Senmail) así como también el núcleo Linux, proviene de la investigación de las universidades. Estas deberían reconocer la importancia del software libre y liderarlo. Deberían evitar dejarse persuadir por las aparentes ventajas del software comercial.

El Software Libre en los países ricos, igual que sucedió con Internet, han tenido mucho que ver con las universidades, tanto con el personal docente e investigador como con los estudiantes. A grandes rasgos, el proceso de introducción de los programas de Software Libre suele ser el siguiente (Hypatia, 2003):

1. Algunos gurús (investigadores, profesores de universidad e incluso alumnos avanzados) tienen conocimiento de programas que aún no han terminado de madurar pero que prometen ser interesantes. Estas personas se incorporan a la comunidad que los desarrolla, experimenta y evalúa.
2. Otras personas, siempre en el entorno universitario, se animan a probar esos programas de que los gurús hablan. Obtienen una experiencia positiva y el círculo se amplía.
3. Las universidades empiezan a usar los Programas Libres como herramienta privilegiada de buen nivel tecnológico y alto valor pedagógico ya que se trata de productos accesibles, abiertos y disponibles sin necesidad de seguir complicados trámites burocráticos ni de solicitar gastos costosos.
4. Los alumnos de estudios superiores tecnológicos, movidos por la necesidad, pero también por la curiosidad, buscan la manera de probar Programas Libres. Disponen en su mayoría de computadora e incluso de la posibilidad de conectar a Internet su computadora. Pueden tomarse la libertad de instalar y probar aplicaciones a voluntad.
5. Se producen unas primeras promociones de técnicos que han usado Programas Libres en su formación universitaria. Al llegar a las empresas encuentran problemas

cuya solución es más fácil, o más económica, o de mayor calidad, o todo ello a la vez, en comparación a los productos comerciales.

6. La industria empieza a tener conocimiento de los Programas Libres y sus ventajas. Los nuevos titulados valoran el conocimiento de esos programas indicándolos en sus CV.

7. De manera paralela, muchos productos alcanzan la madurez. La industria cuenta con productos de calidad y con una masa crítica de técnicos suficiente con conocimientos para mantenerlos. Se empieza a extender el uso de determinados productos abanderados de los Programas Libres.

8. Los Programas Libres adquieren prestigio y la industria empieza a cooperar en su soporte, soporte de sus productos hardware e incluso versiones enteras de productos.

9. Productos comerciales optan por convertirse en programas de fuente abierta para incorporar a la comunidad en el proyecto de desarrollo.

10. La madurez se generaliza cuando la administración pública, la educación pública, el sistema sanitario, etc. empiezan a optar por el Software Libre (Hypatia, 2003).

El software libre ha crecido en los últimos años, las actuales tendencias macroeconómicas están reforzando a GNU/Linux y al software libre. Las administraciones públicas están siendo la mejor ayuda para promocionar este tipo de tecnologías, pero su uso dentro de las instituciones se reduce prácticamente en los centros educativos, en lo referente a universidades algunas están empezando a aplicarlo en su gestión además de poner en marcha proyectos de difusión entre los alumnos. Las universidades están siendo el verdadero centro neurológico del desarrollo de actividades en temas de software libre. Además la gran presencia de grupos de usuarios, son los propios maestros de la universidad los que ya han tomado la iniciativa.

Capítulo 6. Casos de implementación y éxito del uso del software libre.

Cuando se navega por Internet, se utiliza Linux. Cuando se busca en Google, se habla en Facebook o se juega con un teléfono Android (sistema operativo basado en Linux) —850.000 se activan a diario—, retirar dinero de un cajero, también se usa este sistema operativo. Linux está en el corazón de múltiples actividades cotidianas, aunque no sea consciente.

El sistema operativo abierto más implantado en el mundo y motor del movimiento del software libre no está, sin embargo, masivamente instalado en las computadoras de escritorio donde reina Windows (Microsoft) con un 92% de cuota de mercado. Al igual que en los noventa, cuando Linus Torvalds (Helsinki, 1969) desarrolló Linux. El viernes 20 de abril del 2012, la Academia de Tecnología de Finlandia reconocía a su compatriota por crear un sistema de "gran impacto en el desarrollo de los programas abiertos, el trabajo en red y la apertura de la Web para hacerla accesible a millones de personas" (EL País, 2012).

Linus Torvalds tenía 21 años y estudiaba ingeniería informática en la Universidad de Helsinki en 1991. En su habitación empezó "un pequeño proyecto, un divertimento para aprender. Pero resulta que hacía todo lo que un sistema operativo debe hacer".

El joven subió la primera versión de Linux a la Red y el boca a boca digital hizo el resto con un sistema protegido bajo la Licencia Pública General (GPL), que permite su uso, copia, modificación y libre distribución. A diferencia de otros, Linux mejora de forma colaborativa. Cerca de 8.000 desarrolladores y 800 compañías han contribuido en sus 15 millones de líneas de código desde 2005. La Ilíada, de Homero, tiene 15.000. Y cada tres meses sale una nueva versión del núcleo bajo la supervisión de Torvalds.

"Con Linux se dieron por primera vez las condiciones para que cualquiera instalase y utilizase un sistema operativo modificable", explica Miguel Jaque, director del Centro Nacional de las Tecnologías basadas en fuentes abiertas (Cenatic). "Podías saber

cómo funcionaban sus líneas de código. Se acabó el secreto. Y empezó el auge del software libre".

Veinte años después no ha conseguido colarse en las computadoras domésticas (0,98% de cuota mundial según Netmarketshare), pero reina en móviles, empresas, centros de datos, entornos críticos y en la infraestructura de la Red. El 80% de las transacciones bursátiles se realizan en sus plataformas y 9 de cada 10 superordenadores llevan el símbolo del pingüino en su interior. Hasta televisiones y coches. El 25% de su costo ya es software y en cuatro años será el 75%. De ahí que gigantes como General Motors Company (fabricante estadounidense de vehículos), BMW (fabricante alemán de automóviles y motocicletas), Hyundai (conglomerado de empresas y negocios de Corea del Sur), PSA Peugeot Citroën y Renault-Nissan (fabricante de automóviles) construyan una plataforma abierta para los sistemas de entretenimiento e información (consorcio Genivi).

En España, Administración y comunidad educativa han liderado su avance. El 83% de organismos públicos tiene instalado algún tipo de software abierto, sin olvidar que Extremadura se puso a la cabeza en ordenadores por alumno gracias a Linex en el lejano 2003. Lo sigue usando, aunque no lo predique tanto. Una marea que se trasladó a Andalucía y Cataluña (Lincat) entre otras siete autonomías. El Guadalinex andaluz atiende hoy a 1.800.000 alumnos de 5.882 colegios con una red de 640.000 PC y 4.200 servidores (EL País, 2012).

El Ayuntamiento de Múnich ha ahorrado un tercio de su presupuesto tecnológico (4 millones de euros) y ahora, en época de crisis, se podría ahorrar más si las Administraciones "compartieran y reutilizaran los recursos informáticos". En 2011, según el Cenatic, el 46% de ellas crearon sus programas, pero solo el 18% los liberaron (EL País, 2012).

Sin duda alguna las Universidades en México son piezas fundamentales en el impulso del uso de software, la mayoría incluye en sus programas el uso de Sistemas Propietarios, e incluso sus laboratorios están equipados en su mayoría con todo tipo de

software de este tipo. La Universidad Politécnica de Pachuca, apuesta por el uso de Software Libre en todos sus equipos, buscando disminuir el uso de software licenciado a lo realmente necesario. De esta forma, el 95% de los servidores trabajan con el sistema operativo CentOS, (el otro 5% utiliza Debian). Se esta haciendo un análisis exhaustivo para lograr que el personal administrativo realice sus tareas con Software Libre en vez de Software Propietario, con lo que se pretende impulsar el uso de esta tecnología. De esta forma, la Universidad hoy en día utiliza distintas versiones de Software Libre adecuado a sus necesidades, desde el sistema de telefonía basado en Asterisk, hasta el sistema de acceso programado en la Universidad en PHP y MySQL (Estepa Nieto, 2007).

Michoacán ahorró 2 millones 166 mil dólares al no pagar por las licencias de los programas de cómputo que utiliza el gobierno en el año 2006. Y se siguen utilizando en gran cantidad software libre a la fecha. Entre los principales beneficios de estos programas "libres" se hallan los ahorros, pero es importante también la autonomía que proporciona implementarlos y la independencia hacia marcas establecidas que cobran por licencias y asesoría. Además, permite desarrollar capital humano altamente calificado. Existen dos condiciones que deberían ser indeclinables para los gobiernos actuales: el uso de estándares abiertos, y la utilización de software libre, pues otorga independencia de los proveedores y da mayor control de la información pública (Sandoval, 2006).

El Instituto Nacional de Antropología e Historia desde ya varios años a implementado Software Libre en sus sistemas como lo son en redes, telefonía, monitoreo y operación de infraestructura además desarrolla sus sistemas en base a software libre.

Capítulo 7. Aspectos generales de un estudio de factibilidad.

Estudio de Factibilidad

Según Kendall y Kendall (1997), es un análisis cuyo fin es determinar que un proceso, diseño, procedimiento o plan, puedan ser cumplidos exitosamente en un periodo de tiempo establecido. La factibilidad del proyecto son las decisiones basadas en los datos recolectados por el investigador, los cuales deben ser representados por el analista. Los objetivos de control exigen que para todo sistema que se diseñe, debe realizarse un estudio de factibilidad, previa definición de requerimientos para cada una de las opciones revisadas, sugeridas y evaluadas, que comprenden la Factibilidad Técnica, Operativa y Económica.

El éxito de un proyecto esta determinado por el grado de factibilidad que se presente en cada uno de los tres aspectos anteriores, el estudio de factibilidad sirve para recolectar datos relevantes sobres el desarrollo de un proyecto y en base a esto tomar la mejor decisión, si procede su estudio, desarrollo o implementación. El estudio de la factibilidad se hace para encontrar cuáles son los objetivos organizacionales, y luego determinar si el proyecto sirve para mover el negocio hacia sus objetivos en alguna forma.

7.1. Factibilidad Técnica

El analista debe encontrar si los recursos técnicos actuales pueden ser mejorados o añadidos, en forma tal que satisfaga la petición bajo consideración. Si los sistemas existentes no pueden ser añadidos, la pregunta es si hay tecnología en existencia para satisfacer las especificaciones. Si la respuesta sobre si una tecnología particular se encuentra disponible y es capaz de satisfacer las peticiones del usuario es “si”, entonces la pregunta se convierte ahora en económica.

7.2. Factibilidad Económica

Esta es la segunda parte de determinación de recursos. Los recursos básicos a considerar son: el tiempo propio y el del equipo de sistemas, el costo de hacer un estudio de sistema completo (incluyendo el tiempo de los empleados con los quien se

trabajará), el costo del tiempo de los empleados del negocio, el costo estimado de Hardware y el costo estimado de software y/o desarrollo de software. Si los costos a corto plazo no son sobrepasados por las ganancias a largo plazo, o no producen una reducción inmediata en los costos de operación, el sistema no es factible económicamente y el proyecto ya no debe continuar.

7.3. Factibilidad operacional

Supongamos por un momento que se juzguen adecuados los recursos técnicos y económicos. La factibilidad operacional depende de los recursos humanos disponibles para el proyecto, e involucra proyectar si el sistema operará y será usado una vez que esté instalado. Si los usuarios están casados virtualmente con el sistema presente, no ven problemas en él y, por lo general, no están involucrados en la petición de un nuevo sistema, la resistencia ante la implantación del nuevo sistema será fuerte. Las oportunidades de que alguna ves llegue a ser operacional son escasas. En forma alterna si los usuarios han expresados la necesidad de que un sistema de que es operacional la mayor parte del tiempo tenga una forma mas eficiente y accesible, se tiene mejor oportunidad de que el sistema solicitado llegue a ser utilizado. La determinación de la factibilidad operacional requiere imaginación creativa por parte del analista de sistemas, así como de su poder de persuasión, que permita que los usuarios sepan cuáles interfaces son posibles y cuáles satisfacen sus necesidades. El analista también debe escuchar cuidadosamente lo que en realidad quieren los usuarios y lo que parece que usaran.

7.4. Evaluación de la factibilidad

La evaluación de la factibilidad de los proyectos de sistemas nunca es una tarea fácil o bien definida. Lo que es más, la factibilidad del proyecto no es una decisión a ser tomada por el analista de sistemas sino por la administración. Las decisiones están basadas en los datos de factibilidad recolectados en forma experta y profesional presentados por el analista. Es necesario asegurar que las tres áreas de factibilidad, técnica, económica y operacional, sean atacados en el estudio. El estudio de un proyecto debe de ser logrado rápidamente, a fin de que los recursos que se le dediquen

sean mínimos. El estudio de factibilidad es una herramienta útil para usar eficientemente los recursos escasos, como el dinero, el tiempo, etcétera. Con estos se cuenta para la puesta en operación de un proyecto; porque contempla todos los elementos para evaluar y determinar si el proyecto es bueno o malo, y en cuáles condiciones se debe desarrollar para su éxito (E. Kendall & E. Kendall, 2005).

Capítulo 8. Caso Práctico

8.1. Factibilidad Del Uso Del Software Libre En La Administración De La Facultad De Contaduría Y Ciencias Administrativas De La Universidad Michoacana De San Nicolás De Hidalgo

Con el objetivo de conocer si dentro de la facultad se utilizan algún tipo de software libre, y si los empleados saben lo que es, se aplican entrevistas cerradas para levantar información. Los entrevistados se determinan conforme al conteo de los puestos de trabajo que son 27 y si los entrevistados se encuentran, se le aplica la entrevista de lo contrario no se aplica.

Muestreo Aleatorio simple.
Se caracteriza por que otorga la misma probabilidad de ser elegidos a todos los elementos de la población.
Características técnicas.
Población: 27
Tipos de entrevista: Estandarizada
Ámbito: Local Facultada de Contaduría y Ciencias Administrativas
Población de estudio: Administrativos de la facultada de Contaduría y Ciencias Administrativas
Tamaño de la muestra: 19

Grado de Estudios	No. Trabajadores
Educación Media Superior	3
Educación Superior	12
Educación Posgrado	4
Total	19

Tabla 4. Personal Administrativo.

Para la recolección de información se aplica el siguiente cuestionario a los entrevistados y las preguntas son las siguientes:

Preguntas echas.

1. ¿Sabe qué es el software libre?
2. ¿Usas software libre?
3. ¿Qué tipo de software prefieres?
4. ¿El software libre te da confianza?
5. ¿Conoces algún sistema operativo que sea Software libre?
6. ¿Desde tu punto de vista será factible utilizar software libre para las funciones administrativas?

Conocimiento del tema.

¿Sabes que es el software libre?

Gráfica 1 Conocimiento del tema.

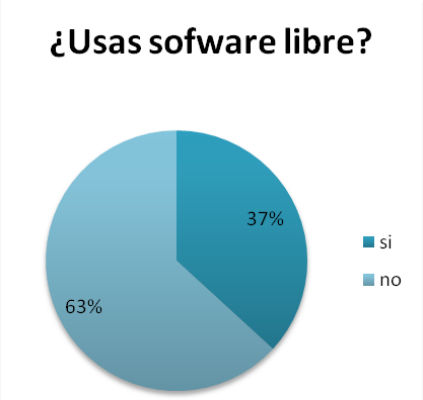
El 32% de los entrevistados tiene algún conocimiento sobre el término “software libre”. Es un porcentaje bajo, sin embargo solo una pequeña parte de ellos conoce bien la filosofía que hay de tras de estos programas. La asociación Software libre es igual al software gratis es muy común. Esta apreciación de hace que se tenga una visión distorsionada de la filosofía que promueve este software pues los usuarios se fijan mas en el aspecto económico y no en la libertad de los programas.

El restante de los entrevistados 68% no conoce ni saben que es el software libre ellos nada mas usan los programas como herramientas para trabajo sin conocer si existe o no el software libre.

Uso del Software libre.

¿Usas software libre?

En esta gráfica podemos ver que un 63% de los encuestados no usan software libre porque no lo conocen, mientras que el 37% restante saben un poco de lo que es el



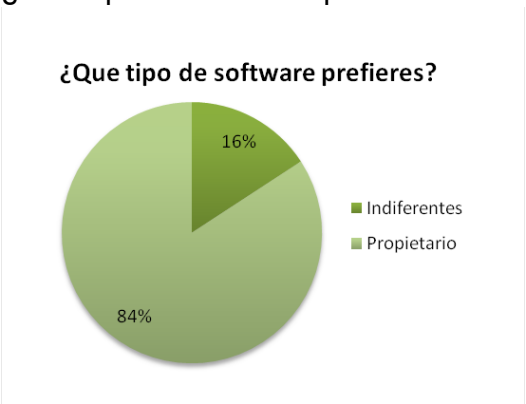
Gráfica 2 Uso del Software libre.

Software libre pero no como tal, por lo tanto utilizan software gratis como programas que bajan del Internet sin ningún costo por ello, y para ellos es libre.

Esto quiere decir que la mayoría de los empleados no utiliza software libre.

Preferencia de software.

¿Qué tipo de software prefieres?

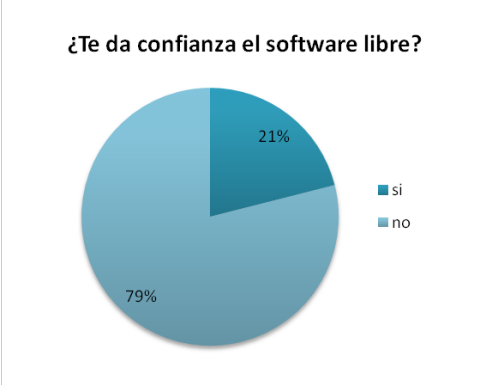


Gráfica 3 Tipo de Software.

La mayoría que conoce sobre software libre, en sus actividades diarias no lo usan de manera permanente simplemente lo conocen y en ocasiones lo utilizan para fines de aprendizaje o por el simple hecho de que ya están acostumbrados a utilizar el propietario en su vida diaria, y se le es más cómodo seguirlo utilizando a cambiarse de programas. Usan el propietario aunque digan que el software libre es bueno. Y en cuanto a los que usan software propietario y no conocen el software libre me da a entender que si se les cambia en lo mas mínimo un programa del que tenían anteriormente causa un descontrol y por consecuencia pérdida de tiempo, sin saber que las funcionalidades del programa cumplen con las mismas tareas que los usados por ellos diariamente.

Grado de confianza

¿El Software libre te da confianza?



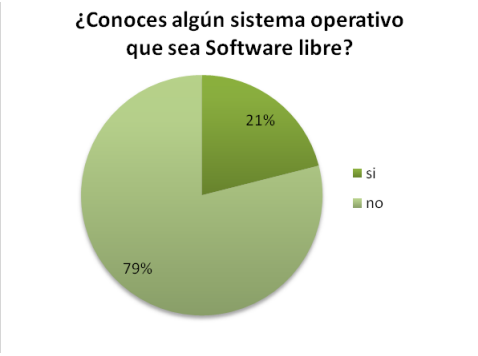
Gráfica 4 Confianza del Software libre.

Se ve que solamente 21% de los encuestados que son 4 le tiene confianza porque saben que en el desarrollo de este software son muchos los que intervienen para lanzar una versión estable y por ende le tienen confianza de usarlo.

El restante con un 79% no le tienen confianza porque temen que cargue con código maligno o con virus, al ser gratis un programa como ellos lo definen no le dan la confianza de usarlo.

Conocimiento de Sistemas Operativos.

¿Conoces algún sistema operativo que sea Software libre?



Gráfica 5 Conocimiento del Sistema Operativo.

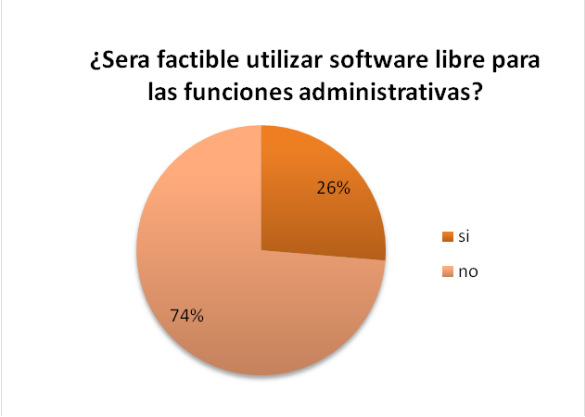
De los encuestados solamente 4 conocen que si hay sistemas basados en Software libre como lo es Linux y se menciono Solaris.

De los demás encuestados no saben que hay otros sistemas operativos y se les menciono el termino Linux y no saben que es.

Factibilidad del software libre en funciones administrativas.

¿Desde tu punto de vista será factible utilizar software libre para las funciones administrativas?

El 74% de los encuestados decide que no es factible la implementación de Software libre y el restante que son 26% deciden que sería factible la implementación de software libre en funciones administrativas.



Gráfica 6 Factibilidad del Software.

Una vez terminada las preguntas decido explicarles lo que es el software libre, sus ventajas y la mayoría dice que si sería factible utilizar software libre puesto que se acomodaría a sus necesidades de trabajo y que si estaría bien con esto de gasto de licencias, ahorrando dinero por el pago de la misma. Entre los comentarios de los que saben que es el software libre hubo quienes decían que solo el software libre estaría bien implementarlo por ahora en el lado del servidor no del cliente por que no están preparados para un cambio puesto que son personas que están por arriba de los 30 años y están impuestas a utilizar un programa en especifico.

Conclusiones de entrevistas.

La mayora que conocen o han utilizado software libre, coinciden que es seguro y fiable y que estaría bien dar a conocer sobre herramientas similares a las que ellos utilizan tal es el caso de el diseñador grafico que utiliza solamente programas propietario las cuales sabe usar, y en el caso de los demás empleados utilizan lo que es el Office de Microsoft. La gran mayoría de los encuestados asocian el software libre como software gratuito y esa no es la filosofía del software libre.

La difusión de lo que es el software libre entre los administrativos seria de gran ayuda para su consideración en alguna toma de decisiones como en la reducción de costos en cuanto a licencias, reutilización de los equipos de computo, seguridad en las redes, menos vulnerables, prácticamente cero virus, beneficios más allá de lo económico daría valor agregado a la institución.

Bastaría con explicar de forma más sencilla y amena posible.

- Las libertades y características de este software.
- Por qué es gratuito.
- Programas más populares.

Explicar cómo se desarrolla para generar confianza, hay que incidir no sólo en dar a conocer la parte teórica del software libre (su filosofía) sino cómo se aplica ésta en la práctica:

- Los desarrolladores se agrupan en comunidades con especialistas en diferentes áreas.
- La colaboración es altruista, insertar código dañino sería perjudicial para su imagen y sería trabajo en vano.
- Hay sistemas de control para las revisiones que realiza cualquier desarrollador.
- Los cambios son estudiados y testeados antes de aplicarlos en el programa

Obviamente cada caso es diferente, pero explicar hay detrás ayudará a bajar los niveles de desconfianza.

Situación en la facultad con software libre.

Para saber si en la facultad se está trabajando con este software y qué se está haciendo en base al desarrollo de aplicaciones en apoyo al área administrativa, se investigo lo siguiente. La facultad de Contaduría y ciencias administrativas, cuenta con una red de 27 computadoras que se utilizan en su mayoría para tareas administrativas como correo, mensajero, aplicaciones de oficina, multimedia, navegación web y que en la actualidad todas están funcionando con Windows XP Profesional con licencia y office 2010 también con licencia. En ninguna de estas computadoras esta instalado algún sistema de licencia libre. Cuenta con una red para mandar a imprimir documentos.

Cabe mencionar que solo laboratorios de la facultad cuentan con arranque dual que se puede escoger entre un sistema libre y uno privado.

Dinero destinado para el desarrollo de plataformas administrativas. Como tal no existe dinero para el desarrollo e investigación para plataformas administrativas. Es solo sueldo base de un programador y un Diseñador. Han desarrollado aplicaciones para funcionamiento de diferentes actividades, que en conjunto son 5 sistemas que están actualmente en desarrollo o en funcionamiento.

Los cuales son los siguientes.

- Control de asistencia.
- Control incidente informático.
- Control de inventario.
- Control bancario.
- Control contable.
- Sistema Kardex.

Sistema de Kardex en apoyo a la división de estudios de posgrado. Este sistema es en apoyo para dar de altas, bajas, consultas y modificaciones llevando control de los maestros, calificaciones de los alumnos, maestrías en el plantel y está dividida para su funcionamiento en administradores y usuarios. Los administradores de cada cuenta son los maestros que imparten clases en maestrías y doctorados y los usuarios que son los alumnos (como un siia).

Todo esta desarrollado bajo software libre que es Apache, Php y Mysql en el lado del servidor corriendo sobre un sistema Windows server 2008. Y del lado del cliente cualquier navegador web. Se comenta que se está tratando de migrar lo que es el servidor a un servidor con una distribución GNU/Linux pero está en proceso.

El software al implementarlo se tendrían ventajas y desventajas y son las siguientes.

Ventajas.

1. Los costos se reducen considerablemente (no se pagan licencias). No es software pirata.
2. El rendimiento de los equipos aumenta ya que los requisitos de hardware, en general, son menores.
3. La seguridad de la red y de los hosts se incrementa de manera considerable, prácticamente libre de virus, los datos viajan con mayor seguridad.
4. La inversión realizada podría quedar en manos de personas del centro de computo de la facultad como lo es hasta ahora un ingeniero en sistemas con su sueldo base y en apoyo a lo mejor estudiantes del mismo plantel (practicantes, servicio social).

5. A su vez, el mantenimiento y servicio técnico quedaría a cargo de estas mismas personas con lo cual generarían experiencia y alta especialización, agregándole valor a este departamento.
6. Se podría adaptar el software de manera más específica y puntual a las necesidades de usuarios, o a la necesidad del plantel, alto grado de personalización, y esto lleva una mayor adaptabilidad por parte del personal.
7. Los sistemas diseñados en la actualidad son más estables en ambientes de escritorio que años anteriores, y estos vienen con todas las aplicaciones necesarias para empezar a trabajar como es ofimática, mensajeros, cliente de correo, navegadores.
8. Generación de scripts para el mantenimiento de los equipos a una hora determinada, y que es este rubro de los scripts se tiene infinidad de usos para beneficio.

Desventajas.

1. Debe considerarse una curva de aprendizaje, para que los empleados se adapten al nuevo sistema.
2. Puede haber dificultad en encontrar personas que nos brinden este servicio.
3. Se toma tiempo en la capacitación la cual en ocasiones no se tiene por diferentes funciones de los empleados que a su vez son maestros del plantel.
4. La edad de los empleados es arriba de los 30 años esto afecta en el aprendizaje que se pudiera tener, tomando más tiempo del considerado.

Son mas ventajas que desventajas pero si son aspectos críticos para una toma de decisión al final del día y conlleva tiempo en dado caso de implementar un software que no se esta acostumbrado a utilizar.

El software libre y la universidad Michoacana.

Dentro de la universidad hay dos facultades que usan software libre en cantidad pero dentro de los laboratorios y estos son Fisicomatemáticas que tienen una aula de equipos que instalaron en ellos la distribución Ubuntu de Linux, pero no en lo que es la administración de la facultad que utilizan software propietario como Windows y office de

Microsoft. Otra de las facultades de la universidad es la de Eléctrica en donde el ingeniero Samuel Pérez Aguilar comenta que en la facultad hace como 9 o 10 años se decidió cambiar lo que era los sistemas operativos en los cuales se tenía de todo como win98 win2000 y en algunas winXP y se cambiaron a Linux Mandriva y Red Had, excepto algunas que se compraron y que venían ya con su sistema instalado como lo eran las de marca y después se cambio a Ubuntu, se hace este cambio por que se sabía que los sistemas instalados eran copias ilegales y que en algún momento les iban a revisar. Todo este cambio se hace en la parte de los laboratorios y en los servidores de la facultad no tocando lo que es la administración de la misma porque se requiere de un plan de trabajo y personas que se dediquen a la implementación del software libre y esto lleva tiempo, cursos de capacitación. Además que todo el equipo de la parte administrativa cuando se compra son equipos nuevos de marca y estos tienen a su vez ya instaladas versiones de sistemas operativos con licencia OEM, solo una secretaria tiene instalado en su equipo Ubuntu de Linux, por tal motivo no hay mucho problema en esa parte. Esto quiere decir que es la única facultad en donde más han implementado software libre lo es también en su biblioteca en donde comenta que la universidad tiene un software para el control de la misma llamado unicornio y este no les cubrió sus necesidades, por tal motivo empezaron a desarrollarlos ellos mismos y se encontraron en el camino un software de código abierto hecho en Perl utilizando base de datos como Mysql y PostgresSQL, adaptándolo a sus necesidades. Esto se refleja en ahorro de dinero por parte de la facultad en cuanto a licencias se refiere además de que el sistema y las aplicaciones son estables y adaptables que únicamente provee el software libre o de código abierto que permite optimizar el software hasta en el detalle más ínfimo. No se encontró que en las mencionadas facultades se utilice software libre en el departamento administrativo.

Factibilidad de una implementación de software libre.

Factibilidad Técnica.

Para determinar la factibilidad técnica consiste en realizar una evaluación de la tecnología existente en la Facultad, este estudio estuvo destinado a recolectar

información sobre los componentes técnicos que posee la organización y la posibilidad de hacer uso de los mismos.

Para la implementación de un sistema operativo de software libre los puestos de trabajo tienen maquinas con hardware requerido para su instalación y es el siguiente:

- Procesador Pentium Core2Duo entre de 2.8 GHz a 3.0 GHz
- 1 Gb RAM
- 500 G. Disco duro.
- Unidad DVD-ROM
- Monitor 17' LCD
- Mouse, Teclado.

Evaluando el Hardware Existente la institución no requiere realizar inversión inicial para la adquisición de nuevos equipos, ni tampoco para actualizar los equipos existentes, ya que los mismos satisfacen los requerimientos establecidos.

En cuanto al software los equipos emplean los siguientes:

- Windows XP
- Office 2010

La institución cuenta con las aplicaciones necesarias para la realización de sus trabajos como es un sistema base que es Windows XP y sobre el corre el aplicativo office 2010 para las tareas ofimáticas. Y en contra posición el sistema libre tiene estas herramientas para trabajar de la misma manera.

Como resultado de este estudio se determina que la institución cuenta actualmente con la infraestructura tecnológica (hardware y software) necesaria para el funcionamiento del departamento.

Factibilidad económica.

Para determinar la factibilidad técnica se determinaron los recursos para implementar y mantener en operación el sistema a implementar, haciendo una evaluación donde se

puso de manifiesto el equilibrio existente entre los costos del sistema y los beneficios que se derivan de este, los cual permitió observar de una manera mas precisa las bondades del sistema propuesto.

	Windows	GNU/Linux
Costo Licencia c/u	$800	0
Costo ofimática c/u	$600	0

Tabla 5. Costo Licencias

Cabe aclarar que los precios son precios aproximados por el convenio que se tiene Microsoft-UMSNH. Para la implementación de un sistema de software libre, tomando en cuenta que no hay costo de inversión inicial por contar con el equipo necesario, y que este tiene las características necesarias para el sistema libre, el costo de adquisición del software es cero por el tipo de licencia solo se necesita descargarlo de internet, costo de una capacitación es cero por que se impartirá dentro de la institución a través de un maestro o alumnos de la misma área de sistemas, costo de mantenimiento y soporte es cero se da así mismo por los integrantes del área de sistemas que en ocasiones son mismos alumnos que imparten su servicio social. Por lo tanto no se requiere aportar capital para su funcionamiento.

Factibilidad Operativa.

Para una posible implementación sobre el software libre no hay una instalación en ningún equipo de la administración y no conocen el sistema nunca lo han visto y por lo tanto nunca lo han utilizado, en las entrevistas se ve que lo importante es darle prioridad al terminar el trabajo y no estar perdiendo tiempo por así decirlo en estar aprendiendo a utilizar algo nuevo o algo que nunca habían visto, al explicar las ventajas que pudiese tener el software libre dentro de la faculta comentando la seguridad, cero virus por mencionar algo aceptan que estaría bien implementarlo pero no están del todo bien seguros de un cambio tan radical.

Evaluación de factibilidad de uso de software libre.

Para este pequeño caso practico no es conveniente llevar a cabo una posible implementación de software libre aun teniendo las ventajas que se tendría utilizándolo. Esta claro que es difícil implementar un software libre en una institución por que al comienzo cuando se decide que software utilizar se escoge el software propietario por ser el de mas difusión, el que se enseña y se aprende en la educación de nuestro país o ser el mas difundido y por todo ese tiempo de utilizarlo ya es fácil utilizarlo. Pero esto esta cambiando en los últimos años con la incorporación de software libre o de fuente abierta en universidades tan solo el hecho de instalarlo en los laboratorio para que nuevas generaciones al termino de sus estudios estén del conocimiento de mas opciones de sistemas y que en su vida laboral no sea un ante ponente por no saber utilizar un sistema libre, y no estén mermando a una institución por no saber utilizar una herramienta que es mas segura, libre, personalizable al 100%, menor vulnerable, con actualizaciones de los sistemas con aproximado de seis meses y no años, no estar a expensas de una empresa a que saque un producto he ir inmediatamente a comprar o a pedir las licencias a planeación universitaria en el caso de la facultad que así lo hace, pero nadie puede ser obligado a utilizar lo que en su vida no ha utilizado retardando la entrega de sus trabajos y perdiendo tiempo por no conocer y esto es lo que menos se quiere en institución ya sea publica y privada por eso se mencionan estas propuestas para empezar a usarlo y así implementarlo con el tiempo de forma positiva.

- Que el encargado de la área de sistemas empiece a conocer sobre sistemas libres que se hayan implementado en otras universidades u organismos públicos y qué beneficios han adquirido.
- Difundir las ventajas que se tendrían al utilizar software libre dentro de la facultad, por ejemplo que no tiene costo, incrementa la seguridad.
- Difundir la necesidad y el deseo del cambio a los involucrados dando platicas, capacitaciones del como se maneja, de que es amigable, y es fácil.
- Quitar la idea de que el software libre no es “software gratuito” por el fenómeno en el cual se asocia lo gratis a productos de baja calidad o deficientes.

CONCLUSIONES

Hoy en día en cualquier parte es una necesidad utilizar una computadora para realizar nuestro trabajo, haciéndolo con menor tiempo y dando buenos resultados, el hombre es adaptable ante las necesidades que valla teniendo es por eso que en los últimos años mas organismos públicos y privados ven con bueno aspecto al software libre, porque ya no es necesario gastar miles de pesos o millones por adquirir sistemas que dan el mismo resultado que uno que realmente no tiene costo por licencia, pero si estos organismos iniciaron de cierta forma pagando para adquirir esos sistemas por la necesidad, facilidad y por no haber mas opciones, es lógico que se siguió en esa línea porque satisfacía y aun sigue satisfaciendo sus necesidades, pero el aspecto económico siempre es un factor a considerar en cualquier organismo por lo tanto sabiendo que actualmente existen sistemas que no se paga por su uso, se ha acrecentado mas su utilización en los últimos años al grado de un cambio total o parcial de todos sus sistemas, obteniendo muchos beneficios dándole valor agregado a sus servicios por ser organismos que manejan mucha información de carácter privado o publico y que esta debe de ser protegida pero al mismo tiempo de rápido acceso por demanda. Por lo tanto el software libre o software de fuente abierta cubre ampliamente con estas necesidades por tener a sus espaldas comunidades enormes de desarrolladores y por cada producto salen actualizaciones mas recientes que una empresa privada no podría sostener en nomina. Este trabajo no trata de forzar a un cambio sino de dar los aspectos que benefician y han beneficiado al cambio y que el mismo software libre genere esa demanda. Vemos países como Brasil, Venezuela, Bolivia por mencionar algunos que tienen como política el uso de software libre y que han ahorrado mucho al no utilizar software propietario y así dando trabajo a la industria local, desarrollando a la medida y con mejor calidad por la misma competencia al adquirir una licitación y el dinero queda en el estado. En el caso de la facultad de contaduría y ciencias administrativas sigue esa misma línea de solo conocer lo que es el software propietario y queda claro que el software libre esta lejos de implementarse una por el desconocimiento y otra por no saber el provecho que realmente se le puede dar a este tipo de sistemas libres.

BIBLIOGRAFIA

IEEE Std 729. 1993. *IEEE Software Engineering Standard 729-1993: Glossary of Software Engineering Termino-logy*. IEEE Computer Society Press, 1993.

Pardo, E. 1993. *Microinformática de Gestión*. Universidad de Oviedo, España. 117-118.

Turing, A. M. 1936.*ON COMPUTABLE NUMBERS,WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM*. Inglaterra. 29pp.

Free Software Foundation. (2007, Febrero). *La Definición del Software Libre*. Consultado el 20 de Junio 2012 en http://www.gnu.org/philosophy/free-sw.es.html.

Stallman, R. 2004. *Software libre para una sociedad libre (Ver. 1.0).* Traficante de sueños, España.

Alonso, A., C. Galán. 2004. *La tecnología y su divulgación: un enfoque transdiciplinar.* Anthropos, España.

Open Source Iniciative. 2012. [En línea]. Disponible en http://www.opensource.org/docs/osd

Montserrat, C., G. Gómez, J., S. Torres S. 2006. *Software libre vs Software propietario ventajas y desventajas.* Creative commons, México.

Rohaut, S. 2010. *LINUX- Preparación para la certificación LPIC-1(exámenes LPI 101 y LPI 102)*. Ediciones ENI, España.

Franco, M. O., Martínez, M. E. (2007, Noviembre). *El origen del Software Libre*. Revista RED. Disponible en http://www.labrechadigital.org/labrecha/articulos/el-origen-del-software-libre.html

Martínez, P. D. 2009. UNIX a base de Ejemplos (2ed). Lulu.com, Espana.

E. Kendall, J., & E. Kendall, K. (2005). *Analisis Y Diseño de Sistemas.* Mexico: Pearson Educación.

Henson-Apollonio, V., Longhorn, R., & White, J. (2002). *Aspectos Juridico-legales del Uso de Datos y Herramientas Geoespaciales en la Agricultura y en el Manejo de los Recursos Naturales. Manual de Conceptos Basicos.* México: CIMMYT.

Bermúdez Silva, I. A. (2008). *Debian GNU/Linux Para El Usuario Final.* Venezuela: Lulu.com.

Black Duck Software, Inc. (2012). *Open Source Resource Center Presented by Black Duck Software*. Recuperado el 30 de Junio de 2012, de http://osrc.blackducksoftware.com/data/licenses/

Centro Nacional de Referencia de Aplicación de las Tecnologías de Información y la Comunicación basadas en Fuentes Abiertas. (16 de Septiembre de 2010). *Cenatic.* Recuperado el 22 de 03 de 2012, de http://www.cenatic.es/publicaciones/onsfa?download=39%3Areport-on-the-international-status-of-open-source-software-2010

Cerón Estrada, M. A. (28 de Junio de 2008). *Grupo GNU/Linux Universidad del Cauca*. Recuperado el 28 de Enero de 2012, de http://gluc.unicauca.edu.co/wiki/index.php/%C2%BFSoftware_Libre_o_Software_de_Fuente_Abierta%3F

EL País. (22 de Abril de 2012). *El País.* Recuperado el 5 de Mayo de 2012, de Linux, el triunfo silencioso: http://tecnologia.elpais.com/tecnologia/2012/04/22/actualidad/1335117737_156353.html

Estepa Nieto, J. J. (Julio de 2007). *Software Libre Para El Desarrollo Del Tercer Mundo.* Recuperado el 29 de Mayo de 2012, de http://observatorio.cenatic.es/index.php?option=com_content&view=article&id=13

Gutiérrez, G. (12 de Abril de 2005). *Recursos e información sobre globalización, desarrollo y sociedad civil en América Latina*. Recuperado el 16 de Mayo de 2012, de http://www.globalizacion.org/analisis/GutierrezSoftLibreAdmPublica.htm

Hypatia. (Octubre de 2003). *Hypatia.* Recuperado el 17 de Mayo de 2012, de http://docs.hipatia.net/ica/SLibreAP.pdf

Jiménez Romera, C. (29 de Junio de 2002). *Biblioteca CF+S Ciudades para un futuro más sostenible*. Recuperado el 17 de Mayo de 2012, de http://habitat.aq.upm.es/boletin/n20/acjim.html

Jorba Esteve, J., & Suppi Boldrito, R. (2004). *Administración avanzada de GNU/Linux.* España: Universidad Oberta de Cataluya.

Masi Hernández, J. (2005). *Software libre: técnicamente viable, ecónomicamente sostenible, socialmente justo.* España: infonomia.

Montes L., S. R. (Diciembre de 2011). *http://es.scribd.com.* Recuperado el 10 de Junio de 2012, de http://es.scribd.com/doc/76504723/Licencias-de-Software-Libre-FLOSS#outer_page_5

Niño Camazón, J. (2011). *Sistemas operativos monopuesto.* España: Editex.

Organización Mundial de la Propiedad Intelectual. (1967). *Organización Mundial de la Propiedad Intelectual*. Recuperado el 21 de 06 de 2012, de http://www.wipo.int/patentscope/es/patents_faq.html

Perez i Noguer, Q. (s.f.). *GNU / Linux Introducción al Software Libre* (Vol. 1). España: Edit Lin Editorial, S.L.

Pons, N. (2005). *Linux: Principios básicos del uso del sistema.* (A. Brugués, Ed.) España: Ediciones ENI.

Rodríguez, G. . (Julio de 2008). *EL SOFTWARE LIBRE Y SUS IMPLICACIONES JURÍDICAS.* Recuperado el 21 de 06 de 2012, de http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0121-86972008000200007#aff*

Sandoval, H. (17 de Febrero de 2006). *Michoacán ahorra muchos millones con software libre.* Recuperado el 14 de Mayo de 2012, de http://www2.eluniversal.com.mx/pls/impreso/noticia.html?id_nota=50094&tabla=finanzas

Zuniga, M. (7 de Octubre de 2007). *E-arquitectura*. Recuperado el 15 de 05 de 2012, de Diferencias entre Software Propietario y Software Libre (FLOSS): http://blog.maz.cl/2007/10/diferencias-entre-software-propietario.html