

## **Conferencia “Software Libre Mitos y Realidad”**

*Permiso para copiar, distribuir y/o modificar este documento bajo los términos de la Licencia Creative Commons (reconocimiento, no comercial y sin obra derivada 2.1) salvo donde se incluya nota expresa.*

*Puerta Marcelo - [pumarfa@gmail.com](mailto:pumarfa@gmail.com)  
Versión del documento 0.2 – 19/03/2007*

*Entender el significado de las palabras “Software Libre” no es un tema complicado; sin embargo entender la filosofía subyacente y el alcance que estas palabras tienen en el entorno informático es un poco más complicado.*

*Nuestra tarea es acercar un nuevo punto de vista sobre el software libre. Especialmente viendo al Software libre desde la perspectiva de la industria de software y del usuario empresarial.*

*Hay una multitud de conceptos que debemos asimilar para entender el fenómeno del Software libre, pero aquí nos vamos a concentrar en unos pocos; aquellos que por diversos motivos se pierden de vista y que en realidad son los que traen luz a este tema.*

### **¿Qué es “el” Software?**

Se define como un conjunto de instrucciones, ejecutadas electrónicamente por una computadora o dispositivo electrónico que permite la realización de funciones o acciones.

Por ej.: Editar un texto, reproducir un DVD, encender la chispa de ignición del motor, etc.

### **¿Porqué es tan importante el Software?**

En nuestra vida cotidiana, nos interrelacionamos con cientos de dispositivos electrónicos, los cuales necesitan “Software” para funcionar. Y muchos de éstos aparatos contienen nuestra información personal. Pero como usuarios, somos incapaces de saber que es lo que el dispositivo hace con nuestra información, no por falta de pericia técnica, sino por falta de Libertad.

### **Un breve repaso a la historia del Software**

El software adquirió relevancia con el advenimiento de las computadoras. Anteriormente a ésta era los dispositivos, especialmente los de calculo, se desarrollaban con el objeto de cumplir su función pre definida. Si en algún momento se debe agregar una nueva funcionalidad, se debía construir un dispositivo nuevo; desde cero.

Las computadoras cambiaron esto. La computadora es capaz de realizar nuevas funciones para las cuales no ha sido creada originalmente, solamente con modificar las instrucciones electrónicas que sigue. El “Software”.

Desde el nacimiento de la ciencias de la computación, a mediados de los años 1940, y hasta principio de los años 1970, no existía una industria del Software. El software se compartía libremente entre los desarrolladores. Y no fue evidente hasta principios de los años 1980, que ésta metodología de compartir fue desapareciendo por la imposición de compañías comerciales las cuales escudadas detrás de formas legales, prohíben la practica de compartir el software.

### **Formas legales del Software**

El software desde su creación ha sido clasificado de forma legal bajo la forma de “propiedad intelectual”. Es equivalente a una obra de arte, una novela, un cuadro o una canción.

A simple vista es evidente que el Software tiene un objetivo muy distinto a las obras de arte. El software es una invención de uso practico; como las recetas de cocina.

Propiedad Intelectual o Copyrigh, es una frase que induce a creer que el “Software” tiene dueño, es propiedad de alguien. Lo cual es falso, las obras intelectuales tienen derecho-habientes, estos derechos no son asimilables al concepto de propiedad ya que las obras

intelectuales permiten uso no exclusivo y el fin de su protección es eminentemente social, no individual.

Que se utilice el desgraciado termino de "Propiedad Intelectual", un oximoron, para referirse a estos derechos induce erróneamente a pensar en la propiedad como "ajeno" y "dueño".

Entonces si le damos su real significado al software veremos que, contrariamente a lo que muchos piensan, la utilización de software está sujeto a un acuerdo una especie de contrato entre el editor del software o su desarrollador y el usuario del mismo. Este acuerdo se denomina "Licencia". El software puede tener una licencia "libre" o "privativa", ésta última llamada muchas veces como "End User License Agreement" (EULA), el acuerdo de licencia de usuario final.

El primer tipo de licencia de software libre fue la GPL, siglas de General Public License (Licencia Pública General), publicado por el proyecto GNU (GNU's not Unix) con el objetivo de permitir la distribución de software de manera libre, utilizando la filosofía de "dejar Copiar" (Copy Left) filosofía utilizada en los primero 30 años del desarrollo de las ciencias de la computación.

### ***Entonces, ¿qué es el Software Libre?***

El software libre es aquel cuya licencia de uso permite cuatro libertades fundamentales:

- Libertad de ejecutar el programa, para cualquier uso. (Libertad 1)
- Libertad de estudiar cómo funciona el programa, auditar y verificar las funciones del programa. (Libertad 2)
- Libertad de redistribuir copias, de modo que Ud. puede ayudar a la comunidad. (Libertad 3)
- Libertad de modificar el programa, y ceder sus mejoras al público. (Libertad 4)

Software Libre, no es una cuestión de precio, es una cuestión de libertad.

Los errores centrales cuando se habla de Software Libre, desde el punto de vista técnico-legal, con relación al Derecho de Autor/Copyright, son pensar:

que una vez que el autor de determinado software o de determinada modificación (parche), distribuye a determinada persona su obra bajo una licencia -GPL por ejemplo-, pierde el derecho de distribuirlo a otras personas - o eventualmente a la misma- bajo otras condiciones; que poner la obra bajo la GPL -u otra licencia-, implica su distribución universal (y gratuita); que los autores de modificaciones están obligados a distribuirlas.

La Constitución Argentina reconoce el derecho de los autores e inventores sobre sus creaciones e invenciones (art 17). El software es una creación, una obra intelectual, y como tal, protegida por la ley 11.723 de Propiedad Intelectual.

Según el artículo 2 de dicha ley, *el titular de los derechos sobre una obra tiene las facultades de disponer de ella*, de publicarla, autorizar su reproducción en cualquier forma, etc. Por lo tanto, todo aquello que signifique una reproducción del software que no cuente con la expresa autorización del autor, significa una infracción a sus derechos y constituye un delito a la ley 11.723, que especifica que *la pena será aquella aplicable al delito de estafa, que es de 1 mes a 6 años de prisión.*

El que utilice software bajo la Licencia GPL, está protegido de infringir la ley. Está utilizando software legal.

### ***Licencias de Software***

*Existen muchas licencias aplicadas a la distribución de software. Hemos nombrado solamente un par de ellas, pero existen muchas más. Para facilitar y aclarar como funcionan las licencias de software, explicaremos las más utilizadas:*

- **Licencia Privativa:** priva a quien la adquiere de permisos de: Estudiar, Modificar y

Distribuir. Restringe el Uso del software y en muchos casos hasta hacer público fallos que se descubran en el funcionamiento del software.

- **Licencia GPL:** garantiza las cuatro libertades. Utilizar, Estudiar, Modificar y Distribuir. Obliga a que cualquier código incluido sea liberado bajo las mismas condiciones; como así también los trabajos derivados.
- **Licencia LGPL:** garantiza las cuatro libertades. Utilizar, Estudiar, Modificar y Distribuir. No obliga a liberar los trabajos derivados.
- **Licencia BSD:** NO garantiza las cuatro libertades. No obliga a liberar los trabajos derivados y cualquiera puede redistribuir el software bajo una licencia privativa.

Otras Licencias muy utilizadas son: MIT (Massachusset Institute of Tecnologi), MPL (Mozzila Public License), OS (Open Source), MSSSL (MS Shared Source)...

Para conocer más licencias y los alcancs de cada una, puede visitar el sitio WEB de la Free Software Fundation (Fundación del software Libre) <http://www.fsf.org>

### ***¿Qué adquiere cuando compra Software privativo?... su Licencia***

Al adquirir un software privativo, el usuario (comprador) no se torna propietario de un bien, está solamente recibiendo una licencia de uso; que es un permiso para el uso, de forma no exclusiva.

No siendo permitido el acceso al código fuente o instrucciones ni a su redistribución, en algunos casos tampoco el usuario puede hacer público fallos en el software y se encuentran casos que el comprador ni siquiera es dueño del soporte en el que se ha entregado el software, como el CD-ROM en el que se encuentra el software para su instalación. Éstas características lo definen como privativo, ya que existe siempre un beneficiario permanente con la comercialización del software.

Un aspecto muy importante del software y su licencia o acuerdo de uso, es que no importa si el software está bajo los términos de una licencia libre o privativa; siempre mantiene su condición de "creación o invención".

El software generalmente se vende "AS IS" (como está) sin garantías explícitas del editor o desarrollador de que dicho software realizará la tarea para la que fue adquirido. Sin embargo, puede haber garantías específicas para situaciones muy específicas, las que se deben negociar y especificar en la licencia en el momento de la adquisición.

En el caso de adquirir Software Libre, está adquiriendo una licencia que le garantiza las cuatro libertades ya nombradas; imponiendo una única condición, el comprador o usuario NO podrá modificar las condiciones de la licencia original cuando ceda, venda o modifique el software.

### ***Software Libre Como Empresa***

*Habiendo planteado los hechos de "qué es realmente lo que se comercializa" cuando se desarrolla software, se vuelve más clara la situación de que el desarrollador de software está comercializando una obra intelectual y que los estereotipos de la industria de manufactura no son aplicables a la industria del software.*

### ***La ilusión de la manufacturación***

*Necesitamos comenzar notando que los programas de computación como cualquier otra herramienta o bien de capital, tiene dos clases distintas de valor económico. Tienen el valor de uso y el valor de venta.*

El *valor de uso* de un programa es su valor económico como herramienta. El *valor de venta* de un programa es su valor como una mercancía vendible (commodity). (En lenguaje económico profesional, el valor de venta es el valor como bien final, y el valor de uso es el valor como bien intermedio.)

Cuando la mayoría de la gente trata de razonar acerca de la economía de producción de

software, trata de asumir un ``modelo de fábrica" que esta basado en las siguientes premisas fundamentales.

1. La mayor parte del tiempo del desarrollador se paga por el valor de venta.
2. El valor de venta del software es proporcional a su costo de desarrollo (esto es: el costo de los recursos necesarios para replicarlo funcionalmente) y a su valor de uso.

En otras palabras, la gente tiene una fuerte tendencia a asumir que el software tiene las características de valor de un bien manufacturado típico. Pero ambas suposiciones son demostrablemente falsas.

Primero; si se realiza un estudio se ve claramente que el 95% del código de Software es escrito "in-house" (hecho en casa), para solventar problemas específicos y no como software de venta. Es el caso del software embebido en dispositivos que contienen microcontroladores. Este software por sus características, está fuertemente relacionado con el medio en el que funciona, y es de difícil replicación y reutilización. Por lo tanto, cuando el ambiente cambia, hay mucho trabajo para los programadores, ya que deben "mantener" el software para reflejar los cambios en el ambiente.

Cualquier ingeniero de software o analista de sistemas estará de acuerdo que el "Mantenimiento" compone la gran mayoría (mas del 75%) del trabajo por el que se le paga a un programador. De acuerdo a esto, la mayoría de las horas de programación se gastan en (y la mayoría de los salarios de los programadores se pagan por) escribir y mantener código "hecho en casa" que no tiene ningún valor de venta.

Segundo, la teoría que dice que el valor del software está acoplado a su costo de desarrollo o de reemplazo es mas fácilmente demolida examinando el comportamiento real de los consumidores. Hay muchos bienes para los cuales una relación de este tipo les cabe (antes de la depreciación) - comida, autos, herramientas. Hay también muchos bienes intangibles para los cuales el valor de venta se acopla fuertemente al costo de desarrollo y de reemplazo - derechos para reproducir musica o mapas o bases de datos, por ejemplo. Estos bienes pueden retener o incluso aumentar su valor de venta después que su productor se ha retirado del mercado.

En contraste, cuando el productor de software sale del mercado (o meramente si el producto es discontinuado), el mayor precio que los consumidores pagarán por él rápidamente cae casi hasta cero sin importar su valor teórico de uso o el costo de desarrollo de un bien funcionalmente equivalente. (Para confirmar esta afirmación, examine los contenedores de productos discontinuados en cualquier casa de venta de software.)

El comportamiento de los vendedores a consumidor final cuando un productor se retira ((fold)) es muy revelador. Nos dice que ellos saben algo que los productores no saben. Lo que saben es esto: el precio que un consumidor pagará se ve efectivamente disminuido por el *valor futuro esperado del servicio prestado por el productor* (donde ``servicio" aquí se entiende ampliamente para incluir mejoras, actualizaciones y proyectos siguientes).

En otras palabras, el software es una industria de servicios operando bajo la persistente pero infundada ilusión que es una industria de manufacturas.

### ***¿Es económicamente viable el software libre?***

Ya hemos visto que el desarrollador o "Propietario Intelectual" de la obra de Software mantiene en todo momento potestad sobre su creación; también es sabido que mucho del software libre, es también gratuito.

Esta situación se da porque el software libre no necesita dinero para que sea desarrollado. Aunque todas las compañías y el estado dejen de invertir en software libre (mediante programadores, código, donaciones, etc.), siempre habrá programadores que lo harán por motivaciones distintas a las económicas directas.

El software es una creación intelectual, aunque se incrementen los consumidores no significa que el o los autores originales tengan una pérdida patrimonial. Éste aspecto es el que se ha

tenido en cuenta para la elaboración de las leyes de protección intelectual.

Pero además de ello el software tiene una característica única a diferencia de otras creaciones humanas: puede ser desarrollado y mejorado por miles de personas, como se ha demostrado el núcleo Linux o el escritorio GNOME.

### ***Si no hay ganancias de ventas de software ¿que incentivos tendrán las empresas para desarrollar software libre?***

Para la inmensa mayoría de las empresas, el software es un gasto, no produce ganancias.

Para más del 90% de esa minoría de empresas que sí desarrollan software, dicho software también es un gasto, ya que ellas no se dedican a producir software para vender y ganar dinero, sino para uso in-house o personalizaciones a otras empresas. Para todas esas empresas el "software libre" sí significa ganancia de dinero: dinero no gastado es dinero ganado.

En pocas palabras, las empresas seguirán necesitando software, y ese software deberá seguir evolucionando y adaptándose al modelo real que evoluciona constantemente.

Es decir, haya o no haya inversión de las empresas para el desarrollo del software libre, éste seguirá creciendo.

### ***¿Se morirá la industria del software?***

El software siempre deberá ser desarrollado por alguien. Seguramente como hobby de muchos programadores pero lo más probable es que las empresas sean las que más inviertan. Además, independientemente de que el software sea libre o privativo, los costes de desarrollo inicial no son los más significativos en el coste total del ciclo de vida de un programa, sino los costes de adaptación y mantenimiento.

Si un programa no es modificado después de la liberación de su versión inicial, el modelo que representa se alejará (gap) cada vez más de la realidad. Esa diferencia se mantiene al mínimo con el mantenimiento posterior, cuyo coste total se estima en un 75% del coste total del ciclo de vida.

En todo caso la pregunta es: ¿morirá el la industria del software privativo?

### ***El Software Libre ¿No hará que los programadores pierdan sus puestos de trabajo?***

Se estima que solamente en USA sólo el 5% de los programadores desarrolla software privativo para venderlos como paquetes.

El restante 95% hace desarrollos para software que será usado en la empresa (in-house), mantenimiento y personalizaciones. Para este 95% el software libre es el paraíso, ya que evitan que programen una y otra vez funcionalidades que ya están disponibles en otros programas. Por otro lado, tal como lo demuestra la experiencia de IBM con Websphere y la fundación Apache, la mejor forma de reducir el coste de mantenimiento de un programa es hacerlo libre.

### ***Pero, ¿Y los costes de oportunidad de los programadores de software libre?***

Es una pregunta típica de los "puristas" economicistas. La barrera de entrada al mercado del software privativo es muy alta, las posibilidades de tener éxito son ínfimas y se necesita mucha inversión, especialmente de comercialización. Es mucho menos problemático liberar un programa como software libre, y esperar contribuciones externas para mejorarlo, que intentar comercializarlo.

### ***¿Pero acaso las licencias de distribuciones como RedHat o SuSE no son más caras que la de MS Windows?***

RedHat o SuSE está vendiendo no sólo un producto empaquetado, sino también el soporte y

garantía para miles de aplicaciones, no sólo el sistema operativo básico. Además hay que tener en cuenta que éstas empresas se especializan en el mercado de empresas y que hay otras opciones como Ubuntu(\*), Debian, Gentoo, Mandriva(\*), la misma SuSE(\*), Fedora, etc. que son totalmente gratuitas.

(\*) son empresas que distribuyen gratuitamente sus productos de software y que además ofrecen como producto independiente el soporte a todo el Software Distribuido.

### ***Si el software libre es tan bueno y eficiente, ¿porqué no se adoptó o extendió antes?***

Porque el software libre no es un competidor del software privativo, sino un sustituto (recuerde además que el mercado del software no es una mercancía [commodity]). Para hacer el cambio se necesita un cambio de actitud y tomar una decisión importante. Para hacer el cambio hace falta ver ejemplos exitosos de los early adopters.

De todas formas hay nichos que el software libre ha sido más exitoso que el software privativo desde hace muchos años, por ejemplo el correo electrónico, servidores web, servidores de ficheros, DNS, etc. En estos casos no tiene justificación el retraso de algunas organizaciones, especialmente universidades y organismos gubernamentales, para adoptar el software libre (más del 90% de los servidores DNS de Internbet son libres, más del 85% de los correos electrónicos pasan por servidores de software libre). Especialmente por la calidad demostrada, el ahorro de costes y flexibilidad de la infraestructura básica de la organización.

Quizás la respuesta es que se han dejado “engañar” (o han sucumbido a los “regalos”) de las estrategias comerciales de las empresas.

### ***¿No innova más el software privativo que el software libre?***

Es otro argumento repetido hasta la saciedad por los defensores del software privativo, especialmente de grandes empresas. Pero habría que pedirles que enumeren quienes han sido los inventores de las siguientes tecnologías: redes locales, Internet, navegador web, correo electrónico, thin clients, administración remota, vídeo digital en el PC, música, sistema gráfico basado en ventanas... ninguna de ellas ha sido inventada por Microsoft ni empresa productora de software privativo. De hecho la mayoría de esas tecnologías son anteriores a 1990 y surgieron en entornos que hoy consideraríamos de “código abierto”.

### ***¿No afectarán negativamente las patentes al software libre?***

Si, como también al software privativo. Cualquier sistema de software de unos pocos miles de líneas seguramente está afectada por unas cuantas patentes. La única forma de estar seguro es borrando todo el software que haya en los discos duros de las computadoras.

El caso de las patentes en en la industria del software es un tema que merece mucho más tiempo para desarrollarse. Pero en pocas palabras, aquellos a quienes más va a golpear la imposición de patentes de software será a las pequeñas y medianas empresas.

### ***¿Cómo será el negocio informático?***

Seguramente muy distinto en lo que se refiere a los paquetes cerrados de software, muy pocos cambios en el mercado de mantenimiento, soporte y personalizaciones y precios más bajos, sobre todo si hay una alternativa de software libre.

En cualquier caso el mercado lo definirán los consumidores, y eso parece no agradar a algunos empresarios

## **Ventajas Competitivas del Software Libre en las Empresas**

### ***¿Porqué Utilizar Software Libre en las Empresas?***

- Independencia Tecnológica.
- Libre elección del proveedor de tecnología.
- Importante reducción de costos de licenciamiento.

- Alta escalabilidad, con menores costos.
- Mejora en la seguridad de los procesos de Información.
- Legalidad de la implementación de Software.

### **Independencia Tecnológica**

La independencia tecnológica garantiza a la empresa la posibilidad de implementar distintas soluciones de software, sin encontrarse atada a las decisiones unilaterales de terceras personas (empresas editoras de software).

Garantiza que la empresa tiene el control sobre sus soluciones de software, pudiendo en todo caso modificar o auditar el software que está utilizando, modificarlo y mejorarlo a voluntad.

Por ej. : Al auditar el software puede eliminar fallos indeseados o funciones peligrosas. (Eliminando el caso conocido de las claves secretas ocultas en el W NT.)

No estar atado a las sucesivas versiones de paquetes ofimáticos; con modificaciones que fuerzan a cambiar toda la base instalada por incompatibilidades en los documentos.

### **Libre elección del proveedor de tecnología**

Las cuatro libertades de la GPL, garantizan la competencia igualitaria entre las empresas proveedoras de software.

De éste modo las empresas que se informatizan o migran a SL, pueden realizar una evaluación realista de las ofertas de los proveedores. Ya que los proveedores compiten por el servicio brindado, que es en última instancia lo que una empresa debe adquirir.

### **Importante reducción de costos de licenciamiento.**

Los costos de adquisición de las licencias para la instalación o informatización son uno de los factores decisivos a la hora de elegir la solución tecnológica.

Una empresa que utilice SL para el desarrollo de sus actividades tiene un costo de licencias de software tendiente a 0 (cero).

Un estudio realizado sobre los costos de implementación de diversas soluciones tecnológicas revela que el costo de el despliegue de soluciones basadas en GNU/Linux y SL; es menor al 50% que el de soluciones basadas en Licencias privativas.

### **Alta escalabilidad, con menores costos**

Considerando los costes de licencias, es natural que la escalabilidad sea mucho más económica. A esto se suma ventajas basadas en la tecnología aplicada al negocio, lo que permite ofrecer más servicios con un costo de infraestructura menor.

Un ejemplo típico es la utilización de GNU/Linux y SL para servicios de correo electrónico. Es posible Implementar un servicio básico e ir incrementando progresivamente los servicios y la cantidad de cuentas que se administran, sin la necesidad de adquirir más licencias para brindar el mismo servicio o licencias de software de terceras partes para brindar protección del Spam o el Malware.

### **Mejora en la seguridad de los procesos de Información**

Debido que el SL viene acompañado del código fuente, las empresas pueden auditar dicho código o encargar la tarea a otras empresas especializadas.

Ésta tarea puede devenir en nuevas versiones de software, con la inclusión de parches que solucionan fallos, o la eliminación de código ineficiente o funciones que no se implementan en el código.

Estas modificaciones, pueden ser un subproducto de la empresa y es factible su

comercialización.

## **Legalidad de la implementación de Software**

El software libre, por sus características, es software perfectamente legal. Su sistema de licencia se ajusta a la legislación Argentina, ley 11.723 de derechos de autor.

## **Mitos sobre la implementación de Software Libre**

Existen muchos mitos sobre la implementación o adopción de software libre por parte de los usuarios finales o empresas. Generalmente estos "mitos" han surgido hace muchos años atrás y, aunque en su momento pudieron ser cierto, actualmente están superados por completo.

En otros casos han surgido a raíz de mala información y de verdades a medias de campañas publicitarias de las empresas editoras de software.

## **La curva de aprendizaje es mayor**

Este es uno de los puntos muy esgrimidos cuando se propone la conversión de una organización a software libre.

La realidad es que si ponemos a dos señoras que nunca han tocado una computadora personal, probablemente tardaran lo mismo en aprender a usar una con windows que Gnome o KDE, la otra...

La curva de aprendizaje de un usuario dependerá más de la política informática y de formación que implemente la organización que del software utilizado. En realidad muchos usuarios han aprendido a utilizar mejor los recursos informáticos cuando se ha llevado a cabo la migración a software libre debido a una activa política de formación. También esta situación es propicia para quitar vicios adquiridos por los usuarios a o largo de años de una utilización incorrecta de los recursos; debido a que no han tenido una formación formal en el uso de las herramientas informáticas.

## **El software libre no tiene garantía proveniente del autor**

Este es el ejemplo de verdades a medias ya que los contratos de software privativo tampoco se hacen responsables por daños económicos, y de otros tipos por el uso de sus programas.

El software generalmente se vende "AS IS" (como está) sin garantías explícitas del fabricante o editor, sin embargo, puede haber garantías específicas para situaciones muy específicas. Las cuales generalmente se pueden negociar con proveedores de software libre y en condiciones más igualitarias.

Una PyME, difícilmente pueda negociar un contrato de licencias con garantías con empresas cuyo capital financiero es gigantesco.

## **Se necesita dedicar recursos a la reparación de erratas**

Este es un punto especialmente interesante. En el caso del software libre, una empresa puede dedicar recursos a la corrección de errores del software y a agregar nuevas funcionalidades.

En el software privativo es imposible reparar erratas.

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

Este es un mito histórico. Hay un número cada vez mayor de usuarios y empresas que aseguran y respaldan, que las interfaces gráficas más populares en el software libre (KDE, GNOME y el manejador de ventanas WindowMaker) son ya lo suficientemente estables para el uso cotidiano y lo suficientemente amigables para los neófitos.

En el caso de la multimedia, no hay nada que el usuario eche de menos al utilizar software

libre. Además de las capacidades avanzadas que se pueden encontrar y que han sido explotadas por empresas como HP y DreamWorks, Lucas Films y un número cada vez más grande de compañías, las cuales basan sus sistemas para animación 3D y renderizado en software libre.

**La mayoría de la configuración de hardware no es intuitiva, se requieren conocimientos previos acerca del funcionamiento del sistema operativo y fundamentos del equipo a conectar para lograr un funcionamiento adecuado.**

Esto es una verdad a medias. La mayoría de los usuarios recurre a un técnico para resolver los temas de configuración e instalación de software y hardware en sus computadoras.

Sin embargo, si decide acometer la tarea de hacerlo en sistemas con software libre, la documentación referente a la configuración del hardware es tan explícita y detallada que permite al usuario neófito profundizar en el conocimiento de su hardware en muy pocas horas y una vez teniendo ese conocimiento la configuración se vuelve trivial.

**Únicamente los proyectos importantes y de trayectoria tienen buen soporte, tanto de los desarrolladores como de los usuarios; sin embargo existen muchos proyectos más pequeños y recientes que carecen del compromiso necesario por parte de sus usuarios o desarrolladores para que sean implementados de manera confiable.**

Este argumento no explora en profundidad cuales son los proyectos importantes en relación a los proyectos pequeños. Estos proyectos importantes que tienen un excelente soporte cubren más del 90% de las necesidades de cómputo del usuario promedio.

En el caso de proyectos pequeños, el soporte suele ser casi personalizado, y si el proyecto es capaz de atraer la cantidad de usuarios necesarios, fácilmente se convierte en un proyecto de envergadura y con excelente soporte. Un ejemplo a mencionar es el proyecto de Joomla, proveniente de una comunidad de desarrolladores.

**El usuario debe tener nociones de programación, ya que la administración del sistema recae mucho en la automatización de tareas y esto se logra utilizando, en muchas ocasiones, lenguajes de guiones (perl, python, shell, etc).**

Como respuesta se puede decir que existen en la actualidad muchas herramientas visuales que permiten al usuario no técnico llevar a cabo tareas de configuración del sistema de una manera gráfica muy sencilla sin la necesidad de conocimientos de programación.

Por otra parte, para el administrador o usuario avanzado, la capacidad infinita de la automatización de tareas por medio de la programación de script, es mucho más que una ventaja.

Un estudio realizado por la consultora especializada RFG, demuestra que es necesario menos administradores de sistemas cuando los sistemas instalados son Software libre, especialmente derivados de \*NIX.

Caso	Servidores por Administrador
Linux	44
Windows	10

**En sistemas con acceso a Internet, se deben de monitorear constantemente las correcciones de bugs de todos los programas que contengan dichos sistemas, ya que son fuentes potenciales de intrusión.**

Esto es completamente cierto y especialmente en el software privativo. También se deben de monitorear constantemente las correcciones de bugs de todos los programas y además es imposible reparar las vulnerabilidades (que en su mayoría son reparaciones triviales) por uno mismo sino que hay que esperar a que la compañía fabricante libere la actualización y en

algunos casos hay que pagar dinero extra por obtener esta.

En el caso del software libre existen herramientas automatizadas de actualización de paquetes como apt-get, redcarpet, rpmget pero se pierde la opción de compilar por uno mismo a la medida o tener aplicaciones sin un canal.

Una diferencia apreciable es que en el software libre, es posible realizar una auditoría y monitorear que la información de correcciones, no conlleve un envío por parte del usuario de información privada...

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

Hay quienes ven esto como una fortaleza, porque se pueden encontrar desde distribuciones especializadas en sistemas embebidos con muchas limitantes de almacenamiento y dispositivos periféricos de uso especializado hasta distribuciones optimizadas para su uso en servidores de alto rendimiento con varios procesadores y gran capacidad de almacenamiento; pasando por las distribuciones diseñadas para su uso en computadoras de escritorio y entre las cuales se encuentran las diseñadas para el usuario neófito que son muy fáciles de instalar y utilizar y las diseñadas para el usuario avanzado con todas las herramientas necesarias para explotar el software libre en todo su potencial.

Cabe notar que la posibilidad de crear distribuciones completamente a la medida para atacar situaciones muy específicas es una ventaja que muy pocas marcas de software propietario pueden ofrecer y que Microsoft está tratando de imitar con el lanzamiento de sus nuevos productos.

**Material consultado para realizar éste trabajo:**

“La catedral y el Bazar” de de Eric S. Raymond - 1998.

“El Caldero Magico” de Eric S. Raymond - Junio de 1999.

“Anti Carranza” de Diego Saravia - Noviembre de 2006.

“II Libro Blanco del Software Libre” A. Abella, M. A. Segovia.

“Introducción a las licencias de Software Libre” Jorge Nonius – abril de 2002.

Sitios Web:

<http://www.fsf.org> – Free Software Foundation.

<http://www.opensource.org> – Fundación Open Source.

<http://www.softwarelegal.com.ar> – Organización Software Legal Argentina.

Audio visuales de :

“RevolutionOS”

“Entrevista a Richad Staldman de TVE Canal 3”