Universidad Nacional Autónoma de México
Escuela Nacional de Artes Plásticas
Posgrado en Artes Visuales

Software Libre, herramienta alternativa para la creación visual digital

Tesis que para obtener el grado de
Maestra en Artes Visuales presenta:

Martha Irene Soria Guzmán

Director de Tesis:
Juan Antonio Madrid Vargas

México D.F. Junio 2012

UNAM
POSGRADO

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

¡COPIA ESTA TESIS!

Esta obra está bajo una licencia Atribución No comercial Licenciamiento Recíproco 2.5 México de Creative Commons. Para ver una copia de esta licencia, visite http://creativecommons.org/licenses/byncsa/2.5/mx/ o envie una carta a Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.

Eres libre de:
* copiar, distribuir y comunicar públicamente la obra
* hacer obras derivadas

Bajo las condiciones siguientes:

Atribución — Debes reconocer la autoría de la obra en los términos especificados por el propio autor o licenciante.
No comercial — No puedes utilizar esta obra para fines comerciales.
Licenciamiento Recíproco — Si alteras, transformas o creas una obra a partir de esta obra, solo podrás distribuir la obra resultante bajo una licencia igual a ésta.

A Chema

por sembrar en mi la semilla de la cultura libre,
por allanar senderos,
por abrir caminos,
por cambiar vidas.

A mi hermano Luis por su compañía y relajación diaria.
A Juan Ramón, por impulsarme a comenzar la aventura.
Al maestro Juan Antonio Madrid Vargas, por su confianza

A Lila Pagola por su ayuda inmensa e invaluable.

Agradecimientos especiales:

A Lilia Soria, Yanina Flores, Julian López Huerta, Gerardo Flores, Mauricio Juárez Servín, Tania de León, José Santorcuato, Christian Oyarzún, Claudia González, Ignacio Nieto, Javier García Alfaro, Ricardo Vega, Fabricio Caizza, Inne Martino, Martín Eschoyez, Felipe Sanches, Dave Crossland, Andrea Sagardoy, Guillermo Espertino

A toda la comunidad de software y cultura libre en latinoamérica
esta tesis es de todos.

Índice

Prólogo 15
Introducción 17

Capítulo 1. El software y el diseño gráfico, la mancuerna que cambió la manera de crear 23

1.1. Antes de comenzar, algunos apuntes sobre la definición de software 24
1.2. Un vistazo a la historia del uso de software en la creación gráfica digital 28
1.3. Flusser, la caja negra y una reflexión al software como herramienta. 33
1.4. Software y creación gráfica digital 44
1.5. El software como instrumento, material y medio 49

Capítulo 2. Software Libre en la creación visual digital. 65

2.1. Un poco más de historia: surge una contrapropuesta, movimiento del software libre 66
2.1.1. Richard Stallman,y el proyecto GNU. Linus Torvalds y LINUX 66
2.1.2. Definición de software libre 69
2.1.3. Las 4 libertades, la ideología detrás del movimiento del software libre 71
2.1.4. Software libre y software privativo: licencias que “permiten”, licencias que “privan” 77

2.2. Software libre en el terreno de la creación visual digital 80
2.2.1. Caraterización, problemáticas y diferencias con el software privativo.
a) El aspecto técnico 80
b) La parte ética y social 85
2.2.2. Problemáticas de la adopción del software libre en la creación gráfica digital 91
a) Generalidades
b) Paricularidades

Capítulo 3. Software Libre y procesos creativos. 107

3.1. Mi experiencia con el software libre 108
3.2. Otras experiencias
3.1.1. Creadores usuarios de software privativo 118
3.1.2. Creadores usuarios de software libre 119

3.3. Procesos creativos y código libre ¿cómo influyen en la creación? 125

Capítulo 4. Antes de concluir. Hacia la enseñanza del FLOSS, ¿por qué enseñar con software libre? 133

4.1. La enseñanza como estrategia de difusión, que el creador se acerque al software libre. 134
4.2. ¿Por qué enseñar software libre? 139
4.2.1. Motivar la generación de conocimiento, un software que fomenta la colaboración 140
4.2.2. No hay límites 143
4.2.3. No existe la herramienta correcta 144

4.3. Un ejemplo de enseñanza con software libre 147
44. La parte ética 149

5.-Conclusiones 155

6.- Anexos I
6.1. Programas Libres para el diseño y creación gráfica digital. I
6.2. Selección de color y software libre XIV

7.- Bibliografía 177

"Graphic design appears to have settled into a complacent middle age, content to be in the thrall of corporatism, branding, marketing, focus groups and quick-fix makeover culture. Even in its more radical guises, design has become self-admiring and masturbatory – a condition utterly alien to innovation and the forging of new directions."

Adrian Shaughnessy**

** El diseño gráfico parece haberse estacionado en una complaciente edad media, el contenido parece ser esclavo del corporativismo, del branding, del marketing, de los grupos de enfoque y de la cultura del maquillaje de reparaciones rápidas. Incluso en sus formas más radicales, el diseño se ha convertido en auto admiración y masturbatorio - una condición totalmente ajena a la innovación y el establecimiento de nuevas orientaciones". Trad.a.

Prólogo

Esta investigación responde a algunas inquietudes que se generaron en mí a raíz de un taller que cursé en agosto del 2006 en el Laboratorio Arte Alameda. El taller se titulaba: Tecnología y Auto-promoción, Medios Electrónicos para Promover el Arte, las Humanidades y las Organizaciones Civiles (MEPART), que fue impartido por José Serralde. Ahí tuve mi primer acercamiento con el Software y la Cultura Libre y comenzaron mis reflexiones sobre la responsabilidad como creadores, dentro de una sociedad con un flujo de información que nos rebasa, con cambios y evoluciones constantes.

Para entonces aún no utilizaba de manera directa programas de edición de imágenes o de vectores que fuesen libres, pero conocía de la existencia de Gimp y de Inkscape como homólogos de Photoshop© y de Illustrator© respectivamente y comenzaba a conocer sus características, sin embargo, nunca me atreví a usarlas por completo.

No fue sino hasta el 2009 cuando fui parte de un proyecto para la Organización Paramericana de la Salud, que consistió en la elaboración de un interactivo para mostrar los datos de salud recolectados en Chiapas en el año 2007. Yo me encargué de realizar el diseño. El líder del proyecto pidió al equipo creativo que trabajáramos las propuestas con software libre, debido a la naturaleza de difusión que tendría el interactivo, para evitar la compra de licencias y para que la información tuviera la apertura suficiente para quien la consultara sin que corriera con la suerte de permanecer codificada de modo casi secreto y evitar que sólo ciertas aplicaciones lo puedan leer y permitir su consulta, como sería el caso de haber utilizado Flash, por ejemplo.

Utilicé por primera vez Inkscape y elaboré mi primer diseño íntegramente en este programa. El proyecto cumplió las expectativas y me probé por primera vez, que podía realizar un diseño con herramientas libres y cumplir con el mercado laboral.

Con la posibilidad de usar una herramienta distinta a la que estaba acostumbrada y la difi-

cultad que me representó migrar de software privativo a software libre[1], comenzaron las reflexiones acerca de la manera en la que yo, como creadora gráfica digital estaba acostumbrada a generar una obra. Comencé a cuestionarme respecto al concepto de "herramienta" y hasta qué punto ésta es nuestra y hasta cual otro, somos dependientes de ella. Me formulé algunas preguntas que en gran parte dieron origen a esta tesis: ¿Soy capaz de generar un diseño y/o una obra con una herramienta distinta al Photoshop©, al Illustrator© y al Indesign©? ¿Soy capaz de crear un objeto cultural digital sin depender de una sola marca de herramienta digital? El hecho de usar una copia no autorizada del software en mi quehacer profesional, ¿introduce a mi profesión en una práctica inmersa en la ilegalidad?, ¿debemos, profesionales del diseño y arte digital, cuestionar el tipo de herramientas digitales que usamos para crear?.

En Mayo del 2009 decidí aplicar para el ingreso a la Maestría de Artes Visuales en la Academia de San Carlos, con el tema del software libre como herramienta para la creación en el diseño y el arte, como una oportunidad para desarrollar y aclarar estas y otras inquietudes que había venido formulando a lo largo de 3 años y que al final, desembocaron en el desarrollo de esta tesis. Al mismo tiempo, migré por completo a software libre, uso una versión libre de GNU/Linux como sistema operativo e Inkscape, Gimp y Scribus principalmente para llevar acabo todo mi trabajo profesional.

Sirva pues, este trabajo de investigación para documentar la experiencia y como iniciativa para el análisis de la relación creador visual digital y software.

Introducción

El diseño gráfico, así como algunas expresiones artísticas en la era digital, se han visto influenciadas por el uso del software como herramienta. El uso de programas computacionales como auxiliares en la generación de objetos culturales y su posterior circulación digital, es cosa de todos los días. Sin embargo, a pesar de la cotidianeidad de su uso, poco se reflexiona respecto al carácter "privativo" de estos programas, a la legalidad y al posible "oscurecimiento" que representa el funcionamiento de los mismos.

A lo largo de esta investigación se reflexiona sobre el uso de las herramientas para la creación visual digital que utiliza el diseñador gráfico. Se plantean algunos de los problemas que representa el uso de software patenta- do y cómo esto puede afectar al proceso de creación. Como una posible solución a éste y otros problemas, se explora el uso del software libre como herramienta alternativa para la creación gráfica digital, argumentando sus ventajas y explorando la factibilidad de su uso en el ámbito profesional aplicado en casos reales.

De esa forma, en el capítulo uno se explora el marco teórico del tema y se retoma la definición de software para luego exponer algunas consideraciones de la relación entre el software y la creación gráfica digital desde sus inicios; con un breve recorrido por la historia de la inclusión de los programas de computación para la generación de gráficos, hasta una reflexión respecto a la "lejanía" entre creador y la herramienta digital, las situaciones que ha representado el uso del software privativo como instrumento para la creación y cómo éste se convierte en algo más que una simple herramienta.

El capítulo más largo, el capítulo dos, entra de lleno al marco conceptual. Habla de la dupla del software libre y la creación gráfica digital. Para ello, se muestra primero la historia del movimiento del software libre, las 4 libertades y la ideología que lo sustentan, para después pasar a una breve descripción de las licencias que diferencian al software libre del software de patente. Por último, un recorrido por las ventajas de los programas libres.

En el tercer capítulo hablo de mi experiencia con el software libre a manera de responder una de las hipótesis de la investigación, la cual sugiere la posibilidad de prescindir del uso de cualquier tipo de software privativo para la creación gráfica digital. Para ello, cuento la experiencia de migrar 100% al software libre durante el proceso de esta investigación. De igual forma, hago una breve descripción de la experiencia de otros diseñadores que utilizaron herramientas libres por primera vez. Así mismo, integro algunas fuentes primarias recolectadas en la estancia de investigación que realicé de Marzo a Junio del 2011 en la ciudad de Córdoba Argentina, entrevistando a algunos artistas y diseñadores que usan software libre desde hace tiempo en su quehacer profesional.

Por último, el capitulo 4 se refiere a la enseñanza del software libre, que a su vez, se convierte en una propuesta para una futura investigación a manera de profundizar y continuar con el tema.

Así pues, los objetivos que se persiguen con esta investigación son: Conocer el software libre y explorar su uso como herramienta en la creación gráfica digital; conocer, y caracterizar dos tipos de software en la creación gráfica digital, por un lado el software de patente, el cual es llamado también software privativo y por otro el software libre; cuestionar el paradigma del uso y enseñanza de un sólo tipo de software para el diseño y las artes gráficas, que subyace en la utilización de "marcas únicas" de software; mostrar experiencias del uso del software libre en la creación gráfica digital, tanto propias como de otros artistas y diseñadores; reflexionar y analizar si es viable y pertinente el uso de software libre en el ámbito profesional de la creación gráfica digital.

Hipótesis de esta investigación: es importante considerar el uso del software libre en la creación gráfica digital, y por lo tanto, es indispensable su enseñanza a nuevas generaciones de creadores gráficos; el software libre puede funcionar en el ejercicio profesional de un creador gráfico digital, es decir, puede ejercer creativa y eficazmente usándolo.

%%title:

Capítulo Uno

<title>
El software y el diseño gráfico la mancuerna que cambió la manera de crear
</title>

En este primer capítulo, se pretende abordar el marco teórico de la investigación, ampliando la descripción de la problemática respecto a la mancuerna de la creación gráfica digital y el uso de un determinado tipo de software. Se pretende precisar el marco de referencia en base al cual girará la investigación, esto es, explorando y reflexionando a cerca del uso del software de patente como instrumento para la creación. Así pues, el objetivo general de este primer capítulo es cuestionar el paradigma de la relación creación-software y comenzar con el reconocimiento de una nueva alternativa para dicha relación.

1.1. Antes de comenzar, algunos apuntes sobre la definición del software.

Desde finales de la década de los 80's, la palabra software comenzó a popularizarse y ser usada entre un número mayor de personas y profesionales, práctica que se incrementó con el correr de las décadas.

El uso del concepto es muy extendido y a veces, poco entendido. Es por eso que haremos una leve revisión sobre la definición de SOFTWARE, palabra en inglés que ya ha sido adoptada y aceptada por la Real Academia de la Lengua como un anglicismo, pero que también ocupamos con regularidad en su traducción al español: PROGRAMA.

¿Qué es un softwareo un programade computadora?. El software se refiere a la parte "lógica de una computadora", la Real Academia Española lo define como "el conjunto unitario de instrucciones que permite a un ordenador realizar funciones diversas, como el tratamiento de textos, el diseño de gráficos, la resolución de problemas matemáticos, el manejo de bancos de datos, etcétera"[1]. Los programas nos sirven como herramienta para generar acciones con la computadora. Se trata, pues, de una serie de instruccionesen un lenguaje formal que pueda ser escrita por una persona y traducida por la computadora en forma de acciones.

En el documental "Codename: Linux", realizado en el 2001 en Francia, Richard Stallman, hace una analogía entre el software o programa computacional y una receta de cocina, en él, explica:

"Hay muchos puntos en común entre un programa y una receta de cocina; con una lista de etapas que hay que seguir y reglas que determinan en que momento se ha terminado o cómo dar marcha atrás, y al final, se obtiene un cierto resultado"[2]

Así pues, se trata de recetas minuciosamente detalladas para la solución de un determinado problema, que puede ir desde hacer una suma hasta escribir una carta, crear un vector o

Hay muchos puntos en común entre un programa y una receta de cocina

editar un video. Estas recetas están escritas en alguno de varios lenguajes formales, de una manera muy similar a como la música se escribe usando notaciones propias. Estos lenguajes formales con los que está escrito el programa pueden en algunos casos traducirse a través de un compilador al llamado "lenguaje de máquina", produciendo una lista de instrucciones que puede ser ejecutada automáticamente y a gran velocidad por una computadora para resolver ese problema específico.[3]

Un software puede ser escrito por cualquier persona que conozca alguno de estos lenguajes formales o lenguajes de programación, algunas de esas personas lo hacen a manera de pasatiempo y/o voluntariamente, algunos otros de forma personal para resolver su propias necesidades en materia de informática y otros más, trabajan para grandes corporaciones que luego venden el derecho de uso de ese programa, o la adecuación del mismo a necesidades específicas de sus clientes. Casi todas las organizaciones de envergadura mediana cuentan con personal capaz de escribir programas simples para sus tareas habituales.[4]

A pesar de que hoy en día, la distribución del software resulta ser muy extendida y su uso, muy habitual; en la década de los 60's y 70's cuando comenzaron a aparecer los primeros programas computacionales, éstos eran concebidos y entendidos de una manera muy distinta.

Richard Stallman comenta en su libro Software Libre para una comunidad libre, que al principio, el software se creaba para cada determinado hardware y su distribución era pequeña, generalmente, se quedaba dentro de una sola empresa y existía la confianza de que un equipo de programadores corregiría dicho software en caso de tener errores, o bien, lo adaptaría a nuevas necesidades. El software como producto comercialestaba aún lejano.

Incluso, entre los propios programadores (es decir, entre los propios creadores y autores intelectuales del software) era muy común compartir programas, pedir parte del código fuente[5] e incluso mejorarlo entre todos: "El acto de compartir software no se circunscribe a nuestra comunidad en particular: es tan antiguo como los propios ordenadores, lo mismo que compartir recetas es tan viejo como la cocina."[6]

En 1969, AT&T Labs publica la primera versión del Sistema Operativo UNIX, creado por Dennis Ritchie, Ken Thompsony y Douglas McIlroy[7], este último, resumió en 3 premisas la filosofía de este sistema: 1) escribir programas que hagan una sola cosa y que lo hagan bien,

Documental Code Linux,
disponible en:
http://www.youtube.com/watch?v=cwptTf-64Uo

2) escribir programas que trabajen juntos y 3) escribir programas que manejen flujo de texto, pues esta es una interfaz universal.[8] El modelo de licenciamiento de este sistema operativo contemplaba dos tipos de licencias: comerciales y académicas; estas últimas eran libres, es decir, su código fuente era visible y además, era gratuito[9]. Hoy en día, otros sistemas operativos como OSX de MAC y GNU/LINUX, están basados en UNIX.

Pero no fue sino hasta 10 años después, que el uso del software comenzó a popularizarse, y más aún, con la entrada al mercado de computadoras personales de escritorio que se volvían cada vez mas accesibles, poco a poco, la cultura del uso y compartición del código que fue cosa de todos los días entre los programadores de los 70's, se transformó paulatinamente en la "privatización" del código como una forma de creación cerrada que desembocó después en el uso de patentes para el mismo.

Christian Oyarzún, artista electrónico Chileno, explicaba en entrevista[10] su acercamiento con el software:

"Yo aprendí a programar por folletines y en papel, la noción de que software tenía un propietario no existía, uno aprendía BASIC[11]como quien aprende inglés, y lo escribía, eso fue para mi la experiencia en los 80's de lo que significaba el software"

En los 80's, las cosas comenzaron a cambiar. Siguiendo a Stallman, él menciona que algu-

Christian Oyarzún
Artísta electrónico chileno en entrevista, abril 2011. Santiago de Chile

nas empresas pioneras como Symbolics comenzaron a contratar cientos de programadores para crear programas que no pudieran compartirse y fueran utilizados solo por la empresa. Algunas computadoras modernas de la época comenzaban a tener su propio sistema operativo para el cual, en ocasiones se necesitaba firmar un acuerdo de confidencialidad para obtener una copia ejecutable:

"Todo ello significaba que antes de poder utilizar un ordenador tenías que prometer no ayudar a tu vecino. Quedaban así prohibidas las comunidades cooperativas. Los titulares de software propietario establecieron la siguiente norma: «Si compartes con tu vecino, te conviertes en un pirata. Si quieres hacer algún cambio, tendrás que rogárnoslo». "[12]

Comenzaba la era de la comercialización de un resultado intelectual, de la venta de un producto, que en poco tiempo, se convertiría en un nuevo negocio muy redituable para algunas empresas.

Entendiendo al software como una "creación intelectual", en las últimas décadas, se han tomado diversos caminos para la distribución, comercialización y sobre todo, para el permiso de uso que le daremos a dicha creación.

Algunas empresas de creación de software no permiten que los usuarios sean capaces de ver el código con el que están escritos sus programas, es decir, imposibilitan al usuario final

que conozca la receta con la que fue creado el programa. Al adquirir estos programas, se está pagando una licencia que autorice el uso legal de ese software y no el programa en sí mismo. En pocas palabras, se paga el permiso para usarlo.
En el siguiente apartado, hablaremos de cómo éste hecho ha repercutido en la creación gráfica digital.

1.2. Un vistazo a la historia del uso de software en la creación gráfica digital

Ya hemos visto un brevísimo recorrido por el concepto de software o programas de computadora, sin embargo, lo que nos atañe en esta sección es hablar de la relación que sostiene con el diseño gráfico y arte digital, tratando de respondernos algunas preguntas: ¿cuál es la relación entre el software y el campo de la creación gráfica digital y cuales han sido sus repercusiones? ¿ha condicionado el uso del software la manera en la que creamos digitalmente?

Uno de los primeros programas en generar gráficos asistidos por una computadora, y que utilizaba una interfaz gráfica fue el programa Sketchpad, creado por Ivan Sutherland en 1963 como parte de su tesis doctoral en el MIT. Sketchpad es considerado por algunos autores como el padre de los gráficos asistidos por computadora. Con él era posible, por medio de un sistema fotoeléctrico, que una pluma fotosensible, dibujara formas simples en un monitor gracias a la sincronización eléctrica de ambos. Se mostró con ello, un nuevo método de interacción humano-computadora; además abrió la posibilidad de que los gráficos por computadora fueran utilizados con fines tanto artísticos como técnicos.[13]

Sin embargo, no fue sino hasta los 80's cuando la conjunción de tres tecnologías dieron origen al diseño gráfico digital como lo conocemos actualmente y cambiaron radicalmente el modo en que la producción gráfica se generaba hasta ese momento.

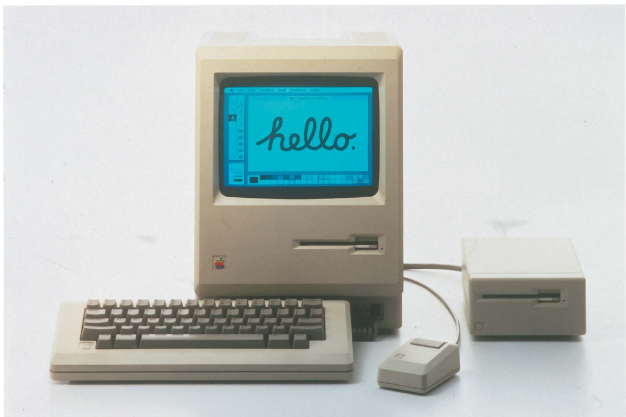
Por un lado, el nacimiento del lenguaje Post Script iniciado en 1976 por John Gaffney, continuado por él y Martin Newell en XEROX y finalmente, diseñado en su forma actual por John Warnock que, junto con Chuck Geschke, fundaron en 1982 Adobe Systems Incorporated, también conocido como ADOBE.Post Script es un lenguaje de descripción de página o PDL

(Page Description Lenguaje) que permite producir contenido de página de alta calidad y visualmente rico incluyendo texto, fotos y arte lineal, así como la posibilidad de imprimirlo. John Warnock y Clarles Geschke decidieron implementar en 1984 este "lenguaje de descripción de páginas suficientemente riguroso como para imprimir tipografía, ilustraciones y fotografías"[14]

A partir de este lenguaje, los gráficos creados por computadora pudieron imprimirse, ya que es independiente del dispositivo de salida y se puede enviar el mismo contenido a múltiples aparatos. Este hecho, con el paso del tiempo permitió que cualquier persona pudiera hacer creaciones gráficas digitales desde la comodidad de su computadora de escritorio. Sin embargo, para que este lenguaje se volviera del uso común y posteriormente en un estándar en la industria, se requirieron de otras dos tecnologías con las que el Post Script ingresó casi de la mano.

La primera de ellas y con la que ADOBE hace su primera alianza es con un equipo de computo que revolucionó e influenció de manera importante el entorno del diseño gráfico digital.

En el año de 1984, Steve Jobs, el entonces joven director de APPLE, presenta con bombo y platillo a la Macintosh. Esto representa la introducción en el mercado de la primera computadora personal que cuenta con una interfaz gráfica que permite una comunicación amigable entre usuario y máquina. Con su lanzamiento, Steve Jobs, pacta con ADOBE la inclusión



Macintosh Clasics, 1984

del lenguaje Post Script en sus equipos y la vuelve compatible con el primer programa de edición que utiliza este lenguaje, el Page Maker. Dichas máquinas se vendieron a casas de diseño con la novedad de que era posible la edición y publicación de gráficos desde un escritorio.

De este hecho histórico cabe destacar el papel tan importante de la estrategia de marketing de APPLE y el apoyo que tuvo de los medios de comunicación. Aún ahora, en Internet está disponible el comercial[16] que fue mostrado en el medio tiempo del Super Bowl del mismo año, anunciando el inicio de una nueva era. Además de que la presentación de Jobs causó revuelo; no sólo entre los asistentes de dicha ceremonia, sino en el mundo de los programadores e informáticos; ya que nunca antes se había causado tanta expectativa para un equipo de computo, al menos no en el plano de la comunicación de masas. Apple comenzó a vender no sólo equipos de cómputo, sino la idea de innovación y sofisticación al poseer una MAC. El marketing convirtió a esta máquina en la computadora generadora de gráficos por excelencia, generalizando su uso entre los diseñadores.

Esta estrategia ha sido seguida por APPLE hasta nuestros días, como lo menciona Delia Rodriguez, columnista del blog de "El País" en un artículo escrito en marzo del 2011: "...es surrealista hacer un seguimiento en vivo del lanzamiento de un producto de una empresa privada cuando no se hace con ninguna otra."[17] Y es que desde sus inicios, según en las palabra del que fue alguna vez su ejecutivo de marketing, APPLE no vende "productos" sino un estilo de vida: "...te esta invitando a experimentar el estilo de vida APPLE y ser parte de la comunidad iPod" ..."Los auriculares del iPod, por ejemplo, son un puro truco de marketing diseñados para que la única parte visible del producto sea un símbolo de estatus: usar blanco"[18]

"Los auriculares del iPod, son un truco de marketing diseñados para que la única parte visible del producto sea un símbolo de estatus: usar blanco"

Sin embargo, la MAC no fue la única influencia de Jobs en el tema de la gráfica por computadora. En el mismo año que aparece el lenguaje Post Script, el presidente de APPLE Computers convence a sus creadores de las ventajas de vender el permiso para usar su tecnología a otros fabricantes para que las incorporaran en sus equipos de impresión[19], incluyendo por supuesto, la Laser Writter de APPLE, la primera impresora en incorporar este lenguaje.

En 1985, ADOBE licencia el Post Script a fabricantes de impresoras, fotocomponedores y a procesadores de imágenes rasterizadas, así como algunas fuentes tipográficas de Linotype y hace convenio con las Linotronica, las fotocomponedoras gráficas más populares de la industria en ese momento. Comienza con ello, el problema de las licencias de uso en la creación digital.

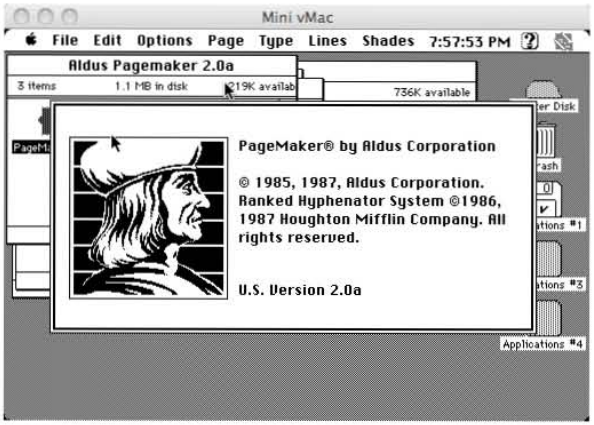


Foto de pantalla de del programa Page Maker, versión 2.0, año 1985

El tercer jugador en entrar a la escena del diseño gráfico digital, es el Page Maker, software que hacía posible el armado de páginas de periódicos y revistas desde la computadora. Fue de los primeros programas de computadora que utilizaba el lenguaje Post Script licenciado por ADOBE. El responsable de la creación de este programa fue la compañia Aldus Corporation.

"En 1984, un ex director de periódico de 36 años, Paul Brainerd formó una compañía llamada Aldus para crear un software que permitiese a los periódicos hacer mejor sus anuncios. En julio de 1985, Aldus presentó el programa Page Maker para ordenadores Macintosh"[20]. Es a su director, Paul Brainerd a quien se le debe la acuñación del término Desktop Publishing, publicación de escritorio o autoedición, palabra que se utiliza hoy en día para referirnos a la edición y maquetación de publicaciones desde una computadora de escritorio.

Así pues, con la conjunción de estas tres tecnologías: lenguaje, equipo y software, comienza la era de la "publicación de escritorio", la "autoedición" o el Desktop Publishing revolucionando así la manera en la que se generaban gráficos digitalmente y se daba inicio al diseño gráfico por computadora. A partir de ahora, los diseñadores gráficos podían ser parte de todo el proceso de producción de los objetos gráficos: "El desarrollo tecnológico durante los 70's y los 80's dio el control absoluto al diseñador de la producción de los objetos gráficos, permitiendo manipular tamaño y posición de los elementos visuales justo antes de su publicación,

Poco a poco cualquier persona que pudiera adquirir un equipo de computo, los cuales, comenzaban a bajar sus costos, podían hacer creaciones digitales, ya que "la tecnología digital también permitió la entrada en este campo de personas sin formación o con una formación mínima"[22]

Desde entonces, comenzaron a crearse otros programas que con el paso del tiempo se fueron convirtiendo en las herramientas más populares de creación gráfica digital. Es así como en 1988 se lanza oficialmente al mercado la primera versión de Illustrator, programa que corrió sólo en el sistema operativo de la MAC (OSX) y fue de los primeros capaces en crear gráficos vectoriales. Hasta el día de hoy, ADOBE tiene la licencia y derechos de este software que se comercializa exitosamente. En 1989 salió al mercado la versión para el sistema operativo Windows. Hasta el 2010, Illustrator ha tenido 20 versiones diferentes, aproximadamente una por año.

Es hasta 1990 cuando aparece la primera versión de Photoshop, programa que trabaja a base de imágenes de mapa de bits. De igual forma, este software sólo se lanzó para el sistema operativo de la MAC. Fue en 1992 con su versión 2.5, que apareció para su uso en sistema operativo Windows. Hasta el día de hoy cuenta con 19 versiones diferentes[23], todas requieren una licencia para su uso y distribución legal.

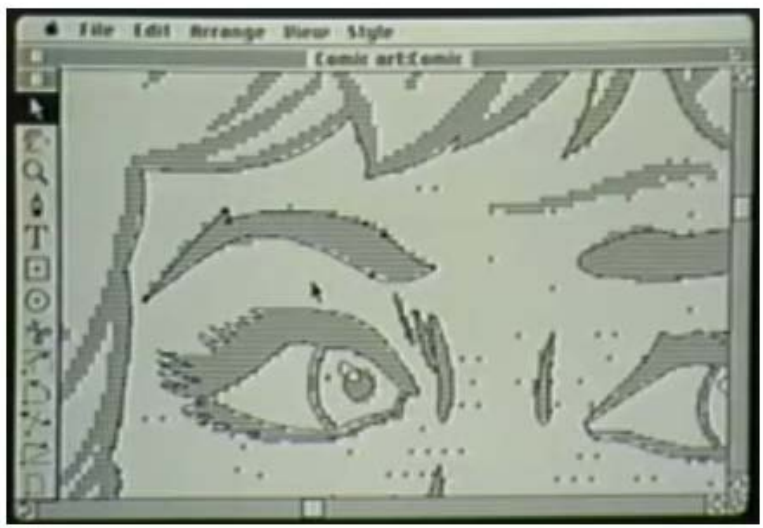


Foto de pantalla de de la primer versión del programa Illustrator, 1988

El diseño gráfico se ha visto afectado de forma irrevocable por el hardware y el software de la informática digital

Los equipos Macintosh han sido relacionadas con el diseño gráfico y los profesionales que lo ejercen, del mismo modo que Illustrator y Photoshop son programas muy populares y muy usados en este y otros terrenos. Se han convertido en las herramientas por antonomasia del diseño gráfico digital pues desde sus orígenes, las estrategias de marketing fueron dirigidos hacia este mercado y gracias a su desarrollo han resultado ser equipos y programas muy poderosos para la creación gráfica digital.

1.3. Fluser, la caja negra y una reflexión al software como herramienta.

Tal como lo dice Philip Meggs: "El diseño gráfico se ha visto afectado de forma irrevocable por el hardware y el software de la informática digital"[24]. Algunos textos afirman que el diseño depende completamente de su relación con la tecnología para poder actuar, ya que en función del avance tecnológico dispone de nuevos espacios:

"La relación entre el diseño y la tecnología ha estado presente desde siempre, cada época está caracterizada por los avances que se tenían en ese tiempo y como el diseño gráfico, adoptó los recursos por generar nuevas alternativas visuales"[25]

Sin embargo, la relación creación gráfica-software podría tener implicaciones mucho más allá de una evidente influencia o incluso, dependencia. Vale la pena reflexionar acerca del condicionamiento que ha tenido la inclusión de la herramienta digital en la creación gráfica por computadora.

Para ello, retomaré primero a Flusser y su concepción de las imágenes técnicas emergidas de los "aparatos". En su texto Hacia una filosofía de la fotografía menciona que la imagen técnica es producida por un aparato, a la vez que este último es producto de los textos científicos, así mismo, los objetos extraídos de las herramientas o aparatos adquieren una forma anti-natural, improbable y se convierten en objetos culturales.[26]Así pues, podríamos entender a la computadora y más específicamente, a los programas que hacen que generemos imágenes técnicas a partir de ella, como un aparato, mientras que los diseños, arte o creaciones digitales que se generen usándola como herramienta se convierten entonces automáticamente en objetos culturales.

La cámara fotográfica, vista por Flusser como un aparato de invención de imágenes técnicas

Ahora bien, Flusser al inicio del libro habla de dos momentos en la historia, que bien podrían definir eras de la creación de objetos culturales, por un lado, habla de la "invención de la cultura lineal" y por otro lado de la "invención de la imágenes técnicas" ésta última con la aparición de los aparatos y más específicamente con la Revolución Industrial.

Dentro de la era de la invención de las imágenes técnicas, se refiere muy en específico a la creación de imágenes técnicas a partir de un aparato fotográfico, es decir, a la cámara fotográfica, la cual, usaremos en este caso en comparación con la computadora para explicar algunos puntos de similitud.

Una de las problemáticas que Fluser menciona de la cámara fotográfica, radica en la noción de libertad ya que para Flusser, esto representa una contradicción entre el propio aparato ya que en la misma medida en la que proporciona nuevas maneras de crear, también, las limita a sus capacidades técnicas y su programa[27], lo cual define lo que puede y no puede hacer, como diría Joan Costa en la introducción de este libro: "se trataría de obligar al aparato a hacer lo que él no quiere o no puede hacer porque no está inscrito en su programa"[28].

Refiriéndose al fotógrafo frente a una cámara, que se vuelve un "funcionario"[29], éste, manipula la cámara, aparentemente a su antojo, a fin de atrapar una imagen, sin embargo, la elección del fotógrafo está restringida por el propio aparato. La cámara hace lo que el fotógrafo quiere que haga y el fotógrafo hace aquello para lo que la cámara fue programada:

"Parece como si el fotógrafo fuera libre de escoger y como si la cámara hiciera precisamente lo que él quiere que haga. Sin embargo, la elección del fotógrafo está restringida por las categorías[30]de la cámara; su misma libertad está programada. La cámara funciona según las intenciones del fotógrafo, pero estas intenciones funcionan de acuerdo con el programa de la cámara"[31]

Algo similar ocurre con las computadoras y sus programas, así como se han vuelto una herramienta esencial para el diseñador, que ha logrado disminuir el tiempo y costos de producción y generado nuevas alternativas y posibilidades creativas, también el propio creador se ha visto limitado por las capacidades técnicas que le ofrecen, algunos, incluso no sólo se han visto limitados, sino se han vuelto esclavos de la innovación tecnológica o viven a expensas de los trucos efectistas que ofrecen los programas mientras ocultan un importante vacío conceptual[32], tema que tocaremos más adelante.

Hablamos entonces de un creador digital que así como es beneficiado con nuevos posibles gracias al aparato, también es determinado por el mismo en cuanto al modo de imaginar y de realizar sus creaciones, el propio objeto cultural queda a expensas del aparato digital, en este caso, del software.

Pero esta determinación, no sólo se da por las limitaciones que ofrece el aparato, sino por el factor que lo convierte en caja negra, es decir, por el desconocimiento que existe del interior y del funcionamiento del aparato.

Expliquemos mejor este punto:

Según Flusser existe un factor entre la imagen y su significado que parece interrumpir la cadena entre esta relación.

Este factor es la caja negra. De hecho, el proceso codificador de las imágenes técnicas ocurre dentro de esta caja negra. Dicha caja que representa al aparato es para Flusser un misterio para el funcionario, ya que por más que conozca cómo alimentar la caja (conociendo su entrada) y saber cómo originar la fotografía (conociendo su salida), la cámara hace lo que el fotógrafo quiere que haga, aunque el fotógrafo no sabe lo que sucede en el interior de una caja negra[33]Esto resulta ser una característica primordial del aparato: "El funcionario domina el aparato mediante el control de su exterior (entrada y salida), y él, dominado a su vez por la opacidad de su interior".[34]Para Flusser, la cámara es una herramienta que oculta la intención de producir fotografías[35].

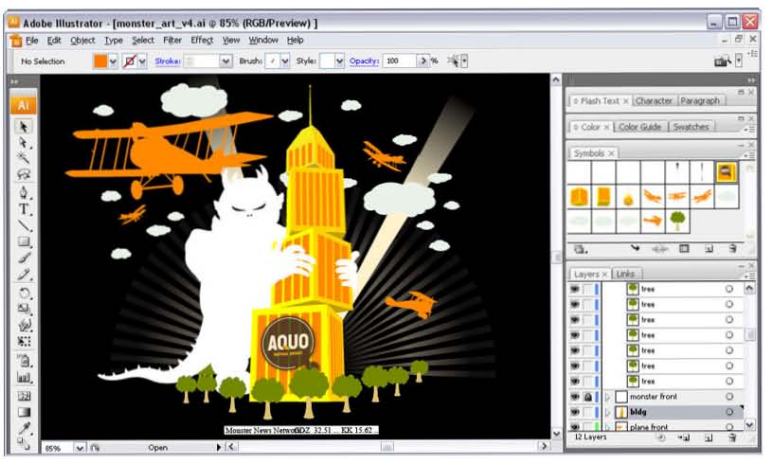
Para Arlindo Machado, en el caso de la computadora, hay siempre dos modalidades de caja negra, tal como lo explica en su artículo Repensando a Flusser en el cual explica:

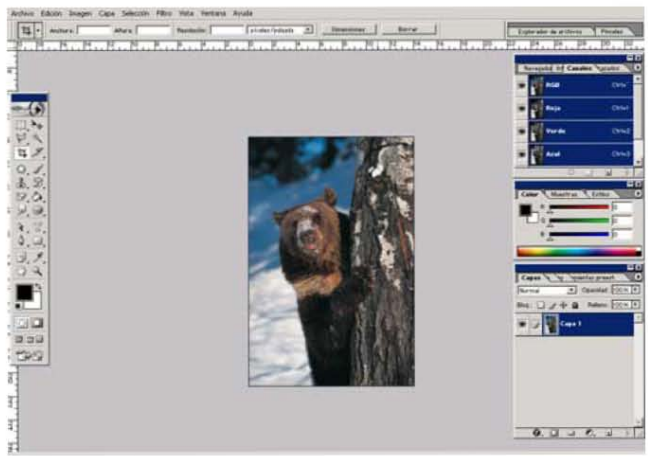
"...(hay dos cajas negras), una sólida, hard, cuyo programa de funcionamiento ya está inscrito en sus propios elementos materiales y la "inmaterial", soft, que se refiere al conjunto de instrucciones formales presentadas de modo general en lenguaje matemático de alto nivel, destinadas a determinar cómo la computadora y sus periféricos van a operar"[36]

La caja negra "inmaterial" que menciona Machado, el soft, se refiere al software, pero, más allá de referirse a todos los programas que determinan cómo va a operar una computadora, el concepto de "caja negra" coincide con los programas cuyas patentes, impiden ver su código fuente y cuyas "recetas" representan un misterio para el usuario, aún para el que conozca "el lenguaje matemático de alto nivel" con el que pudo haberse escrito, y para el cual se necesita de una licencia que permita su uso

Los programas de patente, como el Illustrator o el Photoshop, de los que hablamos anteriormente, así como el propio sistema operativo bajo el que funciona (ya sea Windows u OSX) se convierten entonces en una caja negra, ya que cualquier usuario desconoce su funcionamiento en el interior, los utilizamos, realizamos diseños con ellos pero no sabemos a ciencia cierta cómo es que lo hace, y, por lo tanto, al igual que lo es la cámara para la imagen técnica, este tipo de software se convierte en una herramienta que oculta la intención de producir imágenes digitales, que a su vez representan una solución visual para el diseñador gráfico, que a su vez resuelve problemas de comunicación de un cliente.[37]

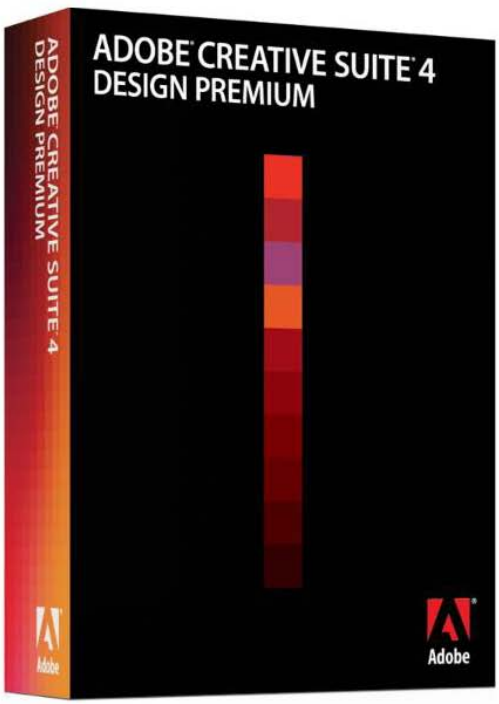
Captura de pantalla de los programas Illustrator (izquierda) y Photoshop (derecha). Software que requiere de una licencia para uso legal





En el caso de los artistas visuales digitales o los artistas electrónicos, esto puede representar el ocultamiento a su propio proceso de producción en cuanto al uso del programa se refiere, no saber cómo fue realizada la obra por la imposibilidad de acceder al código del software con el que fue hecho.

Esto representa, no sólo una limitante proporcionada por el aparato a la hora de crear, ya que eventualmente las opciones del software se irán acabando, sino que también nos enfrenta al dilema de no conocer a fondo esa herramienta que a últimas fechas resulta esencial para el trabajo de cualquier diseñador y creador visual digital. Además, si estamos en el entendido que las "herramientas digitales se han diseñado para emular el comportamiento y propiedades de los instrumentos y materiales existentes"[38], ¿no debería el propio funcionario de esa herramienta conocer el aparato con el que genera sus objetos culturales? Así como el pintor conoce su pincel, como el escultor conoce su arcilla o el fotógrafo intenta conocer el funcionamiento de su cámara ¿no debería el diseñador y el artista visual digital conocer el interior del software con el que trabaja y con ello, ser capaz de conocer conceptos y no técnicas que son sólo posibles dentro de ese tipo de software y usar así otro programa que efectúe las mismas funciones si así lo requiere?



Adobe Creative Suite, la suite de programas para diseño de Adobe, la cual, podría representar una caja negra

Pondré un ejemplo. Muchas de las funcionalidades de Photoshop ocultan la manera en la que se llega a cierto resultado, como lo hace la opción "guardar para web" en el Menú "archivo". Cuando "guardamos para web" el programa automatiza el optimizado de una imagen bajo la consigna de facilitar y agilizar las operaciones y convertirla en un archivo listo para subirlo a internet, pero es cierto que se podría hacer de otro modo, (cambiando la resolución, el tamaño en pixeles de la imagen y en general, optimizándola conociendo su composición). Al utilizar y enseñar estas opciones preconcebidas, se corre el riesgo que el profesional no

conozca a fondo, no sólo su herramienta, sino que desconozca conceptos básicos en miras de resolverlo todo "más rápido".

El esclarecimiento del funcionamiento de los aparatos, según Flusser es esencial para no permanecer ignorantes a estas imágenes técnicas:

"...Y toda la crítica de las imágenes técnicas debe concurrir al "esclarecimiento" del interior de esa caja negra. Mientras la crítica fracase en esto, permaneceremos ignorantes en lo que respecta a las imágenes técnicas"[39]

Además de que las imágenes técnicas poseen una fascinación mágica que rige en muchos sentidos el modo en la que experimentamos, conocemos y evaluamos al mundo, por lo tanto, "es sumamente importante preguntar qué tipo de magia esta implicado aquí"[40]. Lo mismo sucede con las imágenes digitales pues eventualmente, las ciencias de la cultura buscan la intención humana escondida en los objetos, se preguntan ¿por qué? y ¿para qué?[41]. Quizá sea el momento de preguntarse de dónde se originan esas imágenes digitales, cómo y de donde surge esa magia; y así como el fotógrafo de desliza hasta el interior de la cámara a fin de descubrir los trucos allí escondidos, así mismo podría hacerlo el diseñador y/o artista visual digital que desconoce el funcionamiento de su herramienta, de su software, de su caja negra, pues desconoce el código con el que esta hecho.

Para Flusser, lo que se debe de estar preguntando a la hora de enfrentarnos con cualquier fotografía es ¿qué tanto ha logrado el fotógrafo someter el programa de la cámara a sus propias intenciones y mediante qué métodos?. De acuerdo con estos criterios, las "mejores" fotografías son aquellas en las que el aparato ha sido sometido a la intención humana"[42]

Nuevamente, estas preguntas podrían hacerse también con las imágenes digitales y el software, ¿qué tanto el diseñador y/o artista digital logra someter al software a sus propias intenciones y mediante qué métodos? En el entendido que es él mismo que no conoce el funcionamiento de la caja negra[43]. En todo caso, en el afán de descubrir más y más "trucos" dentro de la cámara, y por lo tanto también, dentro del software, sería necesario descubrir las virtualidades escondidas en el programa que a su vez, le permitan descubrir nueva información, para ello tendría que entrar a la caja negra, ahondar en su funcionamiento, conocerla y desmenuzarla:

¿Qué tanto el diseñador y/o artista digital logra someter al software a sus propias intenciones y mediante qué métodos?

La gente que usa software de patente aprende sólo a usar un programa, no cómo funciona y por lo tanto no aprende a solucionar sus problemas, espera a que lo haga la máquina

"Lo que hace un verdadero creador, en vez de someterse simplemente a un cierto número de potencialidades impuestas por el aparato técnico, es invertir continuamente la función de la máquina que él usa, es manejarla en el sentido contrario de su productividad programada"[44]

En este momento, se podrá objetar que no necesariamente la intención del diseñador es la de modificar su herramienta de trabajo e incluso, que no es necesario conocer cabalmente el software que utilizada, debido a que esto no interfiere de manera sustancial en el resultado de la obra, la imagen que genera o la creación que surge de su proceso. Sin embargo, si conocer la herramienta no implica modificarla, si representa un conocimiento más profundo de la misma y por lo tanto un control más sustancial. Dicho conocimiento implica además, una reflexión acerca de la herramienta que se utiliza y sobre todo, una desmitificación, una posibilidad intencionada, una puerta hacia otros posibles.

Como ejemplo, en entrevista, Christian Oyarzún en su calidad de artista electrónico, menciona que le interesa mucho ser un desarrollador de su propio programa y efectivamente programar, para ser mucho más que un usuario y tener mucho más control sobre la obra que realiza. Por otro lado, el profesor y artista chileno José Santorcuato, piensa que la gente que usa software de patente aprende sólo a usar un programa, no cómo funciona y por lo tanto no aprende a solucionar sus problemas, espera a que lo haga la máquina:[45]

Si bien, el conocimiento de un programa no implica necesariamente la modificación del mismo, si lo es el saber que hay otras formas de resolver ciertos problemas, si ese software no realiza lo que necesito, siempre existirá un plan B. El eventual esclarecimiento de la caja negra con la que trabajamos permitiría entre otras cosas la desmitificación de las herramientas digitales que son vistas como hacedoras de todo, las que lo hacen más eficientemente y más rápido.

Otras consideraciones de la "caja negra"

Hemos dicho ya como el uso del software resulta un factor que se pone en medio del creador y de su obra, y actúa, como una especie de caja negra que es manipulada pero no conocida a profundidad por el funcionario. Esto podría representar de igual forma, una especie de vació entre la creación y el artista o diseñador, un vacío que se refiere al uso de una com-

putadora y los programas que la hacen funcionar concebidos como "hacedores de todo" e "intocables", pero sobre todo, lejanos, software que es creado y provisto por una compañía en particular que además, resulta ajena e inalcanzable.

Esta lejanía, la del aparato concebido como mito[46] y herramienta imprescindible para el diseñador y el artista digital, trae consigo también el riesgo de similitud en el resultado, de nivelación creativa por el uso indiscriminado de programas, trucos y efectos digitales:

"La posibilidad de que con el uso de tan capaz herramienta todo comience a parecer lo mismo, de que se erija el imperio de la crisis de identidad, debería ser el más serio de los aldabonazos para recordarnos que es preciso comenzar con buenas ideas, previas al uso de cualquier utensilio"[47]

La crisis de identidad y la posibilidad de que todo se vuelva más de lo mismo, en gran parte se debe también al uso masivo de la tecnología, en donde cualquier persona con posibilidades y que lo desee, puede tener acceso a un programa de diseño pues no requiere saber el contenido de la caja negra, sino solamente comenzar a usarlo, volverse diestro de la técnica y dominar la interfaz:

"Los programas de computación comerciales, operables a través de menús e interfaces estandarizadas, por lo general traen de fábrica numerosas estrategias que permiten aplicar con un clic, nuevos filtros a imágenes, componer o ejecutar sonidos.

Esto ha permitido el surgimiento de nuevos creadores amateurs que no están interesados en desarrollar habilidades durante años, sino que requieren de herramientas para dar cauce a su ímpetu creativo. Sin embargo, este tipo de software comercial empaquetado, restringe los usos creativos solo a los que la herramienta nos permite"[48]tal como lo explicaremos a continuación.

Las limitantes que muchas veces ofrece el software no sólo restringe a los creadores a trabajar con las posibilidades del programa en turno si no que pueden incluso determinar el resultado y la estética de su trabajo. Ya lo dice John Maeda: "Las herramientas del diseño impulsadas por el Macintosh están explícitamente programadas para expresar un conjunto finito de estilos expresivos, de ahí que implícitamente guíen el trabajo de diseño realizado con ellas hacia ejes estilísticos definidos de manera precisa", en pocas palabras, todo se ve

igual en el diseño digital. Esto, incrementa la posibilidad de que se produzca una uniformización de las manifestaciones gráficas[49].

Lila Pagola explica un ejemplo de lo descrito anteriormente: "Hasta hace poco tiempo, la manera en la que se realizaban algunas páginas dinámicas en la web estaban dominados por la hegemonía de un sólo tipo de software de animación, lo cual, invariablemente definía la estética "dinámica y vistosa" de las páginas realizadas con este programa en específico, incluso en cada una de ellas era posible ver las deficiencias y alcances de dicho software, convirtiéndose en un despliegue de efectos que en ocasiones hacían la página tan atractiva, como pesada, lenta y poco funcional. En el auge de esta forma de hacer sitios en línea, era difícil optar por herramientas alternativas como la programación en javascript o html, pues resultaban menos accesibles para diseñadores, no se enseñaban en las escuelas de arte y diseño y mucho menos, tenían la misma promoción y protección que Macromedia Flash y posteriormente Flash de ADOBE cuando se definieron los "estándares" (de facto) para la animación web". [50]

Machado también menciona este tema un par de ocasiones:"La repetición indiscriminada conduce inevitablemente a la estereotipa, o sea a la homogeneidad y previsibilidad de los resultados... la multiplicación de modelos prefabricados en nuestro alrededor, generalizados por el software comercial, conduce a una impresionante padronización de las soluciones, a una uniformidad generalizada, o entonces a una absoluta impersonalidad"[51] Para lo cual, posteriormente propone: "para evitar la repetición y el cliché, las máquinas y los procesos tecnológicos necesitan ser continuamente reinventados o subvertidos"[52] En este mismo sentido, existen varias posturas respecto a que el uso de software ha generado una especie de adecuación por parte del artista o diseñador a las funcionalidades adscritas al software usado, lo que genera una marcada dependencia al programa en lugar de la libertad de poder crear con cualquier otra: "Crear con las funcionalidades adscritas al software diseñado frente a la homogeneización y la normalización en el diseño gráfico, lo cual sucedió a raíz de la excesiva dependencia y confianza en ciertos programas como el omnipresente Photoshop"[53]

"Crear con las funcionalidades adscritas al software diseñado frente a la homogeneización y la normalización en el diseño gráfico lo cual sucedió a raíz de la excesiva dependencia y confianza en ciertos programas como el omnipresente Photoshop"

Así pues, el software de patente se convierte en un producto para las masas, ofreciéndose precisamente en escala masiva, logrando confundirse, incluso con la propia computadora, (que todas las PC de escritorio tengan instalado de fábrica determinado sistema operativo sin darle cabida a la libertad que el usuario tiene de elegir el SO[54]que más le convenga). Los programas destinados a la creación, siguen también esta misma lógica, entendiendo el uso de Photoshop como el único programa pa-

ra la digitalización y retoque de imágenes, basta con pensar en nosotros mismos diciendo que hay que "Photoshopear" una imagen cuando nos referimos a retocarla digitalmente, dando por hecho que Photoshop es el único programa que realiza esta tarea, convirtiendo al programa en la propia acción de "manipular imágenes" Esto, tal como lo dice Smiers podría no estimular la creatividad:

"Para poder desear otra cosa se necesita una imaginación exuberante y la convicción de que la vida cultural puede ofrecer mas de lo que normalmente y en apariencia de modo inevitable, se ofrece a escala masiva. El contenido de los productos producidos y distribuidos en masa, sin embargo, por lo general, no estimula la imaginación ni alimenta esa convicción"[55]

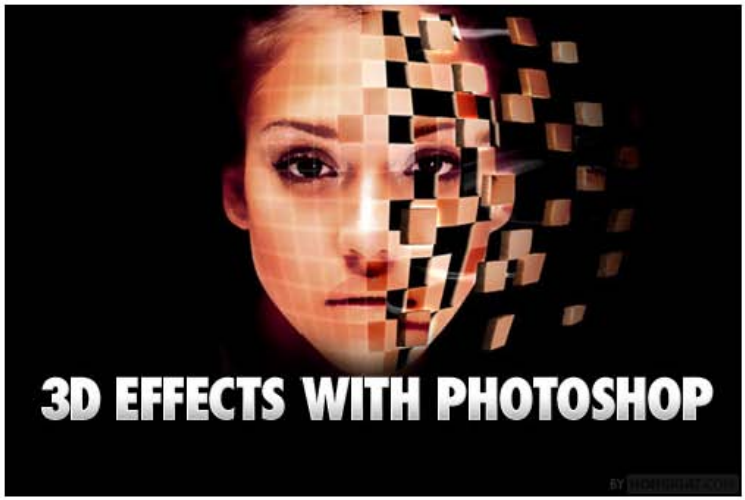
El software comienza a tener una connotación mucho más profunda en el terreno de la creación, pues se convierte en una caja negra que podría determinar la estética de la obra y convertir al funcionario (que en este caso sería el diseñador y/o el artista) en sólo un operador y alejarlo de la obra en sí misma, y por lo tanto de su carácter como creador.

El papel de operador en lugar de creador y la idea de simplificar las operaciones para que sean sencillas y puedan ser usadas por cualquier persona, sin duda, se han visto incrementadas por la masificación, como ya hemos dicho, de las herramientas digitales.

Justamente, menciona Flusser, que ésta es la característica de toda función post-industrial: "las categorías de los aparatos se imponen a las condiciones culturales, filtrándolas durante el proceso. El resultado es una cultura de masas uniforme producida por los aparatos"[56]

Meggs también habla de esto, y compara la introducción del software de autoedición con la invención de la cámara Kodak por George Eastman: "Del mismo modo que en la década de 1880 la fotografía dejó de ser de uso exclusivo de los especialistas y se puso al alcance del público en general, en la década de

"Lo que un toque de Photoshop puede arreglar". Haciendo alusión las soluciones casi "mágicas" que el programa puede hacer por una imagen



"Efectos 3D con Photoshop". Banner web que anuncia tutoriales para replicar efectos usando este programa de edición de imágenes.

1980 la tipografía se alejó del dominio exclusivo de los profesionales y se hizo accesible para una esfera más amplia de la población"[57]

Al hacerse accesible a una esfera más amplia de la población, el mercado comenzó a demandar profesionales del diseño gráfico digital, además de que comenzaba a ser necesario realizar conceptos teóricos que funcionaran con el nuevo medio virtual, en este sentido, "fue imperante la enseñanza de un software especial que requería a su vez hardware cada vez más potente y con gran capacidad de memoria para realizar proyectos"[58]

Es por eso que han surgido innumerables centros de formación donde aprender en un tiempo record las técnicas necesarias que se supone demanda el mercado. Se imparten cursos para enseñar diseño, pero lo único que consiguen es adiestrar a sus alumnos como operadores de MAC OSX o Windows PC[59]y ha hecho que el número de "nuevos operadores" se haya multiplicado y que "cualquier persona con un MAC sea para el cliente un diseñador y desde luego, un competidor para el resto de los profesionales"[60]

Pero la herramienta digital no sólo es enseñada en centros de capacitación técnica, sino también en la academia. Carreras como Diseño y Comunicación Visual en la Escuela Nacional de Artes Plásticas de la UNAM en México D.F., incluye entre sus asignaturas temas como Introducción a la Tecnología Digital donde se enseña el uso de un determinado tipo de software, más específicamente, los mas demandados por el mercado y que resultan ser programas de patente, como lo es Photoshop, Illustrator y Flash, por mencionar algunos.

Esto trae consigo que no solo se les enseñe a los alumnos a usar la caja negra, sino que se garantiza la herencia a futuras generaciones de creadores digitales, perpetuando así los planteamientos que se han hecho hasta ahora, (el desconocimiento de la herramienta, el posible vacío de la obra, la determinación de su estética, su papel como operador, la masificación, la mitificación del aparato, pero también la inmediatez, la facilidad en el proceso, el

1.4. Software y creación gráfica digital

Entonces, ¿cuáles son las herramientas digitales que utilizamos (los diseñadores, artistas y creadores gráficos digitales) para resolver un diseño o un problema de comunicación visual y por qué optamos por dicha herramienta?

La respuesta, según mi experiencia como profesional del ramo, es que se utilizan programas como Photoshop para imágenes de bits, Ilustrator para vectores, Flash para gráficos en movimiento, Dreamweaver para páginas web, 3d Max o Maya para animación, etcétera; los elegimos por que son los que aprendimos a utilizar en nuestra formación universitaria y por que son por costumbre, los programas de diseño usado por años, debido en gran medida, a su fuerte estrategia de marketing (como ya lo mencionamos en el caso de APPLE), también por que en definitiva son los que tienen mayor desarrollo en el mercado, con mas de 20 versiones para Ilustrator y 19 para Photoshop y poseen una amplia gama de posibilidades y herramientas para llevar a cabo un diseño digital, pero además, las usamos por que desconocemos que existan otro tipo de herramientas que realicen las mismas funciones.

Sin embargo, una de las primeras problemáticas con las que nos encontramos al adquirir este tipo de software es que, como sabemos, para ser instalados en una computadora de uso personal o empresarial, se requiere una licencia que permita su uso legal, licencia que por lo regular cuesta alrededor de U$1 899 dólares[61] y que una de sus clausulas restringe la compartición con otros usuarios, es decir, que el comprador de alguna licencia de software no podrá instalarlo en más de un sólo equipo.

Ahora bien, imaginemos que como diseñadores recién egresados no nos hemos capitalizado lo suficiente para tener más 23 000 pesos e invertirlo en una licencia de los programas que más utilizamos, o que somos parte de una asociación civil o una ONG sin recursos suficientes. En el contexto económico de México, es muy probable, que en casos como estos, se recurra a la versión no autorizada del programa en cuestión, pues las necesidades de producción deben de ser cumplidas de manera casi inmediata.

Las copias no autorizadas de la suite de ADOBE, por ejemplo, no son más que una copia sin una licencia legal de la empresa, lo cual significa que no esta distribuida bajo el consenti-

miento del titular de los derechos del autor y que no permite utilizar esta copia del software para los fines que el usuario desee. Es por eso que en caso de que se quiera realizar una obra en este tipo de software y quiera hacerse una distribución apegada a derecho, es probable que tengamos que adquirir una copia legal.

Como diría Lessig en Free Culture, tal pareciera, con los altos precios en los costos de las herramientas, que el proceso creativo es un proceso de pagar abogados por violar las licencias a la hora de querer internacionalizar nuestra obra, de nuevo, un privilegio o quizá una maldición reservada para unos pocos. La pregunta aquí sería: ¿Sólo los que tienen acceso a una licencia del software puede acceder a la realización legal de su obra, y por lo tanto, a su distribución por otras vías[62], ingreso a concursos[63], etc? ¿Por qué? ¿la creatividad y calidad de la obra es mejor por la licencia que se otorga? ¿Cuánta creatividad nunca se realizó sólo por que el coste de obtener los derechos (o licencias) era demasiado alto?

Siguiendo con esta idea, para no ver limitada nuestras herramientas digitales debido a su alto costo, es muy probable que generemos nuestro diseño en una copia no autorizada sin licencia del programa, acto que de primera mano sería un delito, pero que por desgracia, en un contexto como el de México la acción no resulta grave pues "todo el mundo lo hace", inclusive, la propia escuela lo fomenta al instalar copias no autorizadas de sistemas operativos y de programas en las computadoras de uso para estudiantes.

"Laurence Lessig, autor del libro "Cultura libre" cómo los grandes medios usan la tecnología y las leyes para encerrar la cultura y controlar la creatividad

Aunado a los problemas de legalidad del software, existe otra problemática que se refiere a la interoperabilidad entre la plataforma tecnológica que usan los diseñadores, esto es, cuando se poseen distintas versiones del mismo programa, que como ya dijimos, se renueva aproximadamente cada año.

La interoperabilidadse refiere a la habilidad de dos o más sistemas o componentes para intercambiar información y utilizar la información intercambiada, una definición mucho más amplia, habla de la habilidad de organizaciones y sistemas dispares para interactuar, lo cual implica compartir información y conocimiento a través de sus procesos de negocio, mediante el intercambio de datos entre sus respectivos sistemas de tecnología de la información y las comunicaciones[64]. Hay que decir, que interoperabilidad habla de un término muy arraigado a la industria, refiriéndose a organizaciones y procesos de negocio, no hay que confundirlo con el uso de estándares, el cual será explicado un poco más adelante.

Para explicar la interoperabilidad, pongamos un ejemplo: supongamos que realizamos el diseño de una revista este año con Indesign versión CS5. El archivo fue guardado en la extensión nativa del programa .indd para esa versión en particular. Si queremos pasar el archivo a otro diseñador cuya máquina tenga instalado un Indesign CS3, es muy probable que no se pueda abrir o pierda algunas características. Una alternativa sería guardar desde el principio el archivo para que pueda ser abierto en versiones anteriores, sin embargo, aún así los archivos pueden sufrir algunas modificaciones o no respetar ciertas especificaciones, todo en mira a que el diseñador tarde o temprano consiga la versión más reciente del programa, tal como sucede con los programas ofimáticos:

"Microsoft el motor real de incompatibilidades: ellos producen deliberadamente nuevas versiones incompatibles con las anteriores para obligar a los usuarios a comprar cada nueva actualización. ¿Cuantas veces han tenido los usuarios que actualizar Office porque el formato del fichero ha cambiado?"[65]

Siguiendo con la misma idea, si supeditamos la visualización de una obra a la existencia de un software, corremos el riesgo de que el fabricante deje de producir ese software y entonces, los trabajos realizados en él se pierdan o sea más complicado acceder a ellos. Imaginemos que algún día en el futuro, el programa con el que realizamos nuestra obra deje de existir y por lo tanto suextensión nativa quede sin poder abrirse. Esto provocaría que se pierda la obra parcial o totalmente, a menos de que se tuviera acceso al código para poder crear un programa parecido que lo pueda abrir o conservar una vieja versión del software desaparecido, pero esto significaría un severo límite de circulación del objeto cultural. Es aquí donde entra la importancia del uso de estándares en la creación.

Un estándarse refiere a las normas y/o acuerdos documentados que contienen especificaciones técnicas, métodos, procesos, prácticas y otros criterios precisos para ser usados consistentemente como reglas, guías o definiciones de características para

Splash page del programa Director, versión MX de macromedia, año 2002. Obtenido en http://www.guidebookgallery.org

asegurar que los procesos o servicios se ajusten a un propósito. Por el contrario, una costumbre, una convención, el producto de una empresa, estándares corporativos pero no oficiales, etcétera, cuya penetración en el mercado es inmensa y aceptada, es llamado estándar de facto. La diferencia entre un estándar y un estándar de facto es que el primero posee la virtud de pertenecer en el tiempo pues no requiere que un solo programa o sistema lo abra, mientras que el segundo queda supeditado a una sola marca, a una empresa, a la industria y/o al mercado y por lo tanto a la existencia de la misma. Algunos ejemplos de estándaresson el lenguaje HTML y el CSS para web, SVG para los archivos que contengan vectores, PDF para lenguaje Postscript, que puede tener vectores, mapas de bits y/o texto. Ejemplos de estándares de factoson DOC para documentos de texo, AI para archivos que contengan vectores, PSD para mapa de bits, FLA, SWF, WAV, INDD, entre otros.

Para entender mejor esto, pongamos un par de ejemplos:

En el tercer semestre de la licenciatura, uno de los trabajos prácticos de la clase de multimedia fue la de generar un interactivo en el programa Director, de la empresa Macromedia. Años después, Macromedia fue absorbido por ADOBE y con ello, la empresa tomó la decisión de dejar de desarrollar ciertos programas y sacarlos del mercado, entre ellos, Director y Fireworks. Después de múltiples formateos y cambios de equipo que tuve a lo largo de mi vida académica y posterior vida profesional, me fue imposible mantener conmigo una copia de Di-

rector, pero si una de mi interactivo. Sin embargo, dicho interactivo, al igual que muchos otros archivos hechos en versiones de programas pasados e incluso, desaparecidos ahora, me ha sido imposible volver a abrirlos debido a que ya no hay forma de conseguir el programa o las versiones nuevas ya no lo abren.

En este mismo sentido, Lila Pagola menciona su experiencia relacionada con esto: "Yo hice un interactivo en Director para graduarme de licenciatura en plástica y recuerdo que cuando cambiaron las versiones del programa yo me comencé a preocupar por garantizar la compatibilidad entre la versión que yo tenía y la siguiente. De pronto, dejó de ser compatible con el paso del tiempo. Por un tiempo lo abría en la versión nueva y lo guardaba, pero hubo un momento en el que ya no lo hice más. Ahora, para poder abrir ese trabajo es muy probable que tenga que conseguirme ese programa (que ya no existe) en la versión de aquellos años. Es probable que no lo vuelva a abrir nunca más"[66]

Desde el punto de vista curatorial, "la elección para sistemas y licencias abiertas[67] puede simplificar la preservación y el mantenimiento de una obra de arte, independientemente de los volubles estándares de facto y los hábitos de edición limitado heredados de los medios de comunicación no digitales"[68].

Es imposible garantizar la apertura de un archivo realizado en un software patentado pues no sabemos en qué momento la compañía deje de producir ese programa, en qué momento decida sacarlo del mercado o en cual deje de darle soporte, y por lo tanto nos atenemos a la posibilidad de que no podamos abrir el archivo en su versión editable, o al menos no de una manera sencilla.

Laurance Lessig hace una reflexión respecto a este tema en su libro Free Culture: ¿Por qué ocurre que parte de nuestra cultura que está registrada en los periódicos permanece accesible a perpetuidad mientras que la parte que esta registrada en cinta (o digitalmente) no?, ¿Cómo es que hemos creado un mundo en el que los investigadores que intentan entender el efecto de los medios de comunicación de los E. U. Del siglo XIX lo tienen más fácil que los investigadores que intentan hacer lo mismo en el siglo XXI?[69]

La elección para sistemas y licencias abiertas puede simplificar la preservación y el mantenimiento de una obra de arte

Un ejemplo más, obtenido de la entrevista a José Serralde[70], músico y consultor de tecnología digital para las artes. Me explicó uno de los problemas que tuvo en un proyecto que li-

Unificar los criterios con los impresores e integrar las herramientas de creación, se convirtió en una tarea titánica.

dereó en 1998 que consistía en la promoción cultural, divulgación y apoyo para una agrupación de música tradicional de la Mixteca. Dichos problemas fueron en el ámbito editorial, pues al tratar de integrar los trabajos de diseño, estos estaban realizados en distintos programas y diversas plataformas, por lo que unificar los criterios con los impresores e integrar las herramientas de creación, se convirtió en una tarea titánica. Dicho en sus palabras, explica:

"Carlos Veloz (el diseñador del proyecto) trabajaba en la Academia de San Carlos con plataformas distintas, primero con QuarkXPress[71]que en aquel entonces todavía se usaba mucho en la industria editorial y con el naciente entonces Indesign[72]al que yo le aposté absolutamente todo (...) Donde todo se volvió disfuncional fue en la poca posibilidad de tener QuarkXPress instalado en distintas plataformas y en distintos ámbitos de trabajo. Comenzamos a ver que las versiones nuevas de Windows, el Windows 98, Windows xp, resultaban no siempre compatibles con las versiones de QuarkXPress que él (Carlos Veloz) usaba; luego vimos además que los archivos de dicho programa no funcionaban en sus distintas versiones, había un importador de proyectos desde Indesign para QuarkXPress que no funciona del todo bien tampoco y lo siguiente, que fue mucho mas grave (...) Sucedió que yo no tuve ni el tiempo ni el dinero para poder cuidar y migrar un proyecto completo de QuarkXPress a PDF o hacer que esa versión de QuarkXPress funcionara con los Corel[73]de los impresores (...) la opción hubiera sido que nuestro software corriera en todas las plataformas de la misma manera que los documentos hubieran sido lo suficientemente portables. Pero entre 1998 y el 2002 que fue cuando sucedió todo, esto todavía no era del todo posible." Eventualmente, el uso de software de patente como herramienta para este proyecto, y sus problemas con la interoperabilidad y falta de uso de estándares, representaron un problema importante que llevaron a complicaciones mayúsculas en cuanto a producción de la obra se refiere. Su permanencia en el tiempo, también está condicionada debido a esto.

1.5. El software como instrumento, material y medio

Si el software es la herramienta que nos permite realizar nuestras obras; ya sea un diseño, una pintura digital, una obra literaria, una pieza de arte electrónico, entonces el software se convierte en nuestro nuevo y moderno pincel, lápiz o cincel. El software podría comenzar a

ser entendido como el material y un medio para la creación digital, tal como lo mencionan Aymeric Mansoux y Marloes de Valk :

"La relación entre artista, herramienta, contenido y audiencia nunca había tenido tanta importancia como la que tiene en la cultura digital. Los datos se han convertido tanto en instrumentos como en material y medio, todo para ser reproducible sin costo y distribuible instantáneamente."[74]

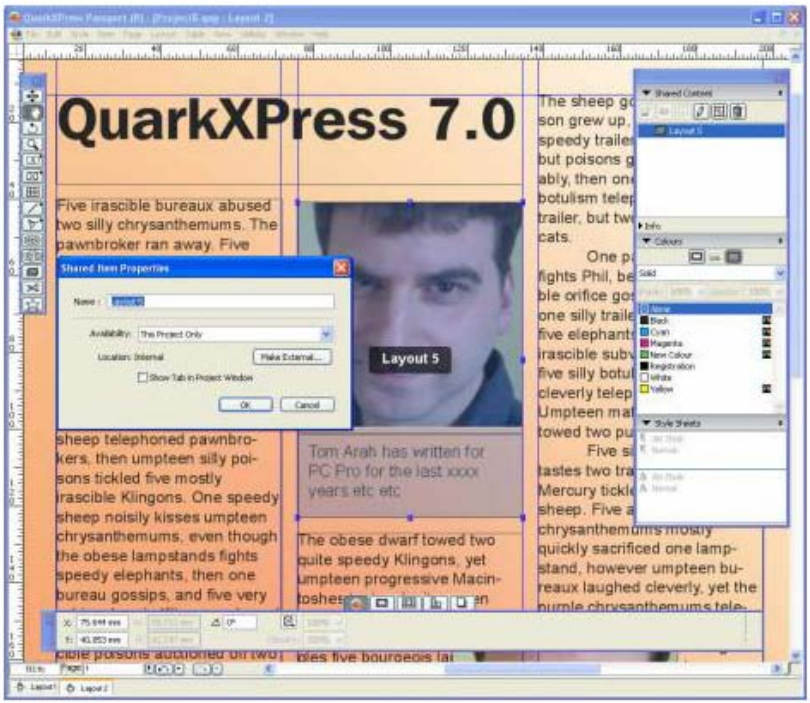
Se comienza a tener conciencia del hecho de que el software es la materia prima de una obra de arte y que la elección del material influye mucho en el trabajo eventual y su contexto.

Cuando un artista toma esta conciencia, es probable que también comience a optar por las autorizaciones abiertas[75]para la propia obra artística, lo cual va a permitir reutilizar el código con el que fue hecha dicha obra y por lo tanto, facilitará en intercambio de conocimientos. La desmitificación del software y el código es un resultado importante de ésta.

"La apertura de una obra de arte no sólo impacta sobre su lanzamiento y su relación con su audiencia, sino que también influye mucho en el proceso en el que se basa. Por eso es importante entender que el software no es sólo un componente técnico del arte digital, es la obra de arte y su código lo que proporciona otra lectura del mismo. Esta capa a menudo se olvida de la interpretación. Debe de estar abierta a todos, para estudiar y comprender el trabajo, aprender, inspirarse, comentar."[76]

Así mismo, cuando se cierra el código de un software, también de alguna manera se está limitando su potencial de futuras adaptaciones a la asfixiante monotonía de una identidad fija, pues como dice Berry Slater en su ensayo titulado Bare Code: Net art and the free software movement, los rígidos controles impuestos por las compañías de software que ocultan el código, ya sea artístico o técnico, alejan a la obra del escrutinio de posibles colaboradores y defensores y se pone en contra del fecundo caos de una invención incontrolable.[77]

Ahora bien, siguiendo con esta idea, si retomamos a Benjamin, tal y como lo hizo Berry Slater, debemos recordar que en El autor como productorse plantea que el autor revolucionario debe de ir más allá del interés limitado con el producto para efectuar la transformación del



Captura de pantalla de QuarkXpress 7.0, programa de publicación de escritorio que mantenía su código cerrado.

mecanismo de producción. Para que este mecanismo de producción se vea transformado, y el arte en tal caso, obtenga una trascendencia y que el interés vaya más allá del producto, es necesario que el artista posea un carácter ejemplar, capaz de instruir a otros en su producción y de colocar un mecanismo mejorado a su disposición, de tal suerte que consumidor y productor se vuelvan indistinguibles, que más espectadores se vuelvan colaboradores.

En este sentido, el uso del software de patente en la obra de arte o en el diseño, teniendo un código cerrado, significa eventualmente que el carácter modelo de la obra se queda sólo en un carácter funcional. ¿En qué medida se puede decir que se transforma el mecanismo de producción, si la herramienta y material con el que realizamos la obra permanece oculta, encerrada, lejos de la posibilidad de enseñanza y transparencia?¿Cómo puede el artista penetrar y conocer el interior de estas cajas negras informáticas? Una forma es justamente el conocimiento del código, el interior de dicha caja.

Predecir el futuro creando

Hemos visto hasta ahora en este recorrido, una reflexión acerca del software como materia prima, medio y herramienta para la creación visual digital, y con ella hemos visto también los problemas que enfrenta y las repercusiones dentro y fuera de la obra. Sin embargo, es necesario no dejar de lado que no se pretende satanizar a la herramienta digital en ningún sentido, más bien, hay que considerar que en algún momento del diseño gráfico por computadora y el uso del software representaba el control por parte del diseñador, eso quizá fue cierto en su momento, en los 80's cuando los costos disminuyeron y la tecnología comenzó a masificarse; pero hoy en día, podríamos encaminar las reflexiones a cuestionar el tipo de herramienta que se usa para crear y fomentar una visión crítica de ellas a las nuevas generaciones de creadores visuales digitales.

El diseñador debe convertirse en intelectualmente propietario de su herramienta, del software y así hacerse intelectualmente responsable"

Con el nacimiento de los programas de auto-edición, el diseñador parecía tener un mayor control de su creación, desde la concepción de la idea hasta la impresión del mismo, sin embargo, con el paso del tiempo, los diseñadores han usado el software a su manera, dominando las funciones básicas que hace tiempo ya están resueltas y se han visto en la necesidad de forzar el software demandando más y más funciones que no podrán llevarse acabo hasta la siguiente generación de hardware, por lo que éste suele estar por detrás de las necesidades de los usuarios.

Entonces, lo que en un principio parecía ser la panacea y la innovación, ahora, podría representar una limitante, pues las posibilidades creativas de cada creador son distintas y al final del camino, quedarán en espera de que la compañía que fabrica el software le de cambios al mismo, nuevos filtros, nuevas novedades para ser usadas hasta que sean agotadas y así continuar un círculo que parece infinito. Además que al simplificar ciertas funciones, éstas pueden tornarse lugares comunes, pues todo mundo comienza a utilizarlas por que son fáciles, accesibles y visualmente atractivas, comienzan a ponerse "de moda".

Uno de esos primeros retos a superar es el esclarecimiento de la caja negra: "El diseñador debe convertirse en intelectualmente propietario de su herramienta, del software y así hacerse intelectualmente responsable"[78]. Una de las formas en las que esto podría ser posible, es que el creador digital rompa las barreras de su propia profesión y se acerque al interior del programa que utiliza, en un intento por conocer y aclarar el contenido de esa caja negra,

incluso, aún cuando su intención no sea la de modificarlo o crear su propia herramienta sino como un mero ejercicio de reflexión, de desmitificación de la herramienta y una actitud crítica hacia ella. Ya ha pasado en otros momentos de la historia en la que muchos diseñadores aprendieron otras técnicas ajenas a las necesidades estrictas de su profesión para poder desarrollarla más y mejor. Tal es el caso del caligrafísta José de Casanova, que aprendió técnicas de grabado para poder así realizar por si mismo las planchas con las que se habrían de estampar sus libros. El resultado es una caligrafía, no sólo más pulcra, sino, sobre todo, mucho más creíble.[79]Quizá sea como propone Cerezo con el nacimiento de una nueva artesanía: "estoy proponiendo deliberadamente el acento en el contacto profesional entre el diseñador y su diseño... esto significa, control del diseño digital y por lo tanto, responsabilidad por parte de los diseñadores"[80], "parejo al control sobre el software comienza a surgir un control sobre el diseño mismo vinculado al control sobre la producción posterior"[81].

"Sólo con un real acercamiento al interior de la caja negra se podría tener un verdadero control sobre el software, y por lo tanto volverse dueño del mismo"[82].

Y es que quizá sea el momento de pensar en manos de quién esta la tecnología. Cerezo se pregunta "¿qué pasaría si APPLE, MICROSOFT y ADOBE -que controlan la gran mayoría del software que utilizamos- decidieran dejarnos en la estacada?...hasta cierto punto los diseñadores nos hemos convertido en consumidores no muy avispados de una tecnología que nos salía al paso y hemos tenido que pagar- y seguimos pagando- un alto precio por tener que ocuparnos de asuntos en los que jamás habíamos pensado"[83]

"Sólo con un real acercamiento al interior de la caja negra se podría tener un verdadero control sobre el software, y por lo tanto volverse dueño del mismo"

Quizá sea importante no perder de vista que dentro de todo, somos nosotros los que podemos formar las lentes que configuran la cultura, y predecir el futuro inventándolo[84], retomando en control de las herramientas que usamos, lo cual no quiere decir que aceptemos las limitaciones de las máquinas actuales, "sino que aceptemos las limitaciones de las máquinas en cuanto tales. Más aún, y lo que es incluso más importante, no estamos diciendo que la máquina sea el árbitro del diseño, eso siempre lo es el pensamiento".[85]

Las implicaciones que llevan a relacionar al software de patente con el de la caja negra, han hecho que un grupo de personas opten por referirse a él con el término **software privativo**, para referirse a un software que "priva" libertades y ofrece ciertas limitaciones. De igual manera han decidido crear y optar por el camino del **software libre**, el cual permite no sólo

que los usuarios conozcan las líneas de código con el que se ha hecho el programa, sino que sea copiado, distribuido y mejorado sin restricción ni problema legal alguno, convirtiéndose así en una alternativa para conocer el interior de la caja negra:

"El software libre sería una forma de acceder al interior de la "caja negra" de la computadora -su parte lógica, el software- en cuanto con el conocimiento suficiente es posible abrir fuentes, estudiarlas y hacer modificaciones, adaptaciones, traducciones y mejoras: eso habilita el código abierto de un software"[86], se muestra como opción al software privativo.

Pero del software libre y cómo puede ser aplicado en el terreno de la creación, sus alcances, repercusiones y su diferencia con el software privativo, hablaremos en el siguiente capítulo.

Citas del Capítulo Uno

[I]El término software libre y software privativo será explicado en el transcurso de esta tesis.

[1]Real Academia Española, definición de "programa informático" recuperado en diciembre 2011, desponible en http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=programa

[2]HBO, Nom de Code: Linux, Documental en línea, dirigido por Hannu Puttonen, Francia: ADR Productions ARTE, 2001).

[3]Beatriz Busaniche y Federico Heiz "Cartillas de introducción al software libre", (citada en Mayo del 2011) Proyecto Nomade ed. Lila Pagola: disponible en: http://www.nomade.org.ar/sitio/?page_id=104

[4]Busaniche "Cartillas de," 1.

[5]El código fuente es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar dicho programa. Por tanto, en el código fuente de un programa está descrito por completo su funcionamiento y esta escrito en algún lenguaje de programación, pero en este primer estado no es directamente ejecutable por la computadora, sino que debe ser traducido a otro lenguaje (el lenguaje máquina) que sí pueda ser ejecutado por el hardware de la computadora. A partir de ahora nos referiremos al código fuente sólo con la palabra "Código".

[6]Richard Stallman, Software Libre para una Comunidad Libre (Madrid: Traficantes de sueños, 2004) 19.

[7]Ricardo J. Castello, Eduardo Gauna, Sandra Arónica y otros, software Libre, moelo de análisis de factibilidad económica-financiera (Informe del proyecto de investigación del 2004, Universidad Nacional de Córdoba) Marzo 2005. 8.

[8]"Basics of the Unix Philosophy, Chapter 1, Phylosophy", Internet FAQ Archives, online education (Citada en enero del 2012) disponible en: http://www.faqs.org/docs/artu/ch01s06.html

[9]Castello, "Software Libre" 8

[10]La entrevista con Christian Oyarzún, junto con la gran mayoría de entrevistas que se encontrarán a lo largo de la tesis,(salvo en las que sea mencionado lo contrario) fueron realizadas como parte de las actividades de la estancia de investigación en la ciudad de Córdoba Argentina y en Santiago de Chile, en el segundo trimestre del 2011 cuyo objetivo consistió en recolectar fuentes primarias de artistas, diseñadores y/o creadores digitales latinoamericanos que usan software libre junto con sus experiencias y reflexiones al respecto.

11BASIC es un lenguaje de programación popular en los 80's cuyas siglas significan: Beginner's All-purpose Symbolic Instruction Code.

12Stallman "Software libre para," 21.

13Ivan Edward Sutherland, "Sketchpad: A mal-machine graphical communication system", ed. University of Cambridge Computer Laboratory (Disertación de grado, Massachusetts Institute of Technology , 1963)

14"Historia del Post Script", Digitecnia (Citada el 30 de marzo del 2011) disponible en: http://www.icono-computadoras-pc.com/postscript.html

15 "Timeline of Computer History-1983", Computer History Museum. (2006 [citado el 7 de Junio del 2009]): disponible en http://www.computerhistory.org/timeline/?year=1983

16El video disponible en youtube en: http://www.youtube.com/watch?v=M0XsFBTb5tA

17Delia Rodríguez "Por qué nos vuelve locos Steve Jobs sobre un escenario" Trending Topics (3 Marzo del 2011 [citada en Junio 2011] Blog de El País): disponible en http://blogs.elpais.com/trending-topics/2011/03/por-que-nos-vuelve-locos-steve-jobs-sobre-un-escenario.html

18Steve Chazin "The Marketing Apple, eBook" Marketing Apple ([recuperado en Junio 2011] Steve Chazin ed.) disponible en: http://www.marketingapple.com/marketing_apple/the-marketingapple-ebook.html

19José María Cerezo, Diseñadores en la Nebulosa, el diseño gráfico en la era digital (Madrid, Biblioteca nueva, 1999), 38.

20Philip Megs y Alston W. Purvis Historia del Diseño Gráfico (México, RM Verlag, 2009), 488.

21Ma. Eugenia Sánchez Ramos "La revolución digital en el diseño gráfico " Actas de Diseño, Vol. 7 Año IV (Julio 2009) 255.

22Meggs, "Historia del," 490

23 "Adobe Photoshop" Wikipedia, La enciclopedia libre (28 mayo 2009 [citado el 6 de junio del 2009]): disponible en: http://es.wikipedia.org/wiki/Adobe_Photoshop

24Meggs, "Historia del," 488

25Ma. Eugenia Sánchez Ramos “La revolución digital en el diseño gráfico ” Actas de Diseño, Vol. 7 Año IV (Julio 2009) 256.

26Vilem Flusser, Hacia una Filosofía de la fotografía (México: Trillas, 1990) 24.

27En este caso, Flusser se refiere a programa a la manera en la que esta elaborada la cámara, en sí a su diseño industríal, al interior de la cámara y la ingeniería que la hace funcionar. De ninguna manera se refiere a la definición de programa o software del que hablamos al inicio de este capítulo.

28Flusser, “Hacia una,” 6.

29Funcionario, según Flusser se refiere a que el hombre no es ni una constante ni una variable, se trata de una relación en la que el hombre y el aparato forman una unidad de función singular, por eso el fotógrafo debe llamarse funcionario de un aparato.

30Con categoría Flusser se refiere al espacio y tiempo que es determinado por la propia cámara fotográfica, un tiempo fotográfico, alejarse o acercarse al objeto. Condiciones que el aparato impone al funcionario..

31Flusser, “Hacia una,” 34.

32Raquel Pelta, Diseñar hoy (Barcelona, Paidos Diseño 01, 2004) 101.

33Flusser, “Hacia una,” 28.

34Flusser, “Hacia una,” 28.

35Flusser, “Hacia una,” 24.

36Arlindo Machado, “Repensando a Fluser”, en: El paisaje Mediático. Sobre el desafío de las poéticas tecnológicas, (Buenos Aires, Libros del Rojas, 2000) 5.

37Nota personal haciendo alusión a la última nota al pie.

38Pelta, Diseñar Hoy, 103

39Flusser, “Hacia una,” 19.

40Flusser, “Hacia una,” 19.

41Flusser, “Hacia una,” 24.

42Flusser, “Hacia una,” 43.

43Cuando menciono que no conoce el funcionamiento del software no me refiero al conocimiento técnico de la interfaz, o a un dominio de la misma, cosa que se nos enseña en la escuela de diseño y que con la practica se perfecciona. Me refiero a un conocimiento en un sentido mucho más amplio, al interior, el funcionamiento del programa en sí mismo, cómo es que genera tal o cual filtro, cómo se genera una curva vectorial, qué son las curvas de Bezier, y en general conocimientos que incluso, no solo nos llevara a resolver el misterio de la caja negra, sino que nos permitiese saber su funcionamiento y generar nosotros mismos una herramienta acorde a nuestras necesidades.

44Machado, Repensando a Fluser, 7.

45Oyarzún, Christian y Santorcuato, José, extracto de entrevista, (Santiago Chile, mayo 2011)

46Como el mito de que la MAC es el equipo por excelencia que usan los diseñadores.

47Cerezo, Diseñadores en la nebulosa, 61.

48Ricardo Vega Mora , “Tecnología, panoramas y paradigmas: cambios en la producción digital de la era digital” en Buscando Señal, lecturas sobre nuevos hábitos de Consumo Cultural, ed. Mariano Barbieri (Córdoba, Ediciones del Centro Cultural España Córdoba, 2009) 58-59.

49Joan Maeda, “Time Ghaphics, 1 of 3 essay for MDN magazine” Maeda Studio (Abril 1995 [citada en abril del 2011]):disponible en
http://www.maedastudio.com/1995/mdn1/index.php?category=all&next=exists&prev=exists&this=time_-grahics

50Pagola, “Software Libre,” 100.

51Machado, Repensando a Fluser, 5.

52Machado, Repensando a Fluser, 5.

53Pau Alsina “La intersección entre las artes y las tecnologías de la información y comunicación” en: E-cultura Les noves tecnologies aplicades a la gestió de la cultura, ed. Melba G. Claudio González (Barcelona: Associació de professionals de la gestió cultural de catalunya, 2006) 57.

54Siglas de sistema operativo.

55Joost Smiers Un mundo sin Copyrigth, arte y medios en la globalización, trad. Julieta Barba y Silvia Jawerbaum (Barcelona. Gedisa, 2006) 57.

56Flusser, "Hacia una," 34.

57Meggs, Historia del diseño gráfico, 490.

58Sánchez, "La revolución digital, " 255.

59Cerezo, Diseñadores en la nebulosa, 64.

60Cerezo, Diseñadores en la nebulosa, 44

61El precio de la Creative Suite CS5.5 Design Premium en 2011, del sitio oficial de Adobe para latinoamérica: http://www.adobe.com/mx/products/creativesuite/design.html

62En capítulos siguientes, se menciona algunos ejemplos de cómo el uso del S.P. limitó la distribución de algunas obras.

63 En la convocatoria del 2do. Concurso de Animación Digital "El cambio comienza con una idea... y puede ser la tuya" Organizado por el IMPI e INDAUTOR en el 2007. Se puede leer en el apartado IV CATEGORIAS DEL CONCURSO inciso a.- animación, sección v: "Los programas de cómputo utilizados en el desarrollo de los trabajos deberán acreditar licencias originales".... disponible en: www.impi.gob.mx/work/sites/.../ConvocatoriaFinal2ConcursoAD.pdf

64"Interoperabilidad" Wikipedia, La enciclopedia libre (4 de noviembre 2011 [citado en enero del 2012]): disponible en: http://es.wikipedia.org/wiki/Interoperabilidad#cite_ref-0

65Bruce Perens, Richard Stallman y otros, "Los líderes del Software Libre permanecen unidos",(citada en Julio del 2011) disponible en:http://www.smaldone.com.ar/documentos/docs/standtogether-es.html . Se trata de una carta abierta dirigida a Microsoft a raíz de las declaraciones de sus ejecutivos en contra del Software Libre.

66Pagola Lila, extracto de entrevista, (Córdoba Argentina, Junio 2011).

67Se refiere con sistemas abiertos, a los que permitan conocer, estudiar modificar y distribuir el código fuente, del cual se hablará con detenimiento en el capítulo 2.A licencias libres se refieren a licencias que tienen sólo "algunos derecnos reservados"

68Aymeric Mansoux y Marloes de Valk , “Prefacio” en Floss and art ed. Aymeric Mansoux y Marloes de Valk (Francia, GOTO10 y OpenMute, 2008) 9.

69Lawrance Lesing Cultura Libre, Trad. Antonio Córdoba/Elástico (E.U. Penguin Books, 2004) 130

70Serralde, José, extracto de entrevista, (México d.f., mayo del 2010)

71QuarkXPress es un programa de autoedición cuya primera versión apareció el 1987

72Indesign es un programa también de autoedición creado por la empresa Adobe cuya primera versión fue la de 1999

73Corel Draw Programa de edición gráfica creado en 1989 siendo de los primeros programas de diseño para windows.

74Mansoux y de Valk, Floss and art 6.

75Abiertas refiriéndose a las licencias que puede tener la obra, que pueden tratarse de licencias Creative Commons o bien, Licencias ligadas al arte como Licencia Arte Libre o Copyleft, qué en general en lugar de “reservar todos los derechos” pretende solo “reservar algunos derechos”, permitiendo que la obra pueda compartirse e incluso modificarse con el pleno permiso del autor.

76Mansoux y de Valk, Floss and art, 10.

77Josephine Berry Slater, “Bare Code: Net art and the free software movement” en Engineering Culture: Control and Commitment in a High-Tech Corporation ed. Gideon Kunda (E.U., Temple University Press, 2006) 133.

78Cerezo, Diseñadores en la nebulosa, 43.

79Cerezo, Diseñadores en la nebulosa, 34.

80Cerezo, Diseñadores en la nebulosa, 43.

81Cerezo, Diseñadores en la nebulosa, 47.

82Aunque uno adquiera la licencia del software patentado, no se es dueño de él, ya que no permite realizar muchas cosas, entre ellas, compartir y re-distribuír el software.

83Cerezo, Diseñadores en la nebulosa, 61.

84Pelta, Diseñar hoy, 39.

85Cerezo, Diseñadores en la nebulosa, 21.

86Lila Pagola, “Software libre, una caja abierta y transparente”, en: Instalando: Arte y cultura digital ed. Troyano: Ignacio Nieto, Italo Tello, Ricardo Vega (Chile, Lom Ediciones, 2007) 96.

%%title:

Capítulo Dos

<title>
El software libre en la creación visual digital
</title>

Una vez explorado algunas problemáticas del uso de la “caja negra”, este capítulo pretende hablar de la historia, características, problemáticas y ventajas del uso de un tipo de software distinto al usado comercialmente. El objetivo es reconocer el origen, definición y características del software libre y las implicaciones de su uso en el terreno de la creación visual digital; así mismo, la diferenciación con la “caja negra” y el porqué puede representar una alternarnativa “abierta y transparente”.

2.1.Un poco más de historia, surge una contra-propuesta, el movimiento de software libre.

2.1.1. Richard Stallman y el proyecto GNU. Linus Torvalds y LINUX

Siguiendo el tema de algunas problemáticas con el uso de la "caja negra", nos referiremos a una anécdota que marca el inicio de una contrapropuesta al software de patente y su consecuente ocultamiendo del código fuente.

En la década de los 70's, Richard Matthew Stallman[1], programador estadounidense del Laboratorio de Inteligencia Artificial (IA LAB) del Massachusetts Institute of Technology (MIT), cuenta que el laboratorio había recibido una impresora donada por la Compañía XEROX, que era una versión modificada de una fotocopiadora de la misma empresa. Dicha impresora, era usada por todos los trabajadores del Laboratorio, de tal suerte que todos podían mandar a imprimir a esta impresora desde sus computadoras conectadas en red. Sin embargo, algo parecía no estar funcionando muy bien a pesar de tratarse de un prototipo de vanguardia. Uno de sus principales problemas era que cada cierto tiempo el papel se atascaba y no se le notificaba al usuario de la situación con ningún aviso enviado por red. Evidentemente la pérdida de tiempo se volvió una constante, ya que en varias ocasiones, los trabajadores enviaban sus trabajos a imprimir y al ir a buscarlos se encontraban la impresora atascada y una cola enorme de trabajos pendientes.

Stallman quiso resolver el problema de la manera que estaba a su alcance siendo un programador y eso era, haciendo código. Se le ocurrió que una manera de resolverlo era haciendo que la impresora enviara por red una notificación de que el papel se había atascado y así, uno o más trabajadores resolverían el problema antes de que la cola de documentos pendientes se hiciera más grande. Para hacer esto, era necesario tener acceso al código fuente de los controladores de la impresora y así poder escribir las instrucciones para que la impresora las ejecutara.

Así que habló con XEROX y les explicó lo que quería hacer. La empresa se negó a darle el código fuente debido a su política de privacidad y propiedad del código. Cuando intentó pedír-

selo a un colega suyo que trabajaba para XEROX, éste le dijo que había prometido a la compañía no entregar una copia del código fuente a nadie, proveniente de un acuerdo contractual con la empresa.[2]"Actualmente este es un ítem estándar en el negocio de la industria de software, pero el acuerdo de no revelar, o NDA por sus siglas en inglés, era un desarrollo nuevo en esa época, un reflejo tanto del valor comercial de la impresora láser de XEROX como de la información requerida para manejarla. "Xerox estaba en esa época tratando de hacer de la impresora láser un producto comercial y habría sido una locura el entregar el código fuente"[3].

El software se había convertido en una posesión tan valiosa que las compañías dejaron de sentir la necesidad de publicar el código fuente, especialmente cuando la publicación significaba entregar a potenciales competidores la oportunidad de duplicar algo de manera económica. Desde el punto de vista de Stallman, la impresora era una caballo de Troya. Después de una década de fracaso, el software apropiado privadamente había ganado una posición firme dentro del Laboratorio de IA a través uno de los métodos más solapados. Había llegado disfrazado de regalo.

"Fue mi primer encuentro con un acuerdo de no revelar (NDA), e inmediatamente me enseñó que los acuerdos de no revelar tenían víctimas," dice Stallman. "En este caso yo fui una víctima. Mi laboratorio y yo fuimos víctimas."[4]

Richard Stallman se fijó firmemente defender la práctica del intercambio y la "libertad" de los usuarios del software, tal como lo menciona Sam Williams en Libre como en libertad:

"Desde ese día en adelante, decidí que esto era algo en lo cual yo nunca participaría," dice Stallman, "Decidí nunca convertir a otras personas en víctimas así como yo había sido una víctima."[5]

A partir de esta experiencia, Richard Stallman tomó la iniciativa en 1984 de escribir un sistema operativo que denominó GNU, cuya característica principal consistiría en que cada parte que lo compone y lo hace funcionar fuera realizado por una comunidad de programadores que lo mejorara y compartiera, así cada persona que tuviera acceso a él podría modificarlo y adaptarlo según sus necesidades y plataforma que se utilizara, aspirando así a crear el sistema operativo perfecto a través de un principio básico de compartición del conocimiento[6] y

lejos de los acuerdos de confidencialidad. En 1985, Stallman fundó la Free Software Fundation (Fundación Software Libre) mientras que GNU comenzaba a utilizarse entre algunas comunidades de programadores. Inició el movimiento del **software libre**[7]

GNU es un acrónimo recursivo que significa Gnu is Not Unix o Gnu no es Unix. *UNIX* se refiere al sistema operativo popular de los 70's que posee una patente y del que hablamos en el capítulo anterior, con el cual GNU es compatible, pero no resulta idéntico ya que, contrario a UNIX, GNU puede ser modificado por cualquier persona. No obstante, a ningún distribuidor se le permitirá restringir su redistribución posterior. Es decir, "no estarán permitidas modificaciones propietarias. Hay que asegurarse de que todas las versiones de GNU permanezcan libres" [8].

Paulatinamente, el proyecto GNU causó una transmisión acelerada de información por compartición, ya que varios programadores decidieron contribuir con él. Fue en 1991 cuando el finlandés Linus Torvalds programó el núcleo y la parte fundamental para el sistema operativo GNU, el kernel, que fue llamado LINUX, basándose en un sistema operativo con fines didácticos, Minix. Hoy día, LINUX es uno de los programas libres más utilizados para el sistema operativo GNU[9]. A partir de que LINUX es liberado, dicho sistema operativo empezó a evolucionar rápidamente debido a que muchos programadores han contribuido a su desarrollo y mejora, lo que lo convierte en uno de los sistemas operativos más estables.

Con GNU y LINUX juntos, comienza uno de los crecimientos más importantes del software libre, la incursión en el mundo empresarial, ya que algunas empresas comenzaron a basar su

Richard Stallman, fundador de la Free Software Fundation y padre del software libre.

estrategia de negocios en hacer consultorías, distribución y servicios alrededor de este nuevo y creciente sistema operativo. La estrategia fue comenzar a vender "misterios" al rededor del software libre los cuales podían ser resueltos por quién supiera de lenguaje de programación y no "secretos" indescifrables que una sola empresa conociera. No se trata de vender el secreto del código, sino la libertad del servicio.

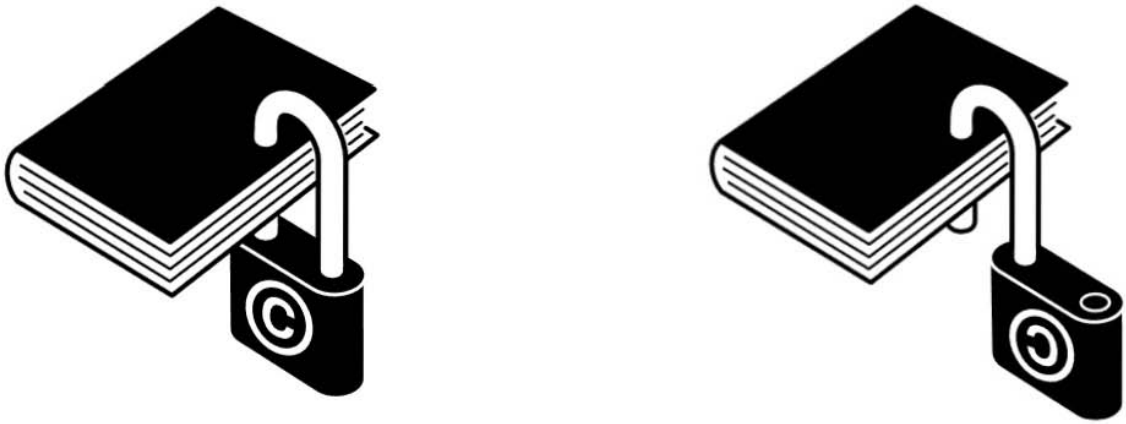
Con estos antecedentes, podemos acercarnos a una definición formal del software libre, cuya ideología está basada en el concepto de "libertad", tal como se explica a continuación.

2.1.2. Definición de Software Libre

El término **Software Libre** o Free Software se refiere a aquellos programas cuyo código está abierto para que cualquier usuario pueda modificarlo, usarlo, estudiarlo y/o distribuirlo sin que esto represente un problema legal. El término **"código abierto**" o en ingles, "open source", se refiere a que las lineas de texto en las cuales están escritas las instrucciones que sigue la computadora para ejecutar dicho programa, están disponibles para que todos los usuarios puedan conocerlas y así, mejorar o adaptar dicho código según sus necesidades. Sin embargo, el S.L. no se trata de una copia no autorizada ni mucho menos de aquel descargable desde Internet de manera gratuita, el llamado freeware.

El software libre debe poseer la autorización para utilizar un código abierto. Free software y open source como términos distintos pero complementarios. El software libre con sus lineas de código abierto, representa la libertad de que cualquiera puede hurgar en él, lo cual a la larga permitiría que el programa sea modificado e incluso mejorado a una velocidad mucho más eficiente que el desarrollo del software patentado, llamado también software propietario, dando como resultado la creación de un mejor software con la colaboración de muchas personas a la vez, tal como lo explica Raymond en La catedral y el bazar en lo que él llama la ley Linus:

"Dada una base lo suficientemente amplia de probadores y colaboradores, casi todos los problemas se identificarán con rapidez y su solución será obvia para alguien. O expresado con menor formalidad: 'con un número de ojos suficiente, todos los errores son irrelevantes'[10]



Candado cerrado y candado abierto como analogía al código privativo y al código abierto, respectivamente.

Ahora bien, el término **free software** fue acuñado en ingles, por lo que, puede existir una confusión en dicho idioma ya que la palabra **Free** se refiere a "libre" y al mismo tiempo al concepto de "gratis", es por eso que la **Free Sofware Fundation**, aclara que:

"El software librees una cuestión de libertad, no de precio. Para entender el concepto, debería pensar en librecomo en libre expresión, no como en barra libre."[11]

Es por eso que el S.L. puede estar disponible para su distribución comercial, sin que eso le quite su calidad del "libre", esto es, que se pudo pagar alguna cantidad para obtener una copia de S.L. para cubrir algunos costos de distribución o bien, obtenerlo sin costo alguno, ambas opciones no afectan que un software siga siendo libre o no siempre y cuando se cumplan las 4 libertades

Estas 4 libertades son a su vez los 4 principios del software libre, son 4 libertades que tienen los usuarios del software y que necesariamente deben de ser cumplidos para que se le considere como tal. Dichas libertades fueron acuñadas por el propio Richard Stallman y posteriormente fueron bandera de la **Free Software Fundation**.

Libertad 0: Usar el software para cualquier propósito

Dichas libertades son las siguientes:

Libertad 0.- Libertad de usar el programa para cualquier propósito

Libertad 1.- Libertad de estudiar el funcionamiento del programa y adaptarlo a las necesidades. El acceso al código fuente es una condición previa para esto.

Libertad 2.- Libertad de redistribuir copias, y con ello poder ayudar al prójimo

Libertad 3.- La libertad de distribuir copias de sus versiones modificadas a terceros. Si lo hace, puede dar a toda la comunidad una oportunidad de beneficiarse de sus cambios. El acceso al código fuente es una condición necesaria para ello.

De estas libertades y ejemplos prácticos en el quehacer humano hablaremos un poco más adelante.

Así pues, el sistema operativo GNU / LINUX se convierte en uno de los pilares de la comunidad del software libre, una comunidad que defiende y promueve la libertad de usar los programas y adaptarlos para cualquier propósito, de mejorar su programación, de estudiar su funcionamiento y distribuirlo libremente. Puede ser entendido como una alternativa al software de patente y la "privatización" del código, así como una filosofía que no sólo se refiere a programas de computadora, sino que sustenta a un movimiento social, político y de resistencia que defiende ciertas libertades.

2.1.3 Las 4 libertades, la ideología detrás del software libre

Ya que mencionamos brevemente cuáles son las libertades del software libre, hablemos específicamente de cada una de ellas y analicemos algunos ejemplos para sustentarlas.

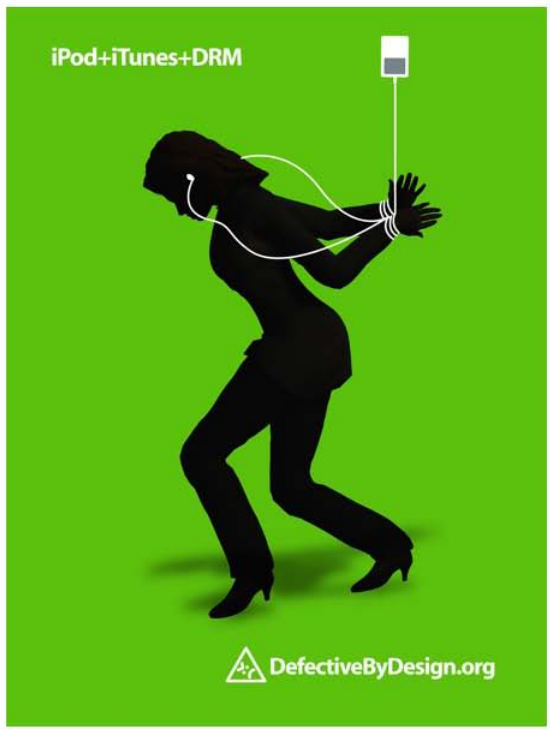
Libertad 0

En el caso de la libertad 0, la libertad de usar el programa para cualquier propósito, se refiere a que los usuarios pueden utilizar el programa para cualquier fin sin necesidad de notificarlo a algún programador que haya intervenido en la realización de dicho software, o a

cualquier otra entidad o institución. En esta libertad importa el propósito del usuario y no del programador, de igual suerte que el usuario puede distribuirlo a otras personas y dichos nuevos usuarios utilizarlo para sus propios propósitos.

Ejemplos de cómo algunos programas de patente no permiten esta libertad, encontramos varios, como el caso de Photoshop que impide a los usuarios abrir imágenes detalladas de algunos billetes como el dolar, el euro y la libra esterlina (sucede con otros papel moneda dependiendo de las leyes de cada país). Photoshop ha incluido la tecnología CDS[12]que reconoce los sellos de agua impresos en el billete e imposibilita su edición.[13]Sucede lo mismo con el software del **iPod**, ya que al comprar una canción en la Apple Store, **iTunes**, no permite que se copie en un equipo que no sea ese **iPod** y/o la computadora asociado a su dueño. Otro ejemplo más o menos reciente fue en el 2009 cuando Amazon borró dos libros de George Orwell, **"Rebelión en la granja"** y "1984" de más de miles de lectores electrónicos Kindle ya vendidos, gracias al control remoto que poseía la empresa respecto al software que hace funcionar el Kindle. ¿La razón?, habían sido publicados por una empresa que no tenía los derechos para hacerloen Estados Unidos[14]. Y por último, el reproductor de música de Windows, el Windows Media Player, cuyas nuevas versiones no permiten la reproducción de canciones que tengan dispositivo DRM[15].

DRM, Digital Rights Management o gestión de derechos digitales, tecnologías de control de acceso para limitar el uso de medios creativos



Libertad 1

Para la libertad 1 que es la libertad de estudiar el funcionamiento del programa y adaptarlo a las necesidades, el acceso al código fuente es una condición previa, ya que para estudiar el funcionamiento del software es necesario tener un libre acceso a sus líneas de código, esto permitirá que posteriormente se modifique según las necesidades del usuario, si así lo requiere. Esta libertad incluye que se puede utilizar esa versión modifi-

Logo de Firefox, navegador de código abierto que actualmente es traducida a diferentes lenguas originaras mexicanas

cada en lugar de la original. El software deja de ser software libre en tanto el código no sea libre y no pueda ser modificado, y mas aún, que no se puedan utilizar dichas modificaciones.

La web es un ejemplo amplio y basto de las aplicaciones que tiene esta libertad, ya que al ser el HTML y el CSS lenguajes descriptivos definidos de manera estándar y abierta, y en lo que muchos sitios están desarrollados, estos pueden ser estudiados para conocer su comportamiento. Es posible ver el código con el que fue hecha una página programada en HTML/CSS y conocer cómo fue realizada, lo cual propicia mejoras futuras. Gran parte del crecimiento tan exponencial que ha tenido la web se debe a que está sustentada bajo un esquema de código abierto.

El estudio del código fuente permite también su modificación, y a su vez, la adaptación de un programa a necesidades muy específicas que de otro modo no hubiesen sido solucionadas, por ejemplo, existen iniciativas para que el navegador libre Firefox sea traducido a lenguas originarias mexicanas gracias al apoyo de algunas asociaciones civiles y a programadores entusiastas mexicanos. Hasta agosto del 2010, se encontraba lista la versión en zapoteco, tarahumara y el maya en un 80%, y se está trabajando en traducirla al náhuatl[16].Ejemplos similares ocurren en Bolivia y en Peru.[17]

La herramienta Spiro, integrada a las nuevas versiones de algunos programas libres, es otro ejemplo de las ventajas de poder estudiar y modificar un código abierto. En el 2009 Raph Laven, desarrolló una técnica para crear curvas interactivas en el diseño digital para su tesis de doctorado titulada "From Spiral to Spline: Optimal Techniques in Interactive Curve Design"[18]. Dichas técnicas, que son muy útiles para el diseño de tipografías, se integraron en Spiro, herramienta que estuvo disponible en las más recientes versiones de programas libres como Inkscape[19]y FontForge[20]. Gracias a la apertura del código de Inkscape, Laven pudo desarrollar una herramienta para este programa y pudo ser integrada después para el beneficio de todos los usuarios. Laven también liberó el código para que todos puedan ver cómo pro-

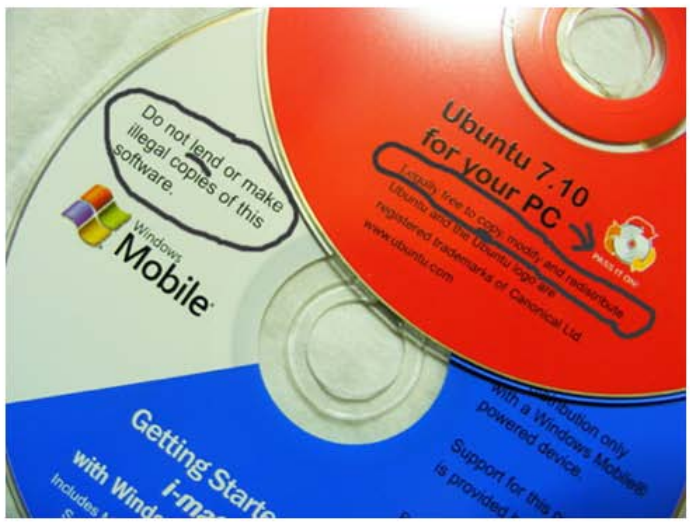
gramó Spiro[21]. Century Catologue, Inconsolata, Museum Caps, LeBe, ATF Bodoniy ATF Frankling Gothic[22], son fuentes tipográficas diseñadas a partir de Spiro.

Libertad 2

La libertad 2 que es la de redistribuir copias, debe incluir el código fuente; tanto para las versiones modificadas como para las no lo están.

Esta libertad tiene implícito el "derecho" de distribuir copias de software libre a manera de compartir una herramienta, ya que si estamos en el entendido que el software es una herramienta que ayuda a solucionar problemas, en el caso que nos confiere, una herramienta de creación, ¿debería de impedirse que compartamos dicho software con el fin de ayudar al prójimo?. Prestar o distribuir copias de software libre es una manera de ayudar a la gente a solucionar problemas y que dicha actividad sea completamente legal.

Ejemplos de esto lo mencionamos ya en el capitulo I, donde explicamos la dificultad gracias al alto costo, de conseguir una copia con licencia de software privativo. El S.L. representa una herramienta que puede compartirse sin ningún problema legal.



Diferencia entre un disco de instalción de Windows y uno de la distribución libre, Ubuntu. El primero dice, en inglés: Legalmente libre de copiar, modificar y redistribuír. el segundo dice: No prestar o hacer copias ilegales de este software.

Libertad 3: distribuir las copias de versiones modificadas a terceros

Para la libertad 3, la de distribuir copias de sus versiones modificadas a terceros, el acceso al código fuente también es una condición necesaria, si se distribuye, puede dar a toda la comunidad una oportunidad de beneficiarse de sus cambios.

Esta libertad tiene implícita la idea de colaboración, ya que al compartir los cambios o mejoras ayuda a que todos los usuarios de dicho software avancemos de una manera más eficaz y rápida. Es la metáfora de subirse en hombros de gigantes, ayudarse de lo que ha hecho el prójimo con sus conocimientos y utilizarlo al beneficio propio y a su vez[23], ser parte de esta comunidad que proporciona y se beneficia de los avances que entre todos, se realizan. Todo esto mediante un acuerdos de partes, es decir, licencias que protegen la distribución del S.L., una de ellas es el GPL, basado en la idea del copyleft, de la cual hablaremos a profundidad más adelante.

Un caso de violación a la GPL que ha sido llevadas a juicio fue en el 2007, donde la compañía vendedora de software Monsoon Multimedia, violó la licencia GPL cuando ofreció el programa libreBusyBox sin permitir el acceso al código del programa a quien lo adquiría. El resultado, fue que Monsoon Multimedia acordó nombrar a un oficial de cumplimiento de código dentro de sus organización para controlar y asegurar el cumplimiento de la GPL, publicó el código fuente para la versión de BusyBox que estaba distribuyendo y notificó a los beneficiarios anteriores de BusyBox sus derechos sobre dicho software. El acuerdo también incluyó una compensación económica a los demandantes[24].

Así pues, la ideología del software libre plantea que las tecnologías digitales de la información ayudan en muchos sentidos a que ciertas actividades resulten más cómodas y sencillas, y sobre todo que "ayudan al mundo haciendo que sea más fácil copiar y modificar información"[25], pero que no a todos le resulta conveniente que esto sea tan fácil, como a los grandes corporativos que se dedican a la venta de licencias de software.

Según estas ideas, los programas de computo no debe de tener un "dueño" que se reserve los derechos, y eso no significará que no sea "autor" del mismo[26]. Se basa en la premisa de que el "conocimiento es de todos", pues al compartir dicho conocimiento ocurre un beneficio, no solo particular, sino a nivel comunitario. Al mejorar un software con la ayuda de una

El abogado estadounidense, Lawrence Lessig, se inspira en la filosofía del software libre y acuña recientemente el concepto de Cultura Libre, la cual define la visión de una cultura basada en la libertad de distribuir y modificar trabajos y obras creativas: "Las culturas libres son culturas que dejan una gran parte abierta a los demás para que se basen en ella, las que no son libres,las culturas del permiso, dejan mucho menos.[27]

Se refiere a basarse en lo que han hecho otros, en lo que se ha hecho en el pasado, en una cultura basada en la compartición y distribución libre del conocimiento.

El conocimiento colaborativo es otro de los pilares del S.L. esto es que entre mas personas colaboren para la creación de un software, se creará un mejor programa a una velocidad mucho mayor y los usuarios se verán beneficiados con ello y parte del principio que "la compartición con los demás constituye la base de la sociedad"[28]

Dentro de la comunidad de usuarios del S.L. se hacen públicas las mejoras que se realicen a algunos programas, esto es accesible para quien este interesado en usarlo e implementarlo y permite que un solo usuario utilice un programa combinado con otro según sus necesidades. También permite que el propio usuario desarrolle su propia herramienta que solucione sus problemas específicos de producción.

Resumiendo las ideas planteadas hasta ahora, el S.L. se basa en la cultura libre, conocimiento colaborativo, compartición de conocimiento, libertad en el conocimiento, copyleft y sobre todo, libertad.

Cultura libre, cómo los grandes medios usan la tecnología y la ley para cerrar la cultura y controlar la creatividad.
Por el abogado estadounidense Lawrence Lessig

Una "patente" trata de proteger "ideas" y al protegerlas, imposibilita que los demás las usen y por lo tanto que se generen otras ideas a partir de ellas

2.14. Software libre y software privativo: licencias que "permiten", licencias que "privan"

Una vez que hemos hablado en el capitulo I del software patentado y en el inicio de éste, del software libre, es pertinente dedicar un espacio a hablar de las licencias que diferencian a ambos tipos de software.

Por un lado, el software de patente, basa sus licencias en el concepto de copyright, es decir, que sus autores se reservan todos los derechos, entre ellos, el código fuente con el que fue creado, impidiendo que sea visible y por lo tanto que otras personas lo usen, distribuyan, copien o modifiquen de manera legal, sin el permiso por escrito del titular de los derechos. El código se encuentra protegido por una licencia, que es necesario pagar si es que quiere utilizarse de manera legal. Sin embargo, esta licencia sólo da el permiso de "usarlo" e incluye algunas restricciones, como instalarlo en más de una computadora, por ejemplo; el código fuente pertenece a una empresa, compañía, fundación o persona y sus autores se reservan todos los derechos.

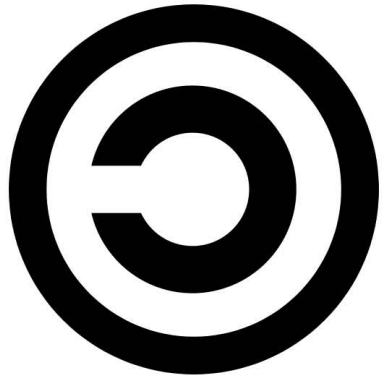
Sin embargo, no sólo el término de Copyright está relacionada con este tipo de software, también lo está el término "patente" y es aquí donde comienza a ser engañoso.

Una "patente" trata de proteger "ideas" y al protegerlas, imposibilita que los demás las usen y por lo tanto que se generen otras ideas a partir de ellas: "Toda patente protege alguna idea. Las patentes de software son patentes que protegen ideas que tienen que ver con el software, ideas que podrían usarse para desarrollar software"[29]Además, son pocos los países que aceptan patentar ideas computacionales (como es el caso del software.)

El problema de "patentar" el software valiéndose del copyrigth, es que los conceptos se confunden a menudo y engañan al usuario diciendo que se están "protegiendo" los derechos de los autores (cuando en realidad protegen el derecho de las empresas, pues en la práctica, los autores seden sus derechos a los editores y/o compañías para las que trabajan). Una diferencia entre el copyright y las patentes, es que la primera regula las condiciones de expresión de una obra, no protege ninguna idea, mientras que la segunda sólo protegen las ideas y el uso de las mismas. Además, el copyright se aplica automáticamente. Las patentes son publicadas por una oficina de patentes como respuesta a una solicitud y son mono-

polios absolutos sobre el uso de una idea. Incluso si pudiéramos probar que la idea es nuestra, sería completamente irrelevante si la idea ha sido patentada por otro[30].

Ejemplos de software patentadoson, en sistemas operativos, los comúnmente usados: Windows en todas sus distribuciones y OSX para MAC, en programas de oficina, Word, Excel, Power Point y todos los pertenecientes a la distribución de Microsoft Office. En programas para edición y creación de imágenes, movimiento y animación que se han convertido en las herramientas de diseñadores y artistas visuales, tenemos a Photoshop, Illustrator, Flash, 3D Max, Maya, Dreamweaver, Indesign, Premier, Pro tools, After Efects, entre otros.



Logo de Copyleft

Hace algunos años, entre la comunidad de software libre se usaba el término "propietario" para referirse a este software de patente, sin embargo, Enrique Chaparro, un activista argentino de software libre propuso en el 2003, el uso del término "**privativo**" dado que es un adjetivo que expresa más claramente la noción de que este tipo de software "priva" ciertas libertades de los usuarios. Es por esta razón que este tipo de software, que mantiene su código fuente no abierto, que no permite ser copiado ni reproducido, mucho menos distribuido sin autorización de sus creadores, será llamado a partir de ahora **software privativo**[31].

Por otro lado, la intención del movimiento del software libre de Stallman es producir código en la medida en que pueda ser transparente y susceptible de modificación haciéndolo libre. El mecanismo para que esto sea posible es un instrumento llamado copyleft que se implementa a través de una licencia llamada GPL (General Public License). Tal como lo menciona Lessig en la introducción de **Software libre para una comunidad libre**: "Usando el poder del copyright, el software libre no sólo asegura que permanecerá abierto y susceptible de modificación, sino también que otro software que incorpore y use software libre y que técnicamente se convierta en obra derivada, debe también, a su vez, ser libre" [32]

El **copyleft** determina que un software (o cualquier otra creación) se distribuya respetando, a su vez, sus características de libre distribución. Esta licencia establece que al redistribuir un programa no puedan añadirse restricciones que falten

con algunas de las libertades anteriormente mencionadas, de tal forma que estas libertades quedan protegidas: "si uno usa y adapta un programa de software libre y distribuye públicamente esa versión adaptada, la versión distribuida debe ser tan libre como la versión de la que procede. Debe hacerse así, de lo contrario se estará infringiendo el copyright"[33]. Es darle la vuelta al concepto del copyright y en "lugar de privatizar el software ayuda a preservarlo como software libre".

Así pues, la licencia GNU-GPL protege los derechos por medio de la combinación de dos medidas: 1.- poner el software bajo copyright y 2.- la propia licencia que otorga permiso legal para copiar, distribuir y/o modificar el software.

El preámbulo de esta licencia dice:

"Las licencias que cubren la mayor parte del software están diseñadas para despojarle de la libertad para compartirlo y para modificarlo. Por el contrario, la Licencia Pública General de GNU pretende garantizar la libertad de compartir y modificar software libre para asegurar que el software sea libre para todos sus usuarios. Esta Licencia Pública General se aplica a la mayor parte del software de la Free Software Foundation y a cualquier otro programa, si sus autores se comprometen a utilizarla".[34]

Una de las posturas de este trabajo es la identificación de 2 corrientes en el uso y realización del software. Por un lado el software privativo que "protege" los derechos de las empresas y realizadores y por otro el software libre que "protege" los derechos de los usuarios.

Free Software Fundation

2.2. Software Libre en el terreno de la creación visual digital

Una vez que hemos visto de manera detallada las principales diferencias entre los dos tipos de software, ha llegado el momento de hablar de las ventajas y las problemáticas de la adopción del software libre. Se hablará de algunas razones por las cuales podría ser necesario considerar el uso del software libre para crear visualmente por medio de un equipo de computo, sin que eso represente necesariamente una comparación con el software privativo.

2.2.1. Caracterización, problemáticas y diferencias con el software privativo

A) EL ASPECTO TÉCNICO

El S.L. puede modificarse, lo cual lo hace flexible, permite integrar soluciones: El acceso al código fuente permitiría al artista y/o diseñador realizar las modificaciones al software que crea convenientes y aunque no se tengan conocimientos de programación, se puede recurrir a la interdisciplina y con la ayuda de un programador, modificar el código y obtener funciones nuevas. Esto es mucho más eficiente en el caso del arte electrónico, por ejemplo, en el que las funciones que debe realizar una obra son muy específicas y es mucho más complicado (y caro) llevarlo a cabo con software privativo. Por ejemplo, en el mes de abril del 2010, el Festival de México FMX presentó en el teatro Julio Castillo la Ópera titulada *"Únicamente la verdad, la verdadera historia de Camelia la Texana"*, en donde, una de las particularidades de la puesta en escena, consistía en controlar en tiempo real un video a 4 pantallas. El problema fue resuelto con"unas tarjetas de vídeo (modelos viejos), y unos scripts corriendo en Linux, (que) sustituyeron los más de $100,000 pesos que habría costado la renta de las licencias y computadoras de cualquier programa privativo para resolver el problema."[35]

Esto se debe a que el carácter modular del S.L. permite la implementación de soluciones, permitiendo "ensamblar" programas libres entre si; tal como lo menciona una de las filosofías del sistema operativo *UNIX* del cual hablamos en el capítulo anterior, "escribir programas que trabajen juntos". La nobleza de poder controlar, modificar y es-

tudiar un código abierto, permite que puedan ser usados scripts que alguien más ya escribió (o escribirlos uno mismo) y mejorarlo para satisfacer las necesidades específicas de cada usuario.

Otro ejemplo práctico de esto es el programa libre de modelado en 3D, Blender. Con Blender se han desarrollado producciones animadas de muy alta calidad, como Big Buck Bunny, Elephants Dreams y la más reciente Sintel. Con cada una de estas producciones se han implementado nuevas herramientas dentro del propio programa según las necesidades del equipo de producción. Para Sintel (2010, Amsterdam) por ejemplo, se llevaron a cabo modificaciones, entre otros, en la interfaz de usuario, en el sistema de partículas (lluvia, polvo), en el esculpido, el sombreado, el sistema de rendición y en la simulación de humo[36]. "Cómo decía Roosendaal de la Blender Foundation, Blender no se trata de un software, se trata de artistas interconectados comunicándose y generando cosas juntos, se trata de comunidad, no de programas"[37]

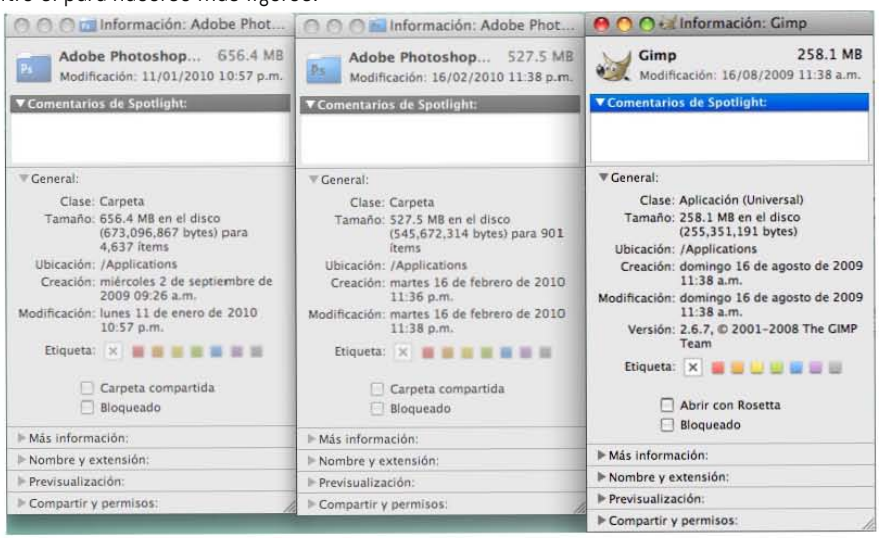
Otro ejemplo más de cómo esto puede llevar a desarrollos importantes de soluciones artísticas, es el caso de PURE DATA, un programa para la manipulación de datos y medios para los cuales, varios desarrolladores escriben el código necesario para resolver diversas necesidades. Cada línea de código se convierte en una linea que hace crecer la librería del código. Recientemente, Hans Christoph Steiner en colaboración con Golan Levin, desarrollaron una fórmula matemática implementada en curvas para animación. Dichas curvas fueron necesarias para la animación de un aliento elegante y bello. Si PURE DATA tuviera un código cerrado, hubiese sido imposible implementar estas curvas y el trabajo artístico probablemente hubiese sido detenido para negociar con alguna compañía de software para que incluyera la solución de dicha necesidad, sin mencionar el costo que esto implicaría. Por el contrario, la implementación que fue posible gracias al software libre, hace que estas herramientas estén disponibles para la comunidad, permitiendo a todos trabajar para alcanzar un nuevo nivel.[38]

Acceso al proceso de las obras. El código abierto permite también que sea más fácil saber cómo se realizó una obra determinada, conocer los plug ins, herramientas y cómo se llevó a cabo el proceso de producción digital. Esto es importante al momento de la conservación y difusión de la obra en un futuro donde no sabemos hacia donde y cómo avance la tecnología, con el uso del S.L. no será necesario pedir permiso a alguna compañía para que nos permita el acceso al código de nuestra obra y que futuras generaciones la analicen si es necesario.

Un ejemplo de esto es el programa libre Alchemy, disponible en: http://al.chemy.org. Alchemy permite realizar trazos digitales como en cualquier programa de trazo libre, (como paint, por ejemplo), con la diferencia de que dichos trazos pueden ser vectorizados y vueltos a ver en la secuencia en la que fueron creados, pues existe una salida para archivos con formato estándar SVG y editado posteriormente en un archivo de edición de vectores, además de que es posible grabar los movimientos que realizamos en dichos trazos y encontrar patrones. El archivo generado en la

obra puede ser abierto en un editor de texto y poder ver, en líneas de código, la característica de cada uno de nuestros trazos, como las coordenadas de los polígonos, el grosor del trazo, el color, el filtro utilizado, etc. Esto podría permitir estudiar una nueva generación de arte digital.

El S.L. es multiplatadorma y no requiere de excesivas especificaciones de hardware. Imaginemos que tenemos la oportunidad de realizar un curso de ilustración digital en una comunidad de difícil acceso y sólo se nos ha proporcionado algunos equipos de muy baja capacidad para dichos fines. En una PC pentium 2, por ejemplo, sería muy complicado instalar Illustrator CS5 debido a que las especificaciones de dicho software requiere de una capacidad mayor a 1G de memoria RAM y un Procesador Intel® Pentium® 4 o AMD Athlon® de 64 bits[39]. En el caso de Inkscape por ejemplo, el homólogo de Illustrator en S.L., es un programa muy ligero que no requiere mas de 512k de RAM. Una de las razones por las que los programas libres son más ligeros se debe a que comparten bibliotecas entre sí, Inkscape y Gimp (programa libre que realiza funciones similares a Photoshop) comparten GTK[40]que es una biblioteca que se encarga de desarrollar las interfaces gráficas de usuario, como los botones del menú. En lugar de que cada programa tenga su propia biblioteca y sus propios botones y con ello, ocupe más espacio en el disco y en la memoria RAM, Inkscape y Gimp la comparten entre si para hacerse más ligeros.



Captura de pantalla comparando el peso de Photoshop y de Gimp.

Además de esto, es común que algunos S.P. como Photoshop especifiquen que las tarjetas de video, el RAM o incluso la resolución de las pantallas no son suficientes para algunas tareas. Eso no sucede en el caso del S.L. que en general precisa de pocos requerimientos de hardware para sus instalación.

Para usar programas libres no se requiere necesariamente que corran en el sistema operativo GNU/LINUX, algunos programas para creación gráfica digital como Inkscape, Blender, Gimp, My paint, entre otros, corren en el sistema operativo Windows y en OSX.

Uso de estándares: Hablamos en el capítulo I de la importancia del uso de estándares y como éstos permiten que una obra digital pueda permanecer en el tiempo.Los programas libres, así como el movimiento del S.L. promueven el uso de estándares en la generación de archivos realizados con esta herramienta, esto es con el fin de que nuestra creaciones, información y contenido que generemos pueda ser visto por todas las computadoras independientemente del sistema operativo que las haga funcionar, es decir, no importando la compañía que nos venda el software, de sus características, deficiencias o virtudes. Además, cuando hablamos de estándar nos referimos a que este sea [41]:

1.- Sujeto a evaluación pública y uso sin restricciones de una manera igual en todas las partes.

2.- Carezca de ningún tipo de componentes o extensiones que tengan dependencias sobre formatos o protocolos que no cumplen la definición de un estándar abierto en sí mismos.

3.- Libre de cláusulas jurídicas o técnicas que limiten su utilización por cualquiera de las partes o en cualquier modelo de negocio.

4.- Gestionado y desarrollado independientemente de cualquier proveedor en un proceso abierto a la igualdad de participación de los competidores y terceros.

5.- Disponible en múltiples implementaciones completas compitiendo por proveedores que compitan entre sí, o como una implementación completa igualmente a disposición de todas las partes.

El uso de estándares nos permitiría a la larga elegir cualquier sistema operativo o aplicación y que podamos abrir y editar nuestros archivos antiguos, por ejemplo, los realizados en algu-

na versión anterior del software sin que eso represente cambios en dicho archivo y por supuesto, permitiría utilizar cualquier software para generar documentos y que cualquier persona pueda abrirlos.

Lila Pagola, comenta su experiencia al respecto: "Procuro guardar en formato estándar por que es donde tengo menos sorpresas entre lo que yo quiero hacer y lo que después sucede con el archivo cuando lo abro tiempo después o en otro programa"

Existe menos incidencia de virus: Hay tantas distribuciones de GNU/LINUX que sería en principio muy complicado que el virus funcionara en todas, y como estas distribuciones de igual forma están en constante actualización, la ejecución de un virus se complica más. Además de que en GNU/LINUX cada usuario tiene ciertos privilegios, si existe un virus, sólo puede estar en la carpeta de dicho usuario y tener sus privilegios, no puede entrar a elementos del sistema que lo haga desvariar, es decir, para poder acceder al interior del sistema es necesario ser un usuario root[42]. Esta cuenta tiene todos los permisos para hacer cualquier cosa en el sistema, incluso, se aconseja no usarla a menos de que se trate de un usuario experimentado o tenga conocimiento del sistema operativo. Para acceder a este control del sistema se requieren ciertos privilegios de superusuario a través de la instrucción sudo[43], un comando que se teclea en la terminal[44]del sistema operativo seguida de nuestra contraseña. Esto impide que cualquier persona ajena a nosotros tenga control del sistema o programas maliciosos se ejecuten sin nuestro permiso, entre ellos, los llamados virus informáticos. Además de que GNU/LINUX, basado en el sistema operativo UNIX, está hecho de forma "modular". Por otro lado, es importante decir que nos referimos al sistema operativo libre GNU / LINUX, en el caso de programas libres que corran sobre un sistema operativo privativo, es decir de WINDOWS y/o OSX, estos quedarán supeditados a las características y seguridades de dichos sistemas.

Estable y seguro: El punto anterior tiene mucho que ver con esta ventaja, ya que el hecho de que no haya virus peligrosos para GNU/LINUX habla de la seguridad que ofrece, no sólo por que se necesitan privilegios para entrar a modificar el sistema, sino también por que este sistema operativo cuenta con aproximadamente 5000 desarrolladores de más de 200 compañías que respaldan sus líneas de código[45], además de que, miles de ojos de programadores de todo el mundo están puestos en las 2.7 millones de líneas de código abierto del kernel de linux y en los errores que éste pudiera tener, en mejorar el rendimiento y reforzar la

Para adquirir una licencia de la Adobe Creative Suite Design, un diseñador tendría que invertir unos 384 salarios mínimos en México

seguridad, haciendo que las mejoras sean realizadas en un periodo de tiempo mucho más corto y los problemas detectados con mayor prontitud. Una nueva versión de kernel es liberada al cabo de 8 a 12 semanas, y está lista para que los usuarios la descarguen e implementen. En estados Unidos, el departamento de defensa, la flota de submarinos de la fuerza naval, la administración federal del aviación, la corte federal, entre otros, así como el gobierno de la ciudad de Munich, el parlamento francés y el banco industrial y de comercio estatal de China, han migrado a sistemas operativos libres, entre otras razones por la seguridad que ofrece[46].

B) ASPECTOS ÉTICOS Y SOCIALES:

Bajo costo. Utilizar software libre puede representar un muy bajo costo de producción al momento de buscar programas que nos ayuden a generar imágenes. En el caso de las Organizaciones No Gubernamentales (ONG) o Asociaciones Civiles (A.C.) en las que no siempre se cuenta con un presupuesto dedicado a la inversión de licencias de software y que requieran generar material gráfico y de difusión para su promoción, el S.L. puede ser una buena opción sin necesidad de recurrir a la copia no autorizada del software, de igual forma que puede serlo en la industria cultural. Cabe recordar que el costo del software libre se ve reflejado en la eventual capacitación para su uso, el cobro de honorarios de las personas que realicen la migración y en el soporte técnico del mismo.

Como mencionamos ya en el capítulo anterior, la Adobe Creative Suite Design Premium tiene un costo de U$1 899[47]dólares, eso es poco más de $23 000 pesos mexicanos, lo cual indica que un diseñador tendría que invertir unos 384 salarios mínimos en México para adquirir la licencia. Además que este costo no sería la única inversión, pues las licencias deben ser renovadas cada que se lleve a cabo una actualización, esto es, aproximadamente cada año. Con el S.L. tampoco es necesario adquirir una nueva versión, existen actualizaciones que pueden realizarse conectándose a la red, sin costo y completamente legales.

Legal: Como ya dijimos en este capítulo, todos los programas libres ofrecen una licencia denominada GPL[48]GNU, que en sus siglas en ingles significa Licencia General Pública de GNU[49]. Dicha licencia pretende asegurar que el software no represente una limitante para quien lo use, para ello, debe de respetarse las 4 libertades del S.L. que también ya mencio-

namos. Este contrato creado por la Free Software Fundation, hace que el S.L también contenga una licencia pero que ésta, más que restringir su uso, copia en otras computadoras, distribución, cambio o mejora, permite que esas libertades se respeten en todas y cada una de las versiones que se realicen de dicho software, esto es, como ya dijimos, basado en el copyleft. El copyleft básicamente surgió de la necesidad de que el S.L no se convirtiera a lo largo el software privativo, así pues:

"La idea fundamental del copyleft es que se autoriza la ejecución del programa, su copia, modificación y distribución de versiones modificadas, siempre que no se añada ninguna clase de restricción a *posteriori.* De este modo, las libertades cruciales que definen el software libre quedan garantizadas para cualquiera que posea una copia; estas libertades se convierten en derechos inalienables". [50]

"La idea fundamental del copyleft es que se autoriza la ejecución del programa, su copia, modificación y distribución de versiones modificadas, siempre que no se añada ninguna clase de restricción a posteriori. De este modo, las libertades cruciales que definen el S.L. quedan garantizadas para cualquiera que posea una copia"

Esto ofrece una ventaja importante en el terreno de la creación pues al usar software libre podemos estar seguros de que nuestra herramienta de trabajo está dentro de los marcos legales, lo cual es muy útil al momento de hacer una distribución digital masiva, por ejemplo.

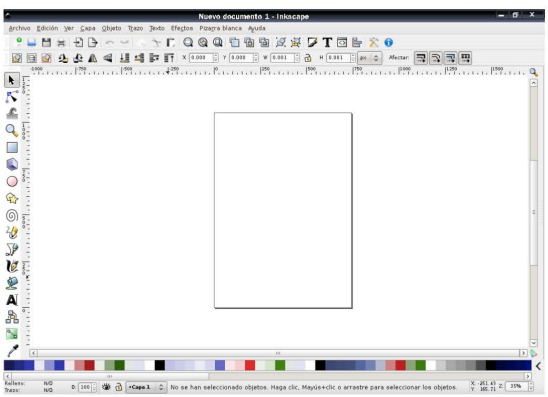
Fácil de conseguir: Para poder acceder a un programa de edición de imágenes libre es necesario sólo una conexión a internet, bajarlo, instalarlo y comenzar a utilizarlo, de igual forma el disco de la distribución de Ubuntu que por lo general se distribuye gratuitamente, contiene no sólo el sistema operativo GNU/LINUX, sino la paquetería de oficina y de gráficos, lo que nos permitiría, de tener dicho disco, operar en cualquier lugar que nos ofrezca un equipo de computo para trabajar. Existen además distribuciones de GNU/LINUX portables, es decir que caben en un usb de poco mas de 1 giga y que pueden correr en cualquier computadora si le indicamos que cargue el sistema operativo desde dicho usb, (hacer un booteo) sería como tener una navaja suiza o nuestra propia distribución portátil para usar en cualquier equipo, sin necesidad de recurrir a las copias ilegales con algún distribuidor no autorizado lejos de nuestro lugar de trabajo o la compra de una licencia que exceda nuestras posibilidades económicas.

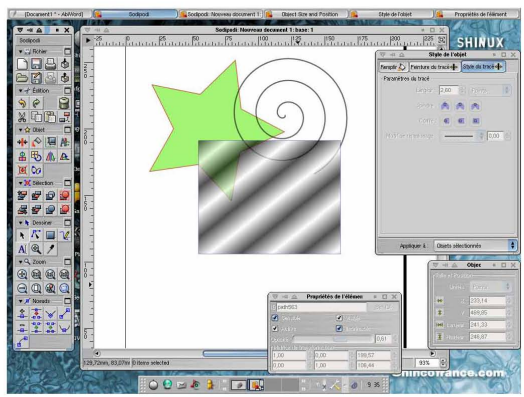
Para ejemplificar esto, hablaré de un caso anecdótico. Durante casi toda mi carrera universitaria viajé constantemente a Tulancingo Hidalgo donde viven mis padres, más de una ocasión necesité hacerle modificaciones a un archivo realizado en Photoshop o Illustrator, cosa que me fue imposible debido a que no contaba con equipo portátil en aquel entonces y en

aquella región de Hidalgo las computadoras de los café Internet apenas y tenían Microsoft Office instalado. Lograr comprar una copia no autorizada del software en aquella pequeña ciudad, era también imposible, pues el uso de estos programas aún no era generalizado en aquella región; pedirlo prestado menos, por que ni siquiera los diseñadores tulancinguenses utilizaban Illustrator, sino Corel en su versión 10. En aquella época, mas de una ocasión fue complicada la tarea de modificar mis archivos a distancia. El software libre, eventualmente solucionaría éstos y otros problemas de accesibilidad a una herramienta de trabajo.

Se trata con personas que desarrollan y no con compañías que crean un producto, no fomenta el monopolio: El software libre es de todos, no le pertenece a ninguna compañía o individuo, es posible que cualquier persona con conocimientos de programación pueda llegar al fondo de algún problema y resolverlo sin tener que recurrir a las soluciones de algún proveedor del programa o al llamado "soporte técnico" que lo provee una sola empresa, pues se esta tratando con "personas" que desarrollan el software. La característica de que sea libre y abierto y por lo tanto, accesible a todos, podría darle un giro humanístico a la tecnología, pues se trata con personas (aunque también pueden ser corporaciones y/u organizaciones) que lo desarrollan o implementan soluciones y no con compañías que venden productos. Esto nos hace incluso dejar de contribuir al crecimiento de los monopolios, acercarnos a lo que podríamos llamar "consumo responsable" en el rubro de herramientas tecnológicas, ya que contrario al S.P. no se beneficia a una compañía o a un monopolio en particular, sino a una comunidad. La posibilidad de generar un software que puede ser modificado por todos da la oportunidad de que se generen los llamados FORKS[51]que son los rumbos distintos que se generan de un programa que ya no se sigue desarrollando, o bien, que no cumple con las expectativas de una comunidad. Por ejemplo, el caso de Inkscape es un fork de un programa anterior llamado SODIPO-

A la izquierda, captura de pantalla del programa vectorial libre Inkscape que es n "fork" de Sodipodi, a la derecha.





DI que tomó una bifurcación en manos de una comunidad con necesidades específicas de diseño.

Respecto a esto, Christian Oyarzún comenta:

"(el software libre) permite también bajar una figura de autoridad que pasa con las tecnologías propietarias, es decir, "la marca". Desaparece la marca y aparecen otros colectivos o investigadores que desarrollan problemas específicos y comienzas a ver muchas formas de resolver problemas o problemas que nunca antes habías visto"[52]

En este mismo sentido, cuando se usa S.L., invariablemente el rol de usuario-cliente que se tiene con el S.P. se transforma en un rol mucho más amplio y participativo, alejado de la pasividad, pues es posible conformar la comunidad que es parte del proceso del desarrollo del software, entendiendo de cerca la relación entre código, soluciones, herramientas y medios, software y resultados, tal como se menciona en FLOSS + ART:

"La diferencia principal entre usuarios agrupados como clientes y usuarios que forman parte de una comunidad de S.L., es que estos últimos tienen la capacidad de ser pro-activos en el proceso de desarrollo de software a través de la escritura de documentación, apoyo, información y en algunos casos participación en la escritura del software.[53]"

En entrevista, Guillermo Espertino, diseñador argentino dice:

"A mi me parece bastante importante el tema de la libertad y de poder involucrarse con el desarrollo. Tener un error en el programa y poder hablar personalmente con quien puede arreglarlo, eso me parece insuperable y nunca habrá esa posibilidad en el software privativo"[54]

Todos para uno y uno para todos: Por último y no menos importante, al usar S.L. desde la perspectiva de un diseñador o un artista visual,contribuiría a que las siguientes versiones de programas se mejoren de acuerdo a nuestras necesidades o a la deficiencia de las anteriores versiones, algo que nadie más que un usuario especializado en imagen podría realizar; estaríamos contribuyendo a que se realicen cambios que beneficien a todos y cada uno de los usuarios de S.L. en el mundo, sin ninguna excepción o limitación. Como lo mencioné en el rubro anterior, se beneficia a una comunidad y es donde entra el dicho de que si uno avanza, los demás también lo hacen.

Guillermo Espertino, diseñador argentino usuario de software libre, en entrevista. Rosario Argentina, mayo 2011

Al respecto, Guillermo Espertino comenta que algunos de sus diseños fueron incluidos en las recientes versiones de Inkscape, más específicamente, los gráficos de los punteros del mouse al cambiar de herramienta: "Como diseñador aparte tienes la posibilidad de aportar tus diseños a un programa que tiene distribución mundial y son tuyos". Existen también repositorios en linea de plug ins para Gimp, el programa libre de manipulación de mapa de bits, creados por la comunidad de usuarios y no por una instancia en particular o una empresa privada: http://registry.Gimp.org [A].

Aunque también es cierto que el S.L. cuenta con una retroalimentación por parte de los usuarios, dichos "feedbacks" difícilmente se reflejarían en cambios al programa que a la propia compañía no le convengan o le representen algún tipo de pérdida, como hacerlos más interoperables con las versiones anteriores, o abrir archivos nativos de software libre o traducir el programa en alguna lengua minoritaria.

Sobre esto, Martín Eschoyes, artista digital y diseñador gráfico argentino, comenta:

"Con el software libre hay un **feedback** constante entre diseñadores y programadores, intercambio de ideas, construir sobre otras cosas que ya han construido otros, eso es un potencial que no tiene ningún otro software"[55]

Martín Eschoyez, diseñador y artista argentino, en entrevista. Córdoba Argentina, Junio 2011

La oportunidad de no lidiar con dueños que restrinjan los derechos del software como obra intelectual permite que la participación se canalice y se generen una cantidad más grande de programas derivados de otras iniciativas. Un ejemplo es el Libre Office, una bifurcación de Oppen Office, la opción de paquetería ofimática libre, o el Gimpshop, una bifurcación de Gimp que pretende tener más similitudes con el software privativo Photoshop. Todos los ejemplos de forks en el S.L. son una muestra de como una comunidad hace virar un proyecto en una dirección distinta a la de los programadores principales.

Lila Pagola comenta al respecto: "La comunidad del software libre me representa una influencia impactante. Sobre todo ha cambiado mucho mi modo de trabajar desde que la conozco. Una comunidad internacional organizada, unas formas de hacer posible esas colaboraciones. Me parece un modelo interesante y potente para replicar"[56]

Nuevamente, Martín Eschoyez da su opinión respecto a este tema:

"Me parece que fomenta la creatividad (el software libre) por que te sientes con la necesidad de compartir todas esas cosas y descubrir que eso es mucho más poderoso que quedarte las cosas y no compartirlas con nadie, esta idea de que el diseñador y/o el artista es súper elitista e individualista; en la filosofía del S.L. no es así."[57]

2.2.2 Problemáticas de la adopción delsoftware libre en la creación gráfica digital

A) GENERALIDADES:

Se necesita de asesoría al empezar: Para explicar este punto hablaré de mi experiencia personal como diseñadora y mis nulos conocimientos con respecto a programación. Cuando decidí iniciarme en el S.L. pedí me fuera instalado una partición de Windows con GNU/LINUX, desde entonces he necesitado eventual asesoría para instalación de programas, fuentes y solución de problemas en cuanto a tratamiento de imágenes se refiere, y aunque muchas cosas he podido resolverlas sola con ayuda de los foros en linea y la experiencia adquirida a lo largo de un par de años, también es cierto que aún no conozco todas las posibilidades de usar la terminal de mi ya no tan nuevo sistema operativo, recurso utilizado mucho más que en otros sistemas y que permite resolver problemas de manera más rápida y eficaz.

Sin embargo, esto no es ajeno a otros sistemas operativos como Windows, ya que es muy probable que cuando nos iniciamos en su uso, hayamos pedido ayuda al menos para instalar y comenzar a utilizarlo o incluso, aún lo hagamos. Adaptarnos al programa y aprender a manipularlo también nos llevó tiempo, años y posiblemente, ayuda de terceros.

Curva de aprendizaje:Una vez que aprendemos a usar un sistema operativo y nos acostumbramos a él, es difícil al principio reconocer y acoplarnos a las características de otro, esto no pasa sólo al momento de migrar a GNU/LINUX, sino cuando alguien acostumbrado a usar MAC, pasa al sistema operativo de Microsoft e incluso es probable que le escuchemos decir más de una vez: “odio Windows”.

En el caso del S.L.,algunas personas opinan que la curva de aprendizaje en mayor, esto puede ser debido a muchas cosas, una de ellas que los programas libres, debido a la poca difusión que tienen en el “mercado”, las usan mucho menos personas que el S.P. Incluso, hay personas que asocian al sistema operativo Windows con todo lo que tenga que ver con computadoras. Es el sistema que se usa en el gobierno mexicano y para el cual los empleados reciben capacitación y el que además eventualmente se enseña en las escuelas. Es por eso que cuando se pretende incursionar en cualquier otro sistema operativo o cualquier otro pro-

grama, es preciso desaprender lo anterior. El software privativo definitivamente es más popular y más usado en computadoras de escritorio que el libre y el cambio implica siempre un nuevo aprendizaje, un nuevo entendimiento de una manera diferente de resolver cosas, un nueva forma de manejo de usuarios, de archivos, de instalar programas, etc.

José Santorcuato, artista electrónico y académico menciona: "Uno de los impedimentos para que el artista se acerque al software libre es la curva de aprendizaje, sin embargo poder replicar su obra en cualquier situación y cualquier entorno, es una recompensa"[58]

Sin embargo, Fabricio Caiazza, artista plástico argentino que además da clases de iniciación artística con software libre a niños, comenta:

"El mayor conflicto lo tienen las personas que ya tienen una experiencia previa con la manipulación de software privativo, como Photoshop, encuentran que en Gimp las herramientas están ubicadas en otro sitio y se desconciertan. Esto no ocurre con los niños (a los que les enseña) que su primer acercamiento con un software para manipular imágenes es justamente el Gimp, ellos trabajan con naturalidad"[59]

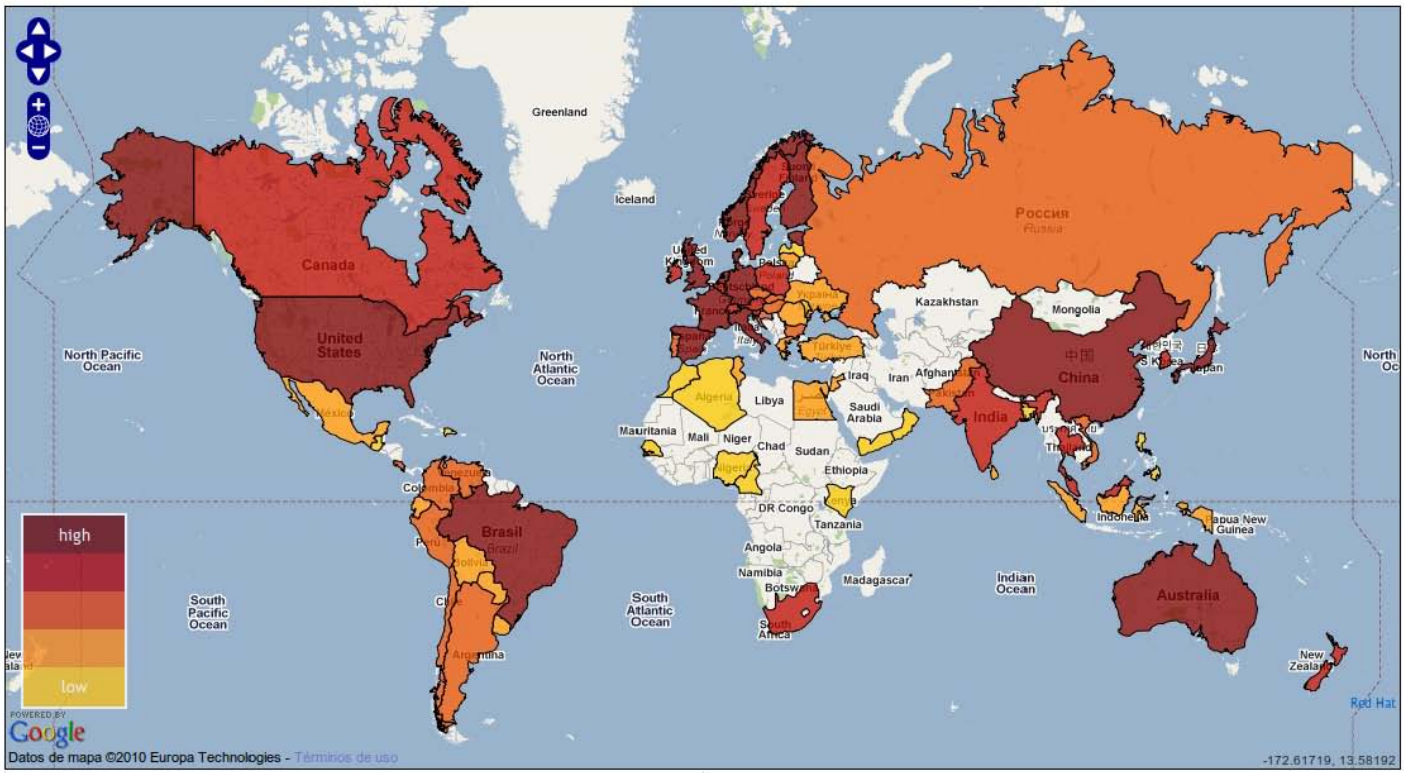
No lo conocen en imprentas ni casas de impresión digital: A pesar de que en muchos países europeos como Alemania, Francia y España y algunos de América como Brasil y Argentina, cuya población esta familiarizada con el software libre y su uso, en el caso de México no pasa igual. En el mapa mostrado a continuación, realizado por el Georgia Tech/Red Hat Open Source Software Potential Index project [60]se puede ver claramente la actividad relacionada con el uso del software libre en el mundo.

México mantiene una actividad aún baja con respecto a otros países, por lo que su popularidad es igualmente baja, más en ciertos sectores, como el diseño y el arte, lo cual hace suponer que somos todavía una minoría de diseñadores y artistas los que trabajamos con estas herramientas libres y por lo tanto, las casas de diseño y las imprentas no cuentan con estos programas a pesar de la facilidad de descarga y de su instalación, además de que les son desconocidos. Eso, aunado a que por confundirlo con software "gratis", la gente piensa que es de baja calidad o que es imposible desarrollar diseños de calidad con él.

Sin embargo, esto nos obliga a pensar nuevamente en la utilización de estándares al momento de realizar nuestros archivos. Mientras las casas de impresión requieren tener mas de un programa privativo (en su mayoría en versión no autorizada) para abrir los archivos guardadas en la extensión nativa de los diferentes programas existentes y en los que decida guardarlo el cliente, (generalmente, diseños pedidos en formato .AI, .PSD o .CDR), en su lugar podemos usar formatos PDF o SVG para que los abra cualquier programa.

Instalación de Hardware: En algunos casos, se requiere de más tiempo para instalar ciertos dispositivos de hardware, como tabletas de dibujo, cámaras, scanners o impresoras. Esto es por que algunos de estos dispositivos están fabricados para que funcio-

Open Source Activity Map



Mapa de actividad del software de código abierto en 2010. Recuperado en Junio del 2011, en: http://www.redhat.com/about/where-is-open-source/activity/

nen en los sistemas operativos más comunes que utilizan controladores de código cerrado, lo cual complica el desarrollo para la compatibilidad con el S.L.. Sin embargo, siempre existe la manera de "parchar" estos problemas, sólo que requiere de más tiempo y paciencia por parte del usuario. Por ejemplo, un músico que recién compra un piano digital requiere que su tarjeta de audio que corre dentro de un sistema operativo libre, reconozca su nuevo piano. La compañía de fabricación del piano digital ha decidido no implementar controladores para GNU/LINUX pues no les representa una comunidad a la cual atender. Por tal motivo, el músico tendrá que visitar foros, pedir asesoría y hacer un esfuerzo extra para que su piano reconozca la tarjeta de sonido. Sin embargo esto resulta muy relativo, ya que, cuando yo compré la impresora en la cual imprimí esta tesis, me aseguré antes de comprarla, que fuera compatible con GNU/LINUX. Cuando la conecté a mi equipo, no tuve que instalar absolutamente nada, sólo la conecté por USB y comencé a imprimir, sin instalar discos, ni bajar controladores. Pero esto no siempre es así. Mucho de esto de debe a lo que algunos llaman trusted computing o computación de confianza, en el que supuestamente podremos confiar en las compañías que generan tecnología pues nos aseguramos que nos proveen de equipos confiables. Sin embargo en el video Trusted Computing[61], sus autores se cuestionan si esto es realmente cierto, ya que, en teoría, una computación de confianza se basa en una reciprocidad, donde compañía y cliente confían entre si. Los usuarios, deberían de ser capaces de decidir qué es bueno para ellos y sus necesidades y la compañía respetar estas ideas y protegerlos. Esto no ocurre así pues la compañía por lo general decide no confiar en el usuario y que no podrá resolver ciertas cosas por él mismo, o bien, decide no dar soporte cuando se trata de ciertos sistemas operativos. Entonces, si la compañía no confía en los usuarios, ¿los usuarios deberían confiar en la compañía que decide por ellos?

El software libre posee muchas distribuciones las cuales representan variadas opciones según las necesidades de cada usuario.

B) PARTICULARIDADES:

Muchas distribuciones que generan confusión: El software libre posee muchas distribuciones[62]las cuales representan variadas opciones según las necesidades de cada usuario que además se actualizan constantemente, y aunque son adaptables para cualquier tipo de usuario y necesidad, esto para un usuario novato, puede causar cierta confusión al no saber que "distro" le conviene más. Para ello se necesita de la asesoría de algún usuario experimentado, de la búsqueda en Internet de experiencias de usuarios y/o desarrolladores o bien,

preguntar en los múltiples foros que se dedican al tema. Cada distribución tiene características muy especificas que las hace útiles para algunos casos, hay versiones mas estables que otras y otras más amigables. A un usuario novato se le recomienda el uso de UBUNTU, que hasta el 2011 liberó su versión 11.04 Beta, la cual tiene una interfaz gráfica de usuario bastante amigable. También existe UBUNTU STUDIO enfocado a diseñadores y artistas que trabajan con edición de sonido, video y gráficos 2D y 3D y Artix3D enfocada a la producción multimedia.

Aunque las múltiples opciones de distros del software libre son una ventaja en cuanto a satisfacer las necesidades a diversos números de usuarios, también es cierto que puede existir confusiones en usuarios que inician, así como puede resultar un inconveniente la solución de problemas específicas de cada distribución.

Desarrollo de los programas: Inkscape tiene aproximadamente 7 años de desarrollo, y ha logrado ser una herramienta poderosa en cuanto a edición de vectores se refiere, sin embargo, es evidente que no tiene el grado de desarrollo que posee la actual versión CS5 de Adobe. En muchas ocasiones, el poco desarrollo que tenga algún programa libre se debe a que no existe una comunidad que demande dicha mejora, esto es, entre menos diseñadores haya utilizando Inkscape, menos posibilidades existen de que se reconozcan los errores o problemas que represente para la creación, ya que sólo un especialista en esto, puede detectarlo en el uso cotidiano. Por otro lado, aunque no es postura de esta tesis "comparar" programas libres versus privativos, es cierto que al migrar de un software a otro, las comparaciones no siempre se pueden dejar de lado.

En Gimp no existe aún una manera eficaz y rápida para deformar una imagen de tal forma que ésta se adecué a una superficie ya deformada, el "WARP"

En mi uso cotidiano del S.L. para el diseño, he encontrado que existen herramientas para las cuales resulta más complicado la resolución de algunos problemas, por ejemplo, en el caso de Gimp, no existe aún una manera eficaz y rápida para deformar una imagen de tal forma que ésta se adecué a una superficie ya deformada, el llamado WARP en Photoshop. Para ello, se tiene que recurrir a deformarlo en perspectiva y luego el filtro de deformar por curva, pero resulta muy tardado pues no se tiene un control suficiente a dicha deformación.

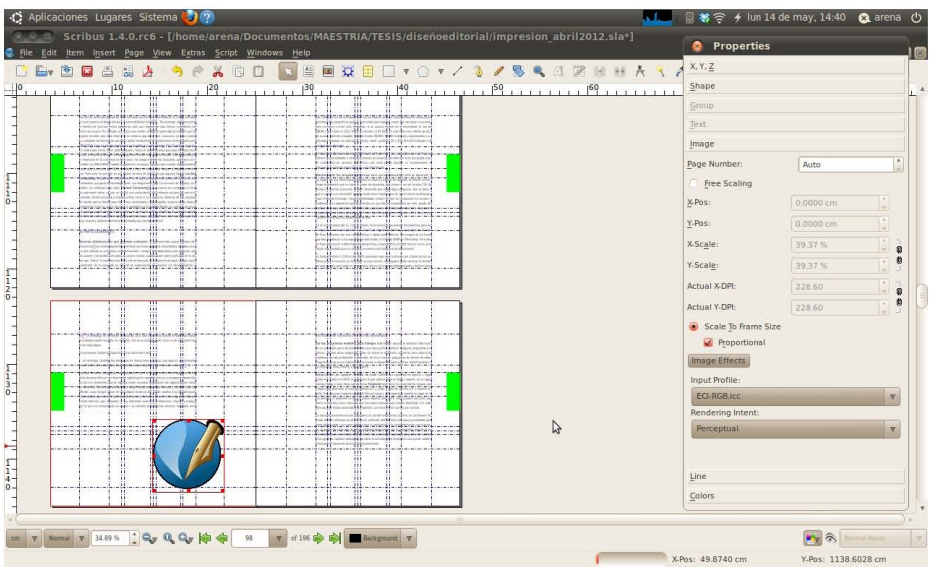
En Scribus[63] versión 1.3.6.svn del 2009, resultaba engorroso la división por sílabas de las palabras, pues al incrustar un texto que ya esta escrito, el programa pedía verificar la división de cada palabra, para saber si es correcta. Esto a la larga resultaba cansado y hasta moles-

to[64]. Sin embargo, en la versión Scribus NG 14.0, con la que está siendo armada esta tesis, el problema quedó resuelto, sin embarbo, aún no es posible poner notas al pie en la parte baja de cada página.

En entrevista, Guillermo Espertino opina sobre este tema:

"...sin embargo, también hay problemas de limitaciones técnicas, que algunos son bastante importantes y hay que hacer un gran esfuerzo para superarlos y otros son menores."[65]

Hay que tomar en cuenta que la comparación entre un programa comercial que lleva varios años de desarrollo con un importante capital que lo respalda, versus un programa desarrollado por una comunidad, que en algunos casos no posee financiación (en algunos otros casos si, donde las industrias invierten en un mejor desarrollo que las favorezca, como el caso de Blender o que tengan apoyo de sus propias fundaciones) puede resultar muy desventajosa. Cabe señalar que tanto el software libre como el software privativo son programas radicalmente distintos que responden a muy diferentes maneras de desarrollo, difusión y trabajo, por lo que una comparación en cuanto a la solución de problemas técnicos resultaría también insuficiente, inconclusa y una vez más, desventajosa.



Captura de pantalla de Scribus NG 14.0

No hay programas estables para trabajar con video: Aunque el software libre cuenta con un amplia gama de posibilidades que bien podrían sustituir a algunos programas privativos, algunos otros programas libres no tienen el desarrollo suficiente para solucionar problemas de tipo profesional o avanzado, tal es el caso de programas de edición de video. Programas libres como Kdenlve aún no tienen el desarrollo para sustituir definitivamente a un programas como Premier o After Efects.

Gimp carece de soporte modelo de color CMYK: La capacidad de separar y luego editar una imagen en CMYK, es algo para lo que todavía Gimp no tienen soporte, en su lugar, existe un plug in para Gimp que se llama *Separate+*que tiene la capacidad de separar una imagen RGB, gestionar el color en perfiles ICC y LCMS y adjuntar perfiles ICC a nuestro diseño. Gimp tampoco soporta mas de 8 bits por canal y solo soporta 3 canales en lugar de 4. Sin embargo, el programa Scribus si posee soporte para CMYK. Hay usuarios que dicen que tanto como Gimp como Inkscape son una buena solución para diseño destinado a la web, pero que para diseño destinado a la impresión, aún tiene mucho camino por recorrer.

Es así como encontramos una alternativa al uso del software de patente, al cual hemos llamado también software privativo. Con el software libre se abre una nueva posibilidad para crear digitalmente y un nuevo terreno para la exploración y la experimentación; por supuesto, sin perder de vista que, al ser un software desarrollado por la comunidad, requiere e incluso depende de la estrecha interacción de los usuarios para generar programas de calidad. El el siguiente capítulo hablaremos de cómo el software libre representa una opción viable y eficaz para el desarrollo de la actividad profesional.

1Richard Stallman es considerado el padre del Software Libre por ser el que acuñó el concepto y se lo otorgó a los sistemas operativos y programas que cumplieran con las 4 libertades que aseguran la libertad para los usuarios.

2Sam Williams **Libre como en libertad,** (Madrid, O'Reilly & Associates, Inc, 2002) 6.

3Williams, "Libre como..." 6.

4Williams, "Libre como..." 9.

5Williams, "Libre como.." 10.

6 "Visión general del proyecto GNU", **Historia del proyecto GNU**, (Enero 2008, [citada el 6 de junio del 2009]), GNU Opening Systems. Disponible en http://www.gnu.org/gnu/gnu-history.es.html

7A lo largo de esta tesis, nos referiremos al Software Libre, ya sea con su nombre completo o con las siglas SL.

8 Richard Stallman **Software libre para una sociedad libre,** (Madrid, Traficantes de sueños, 2004) 47.

9 Neil McAllister "Devices provide a fertile new ground of Linux", **Computerworld. (**18-Sep-2006 [citado el 7 de Julio del 2009]) disponible en: http://computerworld.co.nz/news.nsf/tech/2E968937ADA351D8CC2571EA0013F01B

10Erick Raymond **La catedral y el bazar** (Argentina, Openbiz,1997) 16.

11 Stallman, "Software Libre," 59.

12CDS cuyas siglas en ingles significan Counterfeit Deterrence System, que podría traducirse como un Sistema de Disuasión de Falsificaciones.

13Adobe, "Photoshop and CDS" **Adobe Photoshop CS5** (Julio 2009 **[citado el 8 de agosto del 2011]) disponible en:** http://www.adobe.com/special/products/Photoshop/cds.html [2].

14Brad Stone, "Amazon Erases Orwell Books From Kindle" **The New York Times,** (17 de Julio 2009 [citado el 8 de agosto del 2011]) Sección: Companies, Technology, disponible en: http://www.nytimes.com/2009/07/18/technology/companies/18amazon.html [3]

15DRM Digital Rights Management, termino para designar a las tecnologías de control de acceso para limitar el

uso de contenidos digitales y que impide el uso del contenido digital que no es deseado o pretendido por el proveedor de contenido.

16José Flores, "Mozilla Firefox, traducido a lenguas prehispánicas" **Alt1040** (16 de Agosto de 2010 [citada en Julio del 2011]) ed. Hipertextual, disponible en: http://alt1040.com/2010/08/mozilla-firefox-traducido-a-lenguas-prehispanicas

17En iniciativas como Runasimitpi Qespisqa Software http://runasimipi.org [4]

18Ralph Levien, "Ralph Levien's Thesis" Sitio web personal de Ralph Levien (Septiembre 2009 [citada en Julio del 2011]) ed. Ralph Levien, *d*isponible en: http://www.levien.com/phd/phd.html

19Inkscape es un software libre para la creación y edición de vectores, un homólogo a Illustrator o al Corel Draw.

20FontForge es un programa libre para editar fuentes tipográficas.

21Ralph Levien, "Spiro" Sitio web personal de Ralph Levien (Septiembre 2009 [citada en Julio del 2011]) ed. Ralph Levien, *d*isponible en: http://www.levien.com/spiro/

22Ralph Levien, "Fonts in progress" Sitio web personal de Ralph Levien (Septiembre 2009 [citada en Julio del 2011]) ed. Ralph Levien, *d*isponible en: http://www.levien.com/type/myfonts/

23Lawrance Lesing **Cultura Libre**, Trad. Antonio Córdoba/Elástico (E.U. Penguin Books, 2004)

24Linux Devices, "SFLC, BusyBox, Monsoon dismiss GPL lawsuit" **Eweek Linux Devices** (30 de octubre del 2007 [citado en Agosto del 2011]) ed. Linux Devices, dispobible en: http://www.linuxfordevices.com/c/a/News/SFLC-BusyBox-Monsoon-dismiss-GPL-lawsuit/ [5] recuperada en agosto del 2011.

25Stallman, "Software libre," 63.

26El concepto de "autor" separado del de "dueño". Donde el autor no pierde el crédito por la obra pero tampoco se "adueña" de ella para restringir su posterior uso, mejora, estudio y distribución.

27Lesing, **Cultura Libre**, 34.

28Stallman, "Software Libre," 22.

29 Stallman, "Software Libre," 133.

30 Stallman, "Software Libre," 135.

31A partir de ahora, para referirnos al él, usaremos las palabras software privativo o bien sus siglas S.P.

32 Stallman "Software libre para," 13.

33Stallman "Software libre para," 13.

34 Stallman "Software libre para," 283.

35José Serralde, "Opera, norteños y software libre (I)" Sonoblog, música, tecnología, cultura libre y sonidos por dentro, (10 de marzo del 2010 [citado en septiembre 2010]) ed. José Serralde, disponible en: http://blog.joseserralde.org/opera-video-nortenos-y-software-libre

36Ton Roosendaal, "About" Sintel, the Durian Open Movie Proyect (citada en Agosto del 2011) Sintel ed. Blender Open Movie Proyect, disponible en: http://www.sintel.org/about/

37Eschoyez Martín, extracto de entrevista (Córdoba Argentina, Junio 2011)

38Mansoux y Valk Floss and Art. 13.

39"Illustrator CS5, especificaciones técnicas" Adobe (citada en Julio del 2011) disponible en: http://www.adobe.com/es/products/Illustrator/tech-specs.html

40GTK+ o The Gimp Toolkit es un conjunto de subprogramas que contienen código y datos y proporcionan servicios a programas independientes, es decir, pasan a formar parte de éstos. Esto permite que el código y los datos se compartan y puedan modificarse de forma modular. Este subprograma o biblioteca multiplataforma en particular sirve para desarrollar interfaces gráficas de usuario principalmente para los entornos gráficos GNOME, XFCE y ROX, aunque también se puede usar en el escritorio de Windows, MacOs y otros.

41"Open Standars" Document Freedon Day (Primavera 2010 [citado en septiembre 2010]) ed. FSFE, disponible en: http://documentfreedom.org/2011/os.en.html

42Root es el archivo raíz y tiene todos los derechos sobre el sistema. Todo lo que esta accesible dentro de Linux se encuentra en una subcarpeta de la raíz, se representa mediante /

43De user do, en español, el usuario hace.

44Una terminal en términos de computo es la abstracción generalizada, que comprende tanto una interfaz de programación de aplicaciones [6] para programas y un conjunto de expectativas de comportamiento para los usuarios,

45Greg Kroah-Hartman , Jonathan Corbet y Amanda McPherson , "Linux Kernel Development, How Fast it is Going, Who is Doing It, What They are Doing, and Who is Sponsoring It" (August 2009 [citado en agosto del 2010] The Linux Fundation) pdf dispobible en: www. [7]linux [8]foundation.org/sites/main/files/publications/whowrites [9]linux [10].pdf [11]

46"50 Places Linux is Running That You Might Not Expect" Focus (2011 [citado en diciembre 2011]) ed. Focus Editors, disponible en: http://www.focus.com/fyi/50-places-linux-running-you-might-not-expect/

47El precio de la Creative Suite CS5.5 Design Premium en 2011, del sitio oficial de Adobe para latinoamérica: http://www.adobe.com/mx/products/creativesuite/design.html

48General Public Licence

49Gnu is Not Unix, refirriendose a la base del sistema operativo linux que no tiene que ver con su antecesor Unix que después se volvió privativo.

50Stallman, "Software libre," 28.

51Un Fork, traducido al español significa bifurcación. Se trata de la creación de un proyecto en una dirección distinta de la principal u oficial tomando el código fuente del proyecto ya existente. Comúnmente se utiliza el término inglés.

52Oyarzún Christian, extracto de entrevista (Santiago de Chile, Mayo del 2011).

53Aymeric Mansoux y Marloes de Valk , "Prefacio" en Floss and art ed. Aymeric Mansoux y Marloes de Valk (Francia, GOTO10 y OpenMute, 2008) 10.

54Espertino Guillermo, extracto de entrevista (Rosario, Argentina en Mayo 2011).

55Eschoyez Martín, extracto de entrevista (Córdoba, Argentina en Junio 2011).

56Pagola Lila, extracto de entrevista (Córdoba, Argentina, Mayo del 2011)

57Eschoyez Martín, extracto de entrevista (Córdoba Argentina en Junio del 2011).

58Santorcuato José, extracto de entrevista Santiago de Chile, Mayo 2011).

59Caiazza Fabricio, extracto de entrevista, (Rosario, Argentina, Mayo 2011).

60Red Hat, "Open Source Activity Map – 2008" (2009 [citada en octubre2009] Red Hat) ed. Red Hat Enterprise Linux disponible en: http://www.redhat.com/about/where-is-open-source/activity/ [12]

61Benjamin Stephan y Lutz Vogel "Trusted Computing" MPG4. Dirigido por: Benjamin Stephan y Lutz Vogel (E.U.:LAFKON, 2005). Disponible en: http://www.lafkon.net/tc/

62Las llamadas "distro" entre la comunidad del Software Libre que son las diferentes versiones de distemas operativos, lo que en caso de Microsoft sería los conocidos Windows xp o Windows vista y en Mac, el Leopard o el OS X

63El programa libre para la diagramación y diseño de paginas de libros y revistas, pertenecientes a los llamados desktop-publisher, o bien, la versión libre de Indesing.

64Para términos del diseño editorial de la presente tesis, comencé a experimentar con Scribus y comienzo con su descubrimiento, hasta el momento he encontrado esa deficiencia, aunque sigo trabajando en ello.

65Espertino Guillermo, extracto de entrevista Rosario, Argentina, mayo 2011.

Links externos

[1] http://registry.gimp.org/

[2] http://www.adobe.com/special/products/photoshop/cds.html

[3] http://www.nytimes.com/2009/07/18/technology/companies/18amazon.html

[4] http://runasimipi.org/

[5] http://www.linuxfordevices.com/c/a/News/SFLC-BusyBox-Monsoon-dismiss-GPL-lawsuit/

[6] http://translate.googleusercontent.com/translate_c?rurl=translate.google.com&sl=auto&tl=es&twu=1&u=http://en.wikipedia.org/wiki/Application_Programming_Interface&usg=ALkJrhgXMgJ-LIt0IMmm8unImMchhedBCJg

[7] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[8] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[9] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[10] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[11] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[12] http://www.redhat.com/about/where-is-open-source/activity/

%%title:

Capítulo Tres

<title>

Software libre y procesos creativos

</title>

En este capítulo hablaremos de casos reales de profesionales usando software libre, comenzando por mi propia experiencia. El objetivo es ofrecer algunas pruebas de que el software libre puede ser usado por los profesionales de la creación gráfica digital sin que esto represente una limitante en su proceso, creación y difusión de su obra, así mismo, se explorarán las razones por las cuales se prefirió el uso de este tipo de software y los problemas que han enfrentado, todo esto a lo largo de una serie de entrevistas y documentación de anécdotas.

3.1. Mi experiencia con el software libre

En el prólogo de esta tesis, comenté brevemente cómo me acerqué al uso del S.L., además de mencionar en el capítulo 1 una de mis primeras experiencias usándolo de manera profesional. A continuación, describiré detalles de esa y otras experiencias.

A inicio del 2010, el presidente de la organización no gubernamental en México Preservación de Quelonios, A.C., David García Picazo, me encomendó la elaboración de un cartel que sería exhibido por el Gobierno del Distrito Federal en algunas estaciones del metro y parabuses de la Cuidad de México para dar a conocer al público la exposición de Tortugas (o quelonios) en el túnel de la ciencia del metro La Raza, linea 5 del STC.

La elaboración de dicho cartel se realizó en un 100% con software libre. Lo primero que realicé, una vez que tenía el boceto previo en papel, fue el retoque digital, manipulación, recorte y escala de las imágenes que la A.C me proporcionó. Para ello, utilicé el programa libre para manipulación de mapa de bits, llamado **Gimp.** Después, comencé con la composición gráfica, en donde se incluyó la tipografía y se le dió salida para impresión. En ese caso, utilicé el programa libre para edición de vectores, llamado **Inkscape.** Ambos,dentro del sistema operativo GNU / Linux, con la distribución Ubuntu Lucid 10.04 de 32bits.

Mi proceso de producción se demoró un poco más de lo acostumbrado cuando usaba los programas privativos Illustrator y Photoshop, debido a mi costumbre de usarlos y al tiempo invertido en la familiarización de la diferente interfaz de los programas libres. Recurrí a la consulta de algunos foros en Internet y a preguntar a usuarios con mayor experiencia en Inkscape, sobre algunos métodos para resolver tal o cual procedimiento.

Después de unos cuantos frustrados intentos, pude comenzar a reproducir digitalmente el boceto que había realizado con anterioridad, pudiendo recrearlo de manera exitosa al cabo de unos días.

Ciudad de México
Capital en Movimiento
Gobierno de la Ciudad de México
Diseño Gráfico: IRENE SORIA GUZMÁN Fotografía: DIANA GABRIELA GARCÍA PICAZO
Quelonios...
Una pequeña ventana al mundo de los reptiles
Ven, conoce y aprende más acerca de nuestras amigas las tortugas o quelonios y demás miembros exóticos de esta gran familia en la exposición viva permanente en el Túnel de la Ciencia, estación metro La Raza, línea 5 de lunes a domingo en los horarios de servicio del STC Metro.
Preservación de Quelonios A.C.
Presidente: Mtro. David Elías García Picazo
Exposición Viva Permanente
SISTEMA DE TRANSPORTE COLECTIVO
Tortuga Leopardo
imagen institucional de
Preservación de Quelonios A.C
http://preservaciondequelonios.hi5.com

A lo largo del proceso, fui resolviendo el problema del desconocimiento de la interfaz pero me iba enfrentando a otros más, como por ejemplo, la inserción del logo de la Secretaria de Turismo del Distrito Federal, que se encontraba vectorizado en formato EPS y que contenía un degradado de color. Inkscape no reconoció el degradado y lo importaba como si fuese un logo con un solo color plano.

Resolví el problema convirtiendo el archivo de curvas a un PNG, también en el propio Inkscape, para que respetara la transparencia y el degradado a la vez, el problema quedó resuelto y el cartel entregado a tiempo.

Se imprimieron 214 piezas para las diversas estaciones del metro que se exhibieron del 1 al 31 de febrero del 2010. 100 carteles fueron colocados del 9 al 18 de febrero en las paradas de parabus en toda la ciudad.

Posteriormente, David García Picazo me pidió el cambio de un par de imágenes en el mismo cartel. Esta experiencia me permitió comprender por primera vez las ventajas de trabajar con un archivo estándar y cómo éste puede ser leído sin necesidad de un "programa" determinado y único para esa labor, es decir, al tener el archivo en SVG que es un formato estándar, pude abrirlo con un simple procesador de texto y ver como todos los elementos que formaban el diseño estaban siendo descritos en el formato XML (metalenguaje extensible de etiquetas) que eventualmente podría ser descifrado por distintos programas, incluso, un navegador de Internet o un editor de texto simple. Esto me llevó a comprender, que si dejase de existir la herramienta digital que en ese momento usaba, no determinaría que el diseño pudiera verse; es decir, que si un día Inkscape desapareciera, dejara de tener soporte o, simplemente, un día no estuviese en mi computadora, habría una manera de conocer el contenido del archivo generado en él y guardado en formato estándar. El archivo SVG no se trata de un archivo escrito en un lenguaje codificado suscrito a la interpretación de un sólo y único sistema, no se trata de un archivo "secreto" que únicamente puede ser decodificado por un programa. Redireccioné el llamado de las imágenes por medio del editor de texto simple pa-

ra evitar abrir el programa de edición de vectores y cambiarlas manualmente, sin embargo, también es cierto que sin problemas hubiese abierto Inkscape para hacer todo el proceso cambiando las imágenes una a una. Abrir el archivo desde el editor de texto, paradójicamente, hizo que el proceso de cambio de imágenes fuera más rápido de lo esperado.

Meses después, para la exposición del Mes de la Tierra en el parque ecológico Xcaret, Preservación de Quelonios tuvo una participación importante con la exposición fotográfica "Quelonios, una pequeña ventana al mundo de los reptiles" para lo cual elaboré las cédulas de identificación de las fotografías que ilustraban el catálogo de especímenes que protege esta Asociación Civil.

De igual forma, dichas cédulas fueron elaboradas con Inkscape en su totalidad y usando también Gimp para el retoque digital de algunas fotografías. Fueron enviadas para su impresión en formatos PDF y JPG y dicho proceso no representó ningún problema.

En ese mismo año, hice un diseño en gran formato, una lona de 2 por 1.5 metros para la Instancia de la Mujer a través de la Coordinación de Desarrollo Social del municipio del Tulancingo en el Hidalgo, la cual fue colocada en el stand de "Mujeres Emprendedoras" en la Expo Tulancingo 2010. La lona se conserva para la promoción de dicha Instancia en otros eventos. De igual manera, usé Inkscape para la composición de gráficos y Gimp para recortar y retocar imágenes. Aquí cabe destacar, que la posibilidad de imprimir en gran formato, no requirió el uso de plantillas o medidas predeterminada de resolución; más bien, dependió del conocimiento previo sobre resoluciones para su impresión exitosa y de calidad.

Lona para La Instancia de la Mujer a través de la Coordinación de Desarrollo Social.

Página de la OFCM realizada en su totalidad con software libre y apegada a estándares, validada por la W3C

En el subsecuente trabajo como freelance, realicé algunos otros diseños cuyo soporte fue la web, tal fue el caso de páginas electrónicas para la Orquesta Filarmónica de la Ciudad de México (www.ofcm.m), el sitio personal del profesor y filósofo Enrique Bonavides (ebonavides.com.mx) y el sitio comercial de un local de manualidades (www.novedadesirene.com.mx). Así mismo, carteles que fueron realizados para su distribución digital, como el cartel para el 7mo Encuentro en línea de Educación, software y cultura libre (EDUSOL 2011) y el 1er encuentro de Cultura Libre en la Ciudad de México. En todos estos casos, el proceso de producción se llevó acabo 100% con software libre (Gimp e Inkscape nuevamente) y debido a su naturaleza digital, no hubo mayor inconveniente ni en su realización ni en su formato de salida (PNG y JPG) o en su distribución.

Sin embargo, cuando comencé a trabajar en la Red por los Derechos de los Niños (REDIM) a finales del 2011, justo mientras este trabajo de investigación se encontraba en la recta final, me topé con otras problemáticas que no se habían presentado antes y que tuvieron que ver con el tema de la impresión y el envío de archivos a la imprenta.

En la REDIM, me topé con problemáticas que tuvieron que ver con la impresión y el envío de archivos a la imprenta.

Primero, realicé el logo de la campaña "Infancia sin violencia" en Inkscape sin ningún inconveniente, al igual que la serie de postales para identificar y promover la compaña entre las organizaciones.

Posteriormente realicé dos infografías con el tema de los derechos de la infancia y los derechos de las poblaciones callejeras dentro del Programa de Derechos Humanos del Distrito Federal. Para la impresión de los 1000 ejemplares por cada infografía, la imprenta pidió que se enviara el archivo en CMYK y en capas combinadas. Aquí me topé con un nuevo inconveniente. Ni Inkscape ni Gimp poseen soporte nativo para generar un archivo CMYK, al menos no desde que se comienza a hacer el diseño, el llamado early binding (realizar nuestro diseño en un archivo en donde se elige desde el principio la modalidad CMYK) y es necesario recurrir a plugins para dicho fin.

Esto se debe, básicamente a políticas de desarrollo, de recursos y de arquitectura de Gimp, ya que el equipo de desarrolladores consideran que con un buen sistema de gestión de color no debería ser necesario soportar CMYK nativamente[1].

Esta traba fue solucionada a través de un plugin que instalé en mi sistema operativo Ubuntu Lucid 10.04 buscando repositorios en línea y que lo incluyeron automáticamente en las herramientas de color del Gimp. El nombre del plugin es **Separate+.** Lo que hice fue exportar un archivo en formato PNG de alta calidad desde Inkscape, donde realicé el diseño, lo abrí con el GIMP, invoque el plugin desde el menú "Imagen" y seleccioné el perfil origen de la imagen (sRGB) y el perfil de salida (en este caso SWOP, aunque el impresor no especificó el perfil de color que usaba), activé la casilla BPC y el psuedo-composite CMYK. Con ello pude obtener un archivo en formato TIFF en CMYK, lo guardé y lo envié a la imprenta. Particularmente este tema, requirió que hiciera una investigación mucho mas consensuada en cuando a color se refiere, esto es, tanto para el uso en lo digital y para la impresión, lo cual evidenció que algunas prácticas que damos "por hecho" (como el hecho de abrir un nuevo lienzo con un manejo de color CMYK desde el inicio) son, en gran medida, paradigmas cimentados

en las costumbres del software privativo. Sin embargo, para ampliar y acotar este tema en particular, me parece pertinente ofrecer en los anexos de esta tesis, un apartado dedicado a la "selección de color y tecnologías libres".

Más experiencias personales

Siguiendo con la descripción del trabajo para la REDIM, se me encomendó la elaboración de un par de videos[2] sobre el tema de las infografías ya realizadas y anteriormente mencionadas.

Dichos videos los realicé en colaboración con un editor; yo me dediqué a la parte de creación de gráficos, guión y diseño, mientras que mi colaborador lo hizo a la edición de imagen, audio y realización. Los gráficos 2D que aparecen en estos videos fueron realizados con el software libre Inkscape principalmente, los gráficos en 3D y parte de la edición fueron realizados con Blender. mientras que la música, algunas imágenes y algunos otros elementos utilizados fueron recursos libres bajo licencias permisivas Creative Commons que permitieron su uso. Sin embargo, la edición sustancial y el *render* se realizó con software privativo, dado que no existe un software libre lo suficientemente avanzado que permita un flujo de trabajo en la creación y edición de videos y que de salida a los mismos con la mayor calidad posible en imagen y en audio.

En un balance final, los videos-infografía que se realizaron para la REDIM, fueron elaborados utilizando un 70% software libre. Los videos puedes verse en http://vimeo.com/33458911 y en http://vimeo.com/33726839

La elaboración de esta tesis, fue escrita en Libreoffice y luego maquetada y diseñada en Scribus

Por último, la elaboración de esta tesis, que ha sido, hasta ahora, el único proyecto editorial que he realizado, fue escrita en Libreoffice y luego maquetada y diseñada en **Scribus**, el programa libre que podría homologarse con InDesing de Adobe.

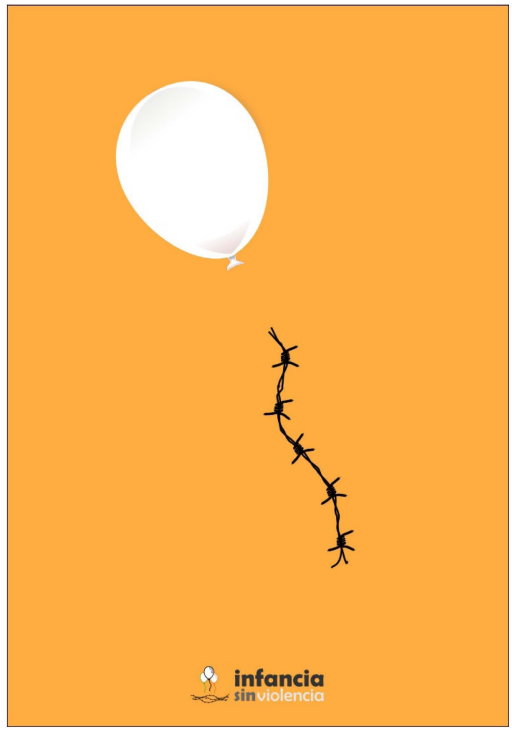
Para este caso en particular, el problema sustancial con el que me topé es que Scribus no tiene un soporte avanzado de citas al pie de página, razón por la cual, en la versión final impresa de este trabajo, las citas se encuentran al final de cada capítulo. Particularmente en este caso y contrario a lo que me pasó con Inkscape y con Gimp, Scribus no representó para mi un problema de "familiarización" con la interfaz, quizá por el hecho de que no había usado

Indesign con mucha regularidad anteriormente y la falta de referente directo, hizo que usara Scribus y lo conociera y aprendiera como quien abre y se familiariza por primera vez un programa que nunca ha usado.

Con el curso normal que representa el aprendizaje de una nueva plataforma, nos encontramos con el conocimiento de un nuevo sistema operativo, nuevos programas, en general, nuevas formas de resolver problemas, y casi sin querer, adquirimos la conciencia de las diferencias que existe con los otros softwares comerciales. Por ejemplo, en mi experiencia como usuaria de software libre, puedo decir que desde que lo uso de manera cotidiana, soy mas consciente de las deficiencias y beneficios que poseen ambos tipos de software, (el privativo y el libre) que cuando lo era como usuaria sólo de software privativo. Por otro lado, pero no menos importante, otra de las cosas con la que me enfrenté como usuaria de tecnologías abiertas, fue con el encuentro de la cultura libre, del libre circulación de objetos culturales y el uso de licencias permisivas como una alternativa al **copyright**. Me topé con una comunidad de software libre que poseía un particular interés por señalar la "autoría" de las obras. Nunca antes, incluso, no dentro de mi propio gremio que "crea" todo el tiempo, vi tanto señalamiento al "crédito en la autoría" de los objetos culturales. Situación que no sucede a menudo en el ámbito del diseño gráfico a pesar de que todo el tiempo "nos inspiramos" en obras de otro o echamos un ojo a lo que se ha hecho ya, antes de comenzar a diseñar o usamos recursos gráficos obtenidos en internet de manera legal o no, como una tipografía, imágenes, vectores, etc. El crédito, no siempre es respetado y eso lleva a que se usen términos como "robo" y a confundir "plagio" con "copia" en el terreno del diseño gráfico[3].

Still de "Copy is not theft" o "Copiar no es robar" de Nina Paley. Disponible con subtítulos en español en: http://www.youtube.com/watch?v=ZmYsLTUjXNI

infancia
sinviolencia



Diseño de logo y postales para la campala
Infancia sin violencia, de la REDIM; 2011

3.2. Otras experiencias.

3.2.1 Creadores usuarios de software privativo

Dentro de la maestría en Artes Visuales que da origen a este trabajo de investigación, le pedí a dos diseñadores, uno de ellos egresado de la UVM y otra de la ENAP; y a una comunicadora de la Universidad La Salle del Estado de Morelos, que utilizaran software libre para elaborar una imagen gráfica por computadora, también libre, que hablara de sus inquietudes con respecto a los conceptos de Cultura Libre que leyeron en el capítulo 1 del libro de Laurence Lessing.

El resultado fue sumamente interesante. Por un lado, ambos diseñadores experimentaron, como yo, cierta impotencia al enfrentarse a un programa desconocido, después de casi 10 años de uso de software privativo, me hablaron de lo tardado que les resultó hacer un diseño en Gimp que de haberlo hecho en Photoshop, hubiese resultado infinitamente más rápido. Esto quizá pueda solucionarse con el uso, pues experimentamos la misma sensación cuando aprendimos a usar Photoshop, sin embargo, esto no fue lo mas interesante, ya que el primer temor al que se enfrentaron, fue el de no encontrar los "filtros" conocidos en Photoshop y todos los pinceles encontrados en Illustrator.

Esto me llevó a reflexionar, por un lado, que esto pueda responder a la necesidad de encontrarnos con lo ya conocido, el temor al cambio y a la búsqueda de las herramientas que ya sabemos manejar, pero también puede tratarse de una dependencia a una herramienta y a sus recursos, esto es, a los filtros y a las opciones predeterminadas que no solo dan inmediatez a las soluciones, sino que puedo atreverme a decir, marcan una estética, la estética del "Photoshop", de los filtros.

Ambos diseñadores pudieron resolver sus procesos creativos casi sin ningún problema, la exploración les ofreció una nueva incursión a recursos que no conocían, aunque nunca pudieron dejar de compararlo con el software privativo que tanto han usado.

Por otro lado, la experiencia de la comunicadora fue completamente distinta, quizá debido a su formación en donde las herramientas para generación de imagen digital no son tan recurrentes o son enseñadas de manera superficial. Ella quedó bastante satisfecha con su incursión en el S.L y consideró que era sencillo, fácil de usar, intuitivo y dinámico, además de que al preguntarle su experiencia, puso énfasis en lo importante que le pareció que pudiese bajarse de Internet un programa para solucionar problemas de diseño de manera tan sencilla, fácil y que además, fuera legal.

3.2.2 Creadores usuarios de software libre

En la estancia de investigación realizada durante el segundo trimestre del 2011, pude entrevistar a algunos usuarios argentinos que llevaban ya algunos años usando software libre y que incluso lo utilizaban para el ejercicio de su actividad profesional. La mayoría de ellos son artistas visuales y/o diseñadores y/o docentes.

Dichas entrevistas arrojaron algunas similitudes en cuanto a razones por las cuales usaban tecnologías abiertas y los problemas con los que se enfrentaron. La mayoría de los entrevistados descubrieron el software libre en una búsqueda por una opción diferente al software privativo que les generaba problemas técnicos o bien, comenzaron a cuestionarse sobre el tema del pago de licencias y el uso de copias del programa no legales. Otros más, se encontraron con el software libre de manera casual y la ideología implicada en él, les hizo eco en su personal manera de pensar.

De esa forma, 8 de los 12 de estos entrevistados, aseguraron que usan Software Libre por convicción ideológica, una vez que conocieron las características de este software y encontraron una relación con su filosofía, pues se apega mucho a lo que creen respecto a herramientas tecnológicas y su quehacer profesional.

Al respecto, Lila Pagola, comenta:

Trabajo experimental de Edgar Martínez Mendoza, usando software libre.

"Para mi, el S.L. fue un gran descubrimiento, (...) una aproximación crítica a la tecnología desde el arte. El S.L. tenía mucho que ver con cosas que a mi me interesaban y que había hecho previamente y cuando fui entendiendo lo que la cultura libre significaba, me fui dando cuenta que eran ideas que ya existían en el arte mucho antes de Laurance Lessig[4](...) en el caso de la cultura y el software libre, no diría que las ideas ejercieron influencia en mi, más bien yo me reconocí en esas lineas"[5]

En ese mismo sentido, el artista electrónico, Christhian Oyarzún, dice:

"Yo vivo de dar clases y distribuir conocimiento es algo que me interesa mucho. Yo creo que uno debe distribuir permanentemente y citar la fuente. Pasó que apareció la idea del software libre, a la cual mi manera de ver el mundo se adaptó mejor. Emergió un discurso social que de pronto, me representó" [6]

Así mismo, 9 de los entrevistados dijeron que una característica importante del software libre que se auna a las razones por las cuales lo usan, se refiere a que es un tipo de software que les permite conocer a profundidad sus procesos de creación digital, que permite cuestionarse el uso de la tecnología en el arte y el diseño, y que es una excelente manera de desmitificar a los aparatos y sobrevaloración a un software que lo hace todo.

Respecto a entender y aprender cómo funciona un software y la posibilidad de modificarlo, Ignacio Nieto comenta:

"Entender el mecanismo interno para el cual fue creado (el software) y entenderlo de la misma forma en como se entiende el mecanismo económico, cómo funciona en términos políticos, cómo se arma y tratar de recrearlo y por qué".[7]

En ese mismo sentido, Guillermo Espertino, dice:

"Con el software libre puedes desmitificar la idea de que los grandes gigantes del software y del hardware siempre hacen todo bien y lo que vienen de ellos es una palabra santa. En el software libre uno tiene que entender cómo funciona el proceso completo para poder lograr un buen resultado (en el caso de impresión en imprenta); uno utiliza la herramienta un poquito más a conciencia."[8]

Ya sea por cuestiones éticas, de experimentación y/o de convicción ideológica, de la misma forma en la que los entrevistados reconocieron las razones por las cuales usan S.L., también se tocó el tema las "limitaciones" que representa su uso y lo tardado que pueden volverse algunos procesos una vez que se esta acostumbrado al uso del software privativo.

Ignacio Nieto, profesor y artista visual en entrevista, Santiago de Chile, Mayo del 2011

Al respecto, Espertino dice:

"El primer problema es de uno, no del software, uno tiene que sacarse primero todos los prejuicios y pre-conceptos que traen desde el otro software. Uno quiero que Gimp sea como Photoshop y no es así."[9]

"En algo mas proyectual como en el diseño, te puedes encontrar con inconvenientes técnicos, entonces yo advertiría que (el diseñador) tiene que tener en cuenta que se va a encontrar con algunas cosas que tienen que ajustar y que deberá desarrollar un flujo de trabajo que sea compatible con el software".[10]

Es cierto que para adentrarse en el uso del software libre una vez que se ha tenido ya un uso cotidiano, constante y prolongado de software privativo, se debe de estar consciente que el cambio representará ciertas complicaciones en una primera instancia, la adecuación a la herramienta a veces requiere tiempo, pero es un proceso igual al que se experimentaría de pasar de Windows a OSX o viceversa, ya que: "Todo cambio de software requiere un aprendizaje, e incluso cuando el usuario más sabe, tanto más le costará olvidar lo que sabe y leer advertencias, prestar atención, tomarse el tiempo para aprender. "[11]

Esta ruptura que en un principio podría parecer una barrera, podría ser también una oportunidad para consultar, para acercarse a la interdisciplina y romper con algunos paradigmas establecidos, tal como lo comenta Christian Oyarzún:

"Te tiene que gustar mucho (usar S.L.) por que hay que consultar y perderte en un mar de información, eso, evidentemente fortalece el proceso creativo y te hace tener más conciencia de la herramienta que usas"[12]

Guillermo Espertino, dice:

"Yo creo que soy mejor profesional desde que estoy usando S.L. por que pude entender mejor qué lo que hago, conocer los procedimientos, encontrarme con aspectos técnicos que a veces dejamos en segundo plano en pro de la velocidad y de cumplir con un cliente. El software libre me obligó a hacer una pausa y mirar más lo que estaba haciendo"[13]

El uso de programas libres puede llevar una exigencia extra de tiempo de producción, sobre todo si se usa por primera vez, sin embargo, esto es sólo cuestión de tiempo, primero para

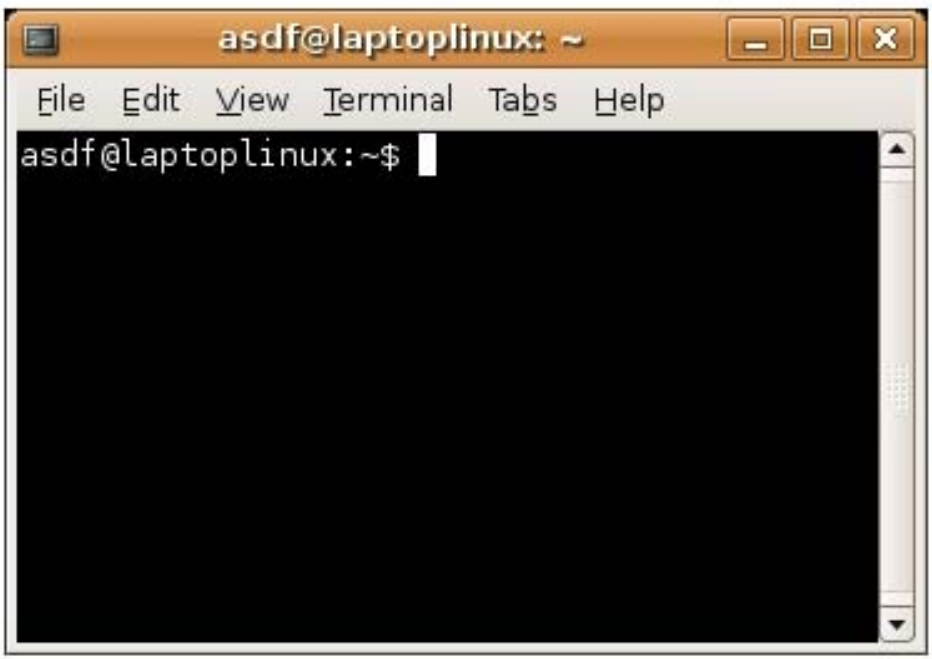
dominar la herramienta y segundo, que representa un largo plazo; las herramientas libres tiene una posibilidad infinita de crearse y transformarse, lo cual garantiza que si no hay un programa libre que resuelva de manera óptima alguna necesidad, lo hará en la medida en que exista una comunidad que lo requiera y a su vez, lo desarrolle y mantenga, ejerciendo así una especie de poder comunitario.

Como dijo Andrea Sagardoy, ilustradora argentina: "Muchas cabezas pensando y que éstas no sean necesariamente las que están en el poder o dirigen al mundo. El software libre es un poder de la gente"[14]

A lo largo de estas y otras entrevistas que se incluyen de una u otra medida a lo largo de esta tesis, podemos observar cómo es que el perfil de los usuarios profesionales de software libre coincide con una reflexión respecto a la herramienta que utilizan para crear. Existe, en la mayoría de ellos, una convicción en cuanto al uso de las herramientas libres, así como un reconocimiento de las problemáticas que puede representar la migración a este tipo de programas. La filosofía que respalda al software libre coincide con la manera de pensar de los entrevistados, con su afinidad a la cultura libre, la colaboración y con la idea de que la construcción de la cultura y el conocimiento se hacen de manera colectiva.

Andrea Sagardoy, ilustradora argentina en entrevista, Mayo del 2011

Además, algo que no estuvo previsto en el desarrollo de este proyecto, es que los entrevistados que manifestaron afinidad y convicción con el uso del software libre coinciden en que, en su temprana edad, estuvieron en contacto con computadoras cuyo uso requería de la interacción con una terminal, o bien, con alguna forma de programación; es decir,que los que ahora exploran de manera directa el software libre, el código abierto, o que incluso, ingieren en las actualizaciones de este (en el caso de Guillermo Espertino que contribuyó con la iconografía del programa Inkscape en sus últimas versiones) y que rebasan el perfil de "usuario", aprendieron que la interacción con una máquina, se hacía, necesariamente a través del código y la programación. Evidentemente, sin que esto sea determinante pues se requeriría de un estudio mucho más profundo y una muestra mucho más amplia para comprobar esta hipótesis. Sin embargo, esto nos podría llevar a plantearnos una pregunta para un futuro proyecto: ¿es importante la enseñanza y aprendizaje de la programación para la alfabetización digital, que el conocimiento del código sea una necesidad tal como la de aprender otro idioma?



Intérprete de ordenes, terminal o consola, como forma de acceder al sistema sin utilizar la interfaz gráfica.

3.3. Procesos creativos y código libre, ¿cómo influye en la creación?

El acceso a la llamada "receta" con la que fue realizada la herramienta digital, abre un mundo de posibilidades al artista y diseñador, tal como si se abriera la puerta de la ingeniería detrás de la creación de su obra, lo cual, invariablemente le ofrecerá muchas más posibilidades creativas, esto es por que gracias a la capacidad de intervenir el código, redistribuir dichas intervenciones, construir con los bloques que otras personas han hecho disponibles, es absolutamente esencial para impulsar el proceso creativo, el software se mueve del "sólo lectura" al "leer, escribir"[15], ya que con los conocimientos necesarios de programación, podríamos compilar[16]varios códigos ya probados y usados por otros usuarios y generar una nueva solución gráfica, o bien, crear los propios y luego compartirlo para que otros se beneficien de ello.

La apertura del código permite no tener que esperar pasivamente a que la compañía de software decida incluir nuestras necesidades creativas en la siguiente actualización de su programa más popular, por el contrario, deja abierta la posibilidad de que cada vez más usuarios se sumen al mejoramiento y distribución de dicha receta.

"Ya no son implementaciones de los conceptos artísticos vinculados por los límites de las aplicaciones de software que existen. Se es libre de crear las propias herramientas de combinación o modificación del sistema operativo GNU/LINUX y la accesibilidad del código fuente de todas sus aplicaciones. Es como crear objetos con bloques de lego"[17]

E incluso, si nosotros no tenemos habilidad y/o conocimiento en cuanto a la manipulación de código se refiere, existe la opción de recurrir a la interdisciplina, tal como lo menciona Inne Martino, artista visual Argentina dice:

"Las limitaciones las tiene cada uno y si con el software libre tienes la posibilidad de llamar a un programador y decirle: necesito que me hagas esto, aquello y lo otro" y que te

pueda adaptar un programa para que te resuelta tu necesidad, no hay limitaciones y puedes resolver todas tus necesidades de creación sin tener que pagar una licencia de software"[18]

La interacción profunda que puede promover los programas libres en la relación usuario-software propicia un acercamiento con la herramienta que de pronto, deja de ser un producto para convertirse en medio de creación.

"El proceso creativo de un artista ya no está limitada por lo que dictan las empresas de software, sólo por la habilidad propia del artista. El software funciona como medio. (...) Y en este modelo, los artistas se dan libre acceso a esta capa de la obra de arte"[19]

Las posibilidades son infinitas, como se citó anteriormente; cuando se crea un objeto con bloques de lego, se arma de acuerdo a las propias necesidades del artista o diseñador, creando soluciones a la medida y muy adhoc con las características del pensamiento divergente que poseemos como creadores y con el proceso creativo, ya que "hay que poner especial atención a las diferencias individuales y a las necesidades personales al momento de instrumentar las estrategias de desarrollo de la creatividad."[20]Tal como lo menciona la filosofía UNIX de la que hablamos en el capítulo 1, hacer programas pequeños que resuelvan tareas específicas.

Incluso esta nueva incursión en el proceso creativo, no debe ser visto como otra categoría de arte, sino como una capa adicional ya que "actúa como una nueva dimensión en la parte superior de los campos existentes de arte digital y enriquece la forma de los artistas y colectivos pueden trabajar mediante la adición de otro grado de libertad en el proceso creativo"[21]

Así pues, el software libre no sólo podría representar una interacción más profunda en el proceso creativo que conlleva la obra, sino que ayudaría a desmitificar la tecnología tomando, como creadores, una actitud más activa y de control frente a lo que hacemos con una computadora y lo que necesitamos de nuestras herramientas.

Citas del capítulo 3

1Entrevista con Guillermo Espertino vía e.mail en Enero del 2012 mientras se encontrada colaborando activamente en la inclusión de nuevos pinceles para la nueva versión de GIMP. Debido a su acercamiento con el proyecto, conoce algunas de estas políticas de desarrollo que no son públicas en el sitio oficial de GIMP.

2Los videos pueden verse y descargarse desde: http://vimeo.com/33458911 [3] y http://vimeo.com/33726839 [4]

3Pensemos cuántas veces hemos oído decir "me voy a robar esta idea" o no lo digo por que "me van a robar la idea" comparando a la copia como el "robo", y refiriéndolo como un acto delictivo, malintencionado o digno de señalamiento.

4Laurance Lessig es el que acuñó en término "Cultura libre". Él explica el concepto en el libro que lleva el mismo nombre.

5Pagola Lila, extracto de entrevista Córdoba, Argentina, junio 2011.

6Oyarzún Christian, extracto de entrevista Santiago, Chile, mayo 2011.

7Nieto Ignacio, extracto de entrevista Santiago, Chile, mayo 2011.

8Espertino Guillermo, extracto de entrevista, Rosario, Argentina, mayo 2011.

9Espertino Guillermo, extracto de entrevista, Rosario, Argentina, mayo 2011.

10Espertino Guillermo, extracto de entrevista, Rosario, Argentina, mayo 2011.

11Lila Pagola, "Software libre, una caja abierta y transparente", en: Instalando: Arte y cultura digital ed. Troyano: Ignacio Nieto, Italo Tello, Ricardo Vega (Chile: Lom Ediciones, 2007) 100.

13Espertino Guillermo, extracto de entrevista (Rosario, Argentina en Mayo 2011).

14Sagardoy Andrea, extracto de entrevista, Rosario, Argentina, mayo 2011.

15Mansoux y Valk **Floss and Art** 6.

16Antes de poder ejecutar un programa escrito en un lenguaje de programación, debemos traducirlo al lenguaje de la máquina sobre la que queremos que corra. Para cada combinación de procesador, lenguaje y sistema operativo existen traductores automáticos, llamados compiladores. Se trata de programas que leen un programa escrito en un lenguaje de programación y, a partir de él, generan uno escrito en el lenguaje de ejecución adecuado para una determinada combinación de procesador y sistema operativo .

17Mansoux y Valk **Floss and Art** 10.

18Extraído de la entrevista a Inne Martino, en Rosario, Argentina en Mayo del 2011.

19Mansoux y de Valk **Floss and art** 10.

20Gabriel Martínez Meave y otros, **Diseño, tipografía y lengiaje**, (México D.F: Editorial Designio, 2004), 108.

21Berry, "Bare Code," 133.

%%title:

Capítulo Cuatro

<title>

Antes de concluír
hacia la enseñanza del FLOSS
¿por qué enseñar con software libre

</title>

Hemos visto ya, a lo largo de los capítulos anteriores, algunos argumentos para considerar el uso del software libre en las artes y el diseño. Ahora bien, imaginemos el uso del software libre no sólo cómo herramienta en la creación de objetos culturales, sino como herramienta en la enseñanza, o mejor aún, enseñar esta herramienta como auxiliar en la creación.

En este último apartado, me gustaría hablar brevemente de algunas razones por las cuales resulta importante también que el software libre se enseñe a los artistas y diseñadores en formación. Esto, debido a que a lo largo de la investigación, pude notar que uno de los problemas por lo cual el S.L. no se conoce, no se utiliza, o bien, es tomado como una opción poco viable para el ejercicio profesional de las artes digitales, es por que artistas y diseñadores desconocen otras opciones lejos de las que les son enseñadas en la academia. Eso hace que ignoren los alcances de una herramienta ajena a las que han usado tradicionalmente y duden de su calidad, tal como lo afirma José Serralde:

Uno de los principales problemas que enfrenta el software libre, tiene que ver con un problema de cultura, con desconocimiento por parte de muchos usuarios hacia este tipo de software y el poco o nulo acercamiento al concepto de "cultura libre"[1], o como lo resume Martín Eschoyez en una frase: "La ignorancia es el principal impedimento para que los diseñadores y artistas se acerquen al Software Libre"[2]

Así pues, dedico este apartado para presentar algunos argumentos que respaldan la enseñanza del software libre, a manera de propuesta para la difusión dentro de la comunidad de creadores. Con ello, se deja abierta la posibilidad de la continuación hacia un nuevo y mas extenso trabajo de investigación que retome los alcances de la enseñanza con este tipo de software.

4.1. La enseñanza como estrategia de difusión, que el creador se acerque al software libre

A pesar de que muchos creadores gráficos digitales, desconozcan la existencia del software libre, no quiere decir que la información sobre los programas libres no exista, o bien, que la difusión de los mismos sea escasa o nula, más bien, no responde a campañas mercadológicas ni son anunciados en comerciales de T.V, ni son incluidos en el programa de estudios de universidades mexicanas y tampoco responden a estrategias de condicionamiento[3], es decir, a la salida y casi obligada adquisición (legal o ilegal) de una versión por año o condicionar un software a un tipo de hardware, como ya lo mencionamos en el capitulo 2.

La difusión, así como la mayoría de las asuntos que conciernen al software libre (desarrollo, documentación, tutoriales, soporte, etc), depende de la comunidad que lo realiza, y muchos de esos esfuerzos por promover y dar a conocer las ventajas del software libre están en Internet. Blogs, foros de discusión, revistas especializadas en línea y artículos sirven como medio para difundir y conocer la ideología del S.L. así como para responder las dudas con una nueva instalación, programas o scripts. Básicamente, mucha de la información que se necesita conocer sobre el S.L. se encuentra al alcance de cualquiera que tenga una conexión de Internet y este interesado en ella. Sin embargo, para llegar a este punto de interés que consecuentemente motive el acercamiento y uso de los programas libres, se requiere al menos un acercamiento previo que permita al individuo conocer de la existencia de un software distinto a los más conocidos.

Para ello, considero necesario que el software libre se de a conocer desde los medios de enseñanza, es decir, en el entorno escolar, ya que la adquisición de un nuevo conocimiento es favorecido cuando la mente se encuentra abierta para recibirlo, y esto sucede con frecuencia en las aulas: "El mayor obstáculo que bloquea a la gente para un fácil acceso al FLOSS[4]es el hecho de que los estudiantes son raramente introducidos en las universidades y escuelas de arte. El mejor momento para familiarizarse con las computadoras y el software es en la escuela"[5]

A los estudiantes rara vez de les da a elegir. La elección de las escuelas y universidades para ofrecer sólo este tipo de producto (s.p.) tiene sus razones

La enseñanza del software libre no sólo es escasa en las escuelas de arte y diseño, sino en cualquier otro ámbito y nivel educativo. En la educación secundaria, por ejemplo, las clases de computación están dirigidas a enseñar el uso de software privativo, que generalmente, comienza con los paquetes de ofimática. Esto hace creer al alumno que Microsfot Word es igual a procesador de texto cuando se trata de una marca que es provista por una empresa en particular.

Lo mismo sucede en el ambiente universitario, donde el uso de un determinado software esta supeditado a las necesidades de un medio laboral, es decir, para generar estudiantes mucho mas empleables:

"A los estudiantes rara vez de les da a elegir. La elección de las escuelas y universidades para ofrecer sólo este tipo de producto (s.p.) tiene sus razones. Aprender a utilizar el software privativo y sistema operativo más dominante, hará de un estudiante mucho más empleable, ya que los productos son también los que más se usan en la industria. Por supuesto esto sólo va para los estudios destinados a la industria, no tiene sentido en el contexto del arte"[6]

En este mismo sentido, cuando se considera el momento de emplear a un diseñador, habilidades como el desarrollo de conceptos, la creatividad en el uso de herramientas y el pensamiento fuera de la caja negra debieran ser mucho mas valiosos para un futuro empleador que saber cómo usar un determinado producto, además de que estas habilidades serán obsoletas tan pronto como ese producto se actualice.

Es por eso que algunos catedráticos de universidades españolas reflexionaron desde hace tiempo la responsabilidad que implicaba enseñar lo mismo tecnologías abiertas que privativas. En un artículo de la Universidad Oberta de Catalunya, se habló acerca del potencial creativo que posee en software libre:

"Consideramos que el software libre potencia la creatividad en un sentido más amplio. Trabajar para producir en el lugar que al mismo tiempo permite ser modificado (re) creado y manipulado, nos parece una opción con un nivel superior de creatividad"[7]

Bajo este modelo de interacción entre la filosofía del software libre y la creación gráfica digital, donde el conocimiento, interacción y modificación del programa es posible, los artistas y diseñadores podrían tener sus propias herramientas y tendrían la libertad para usarlos

cuando y como quieran, podrían diseccionar, hackear, embellecer y compartirlo sin romper ninguna ley.[8]Podría entenderse, incluso, que se pretende "liberar a la creatividad de los grilletes de la propiedad"[9].

En el caso por ejemplo, de la Escuela Nacional de Artes Plásticas, que por definición estatutaria de la UNAM, forma profesionales no sólo al servicio de la industria, sino también, al servicio de la sociedad y del arte, requeriría que sus alumnos decidieran qué tipo de software usar, según el que cumpla con sus necesidades, ya sea, ponderando la inmediatez que nos da el software privativo[10]o la libertad que ofrece el software libre. Dicha decisión, no podrá generarse en los alumnos mientras no se le enseñe las característica y uso de ambos:

"En los alumnos, debe de haber una elección consciente de porqué usa esa herramienta. Esta bueno cuestionarse todo. ¿Me sirve usarla, me conviene usarla?"[11]

Sin embargo, también tenemos que estar conscientes que muchos de los esfuerzos que requiere un cambio de uso de software imposibilitan que los usuarios se acerquen a él, tal como Pagola dice al respecto:"No es que los docentes elijan usar software privativo por que les guste o por el modelo que supone, sino por que éste tiene más funcionalidades dominadas que no pueden en general reemplazar con el S.L. o que no quieren. Suelen resistirse al cambio por que por comodidad o funcionalidad, el S.L. aún no se los ofrece sin esfuerzos."[12]

Escuela Nacional de Artes Plásticas. Foto: Irene Soria

Estuadintes Rusos usan un sistema operativo libre desde 2009. Foto: C-News, disponible en: http://www.cnews.ru/news/top/indexEn.shtml?2007/09/14/266177

La cita anterior, resulta contundente, ya que para que pueda existir un acercamiento al uso de este software a lo estudiantes, quizá primero tenga que haberlo a los educadores. Esto se complica si no existe una política de uso de software libre desde altas esferas, como iniciativas gubernamentales que a su vez se vean reflejadas en la institución educativa; que la propia UNAM promueva y difunda el uso de este tipo de software en sus facultades, tanto para alumnos, profesores, investigadores, como a administrativos.

En este sentido, existen ejemplos internacionales de la adopción de software libre en el ámbito educativo, a partir de políticas publicas, por ejemplo, en el año 2002, el gobierno brasileño tomó la decisión política de usar software libre en todos los ámbitos del Estado. Desde entonces, la iniciativa ha impulsado la migración a este software en las escuelas; como es el caso del proyecto de reutilización de computadores desechadas por el gobierno, empresas estatales y privadas, para ser usados en telecentros comunitarios, escuelas y bibliotecas. Todos funcionan con software libre. Además el estado de Paraná creó el proyecto “Red Escolar de Paraná”: 2. 037 escuelas utilizando software libre[13].

En 2007, la nación de Rusia anunció que todas sus escuelas comenzarían a utilizar S.L. en el 2009. Un informe de la BBC sobre este asunto declaró que en Rusia "las escuelas solían ejecutar copias ilegales de sistemas operativos de Microsoft", pero que esto resultaba una práctica no aceptada. Por lo tanto, en lugar de comprar licencias para todo el software que

había sido adquirido ilegalmente, se optó por usar el sistema operativo libre GNU/LINUX. Si bien, la mayoría de los profesores y los estudiantes no tenían ninguna experiencia con GNU/LINUX, los funcionarios en educación de Rusia, consideraron que la transición fue bien y que el software se adaptó a los propósitos de la escuela[14].

En 2007 más de 560000 estudiantes alemanes y miles de empleados de más de 33 universidades alemanas comenzaron a usar el sistema GNU/LINUX a partir de políticas del estado de Reania del Norte-Westfalia,[15]. La ciudad de Munich también optó, desde el 2005 por migrar al Software Libre más de 14 000 computadoras de escritorio[16].

Sin embargo, no todas las iniciativas educativas de adopción del S.L. vienen desde el Estado. En la Universidad Nacional de Córdoba (UNC) se comenzaron a utilizar productos de software libre a partir de la conexión a Internet (1995) y su uso se comenzó a difundir en forma sostenida. "Alentados por productos confiables, excelente perfomance, limitada demanda de hardware, compatibilidad absoluta y libre disponibilidad, las distintas unidades académicas y dependencias de la UNC continuaron utilizando esta clase de productos y aumentaron su difusión, especialmente en el ambiente de servidores"[17]. Esta incorporación de programas libres no siguió política, metodología o modelo, solo la recomendación personal de usuarios avanzados y más experimentados o bien, con el comienzo azaroso de algunas soluciones, lo cual, es muy común en todos los ámbitos, tanto el académico como el administrativo[18].

Munich, una de las ciudades Alemanas que desde el 2005 optó por migrar a Software Libre. Foto: BY, NC, SA, @giannisl

4.2. ¿Por qué enseñar el software libre?

Tal como lo menciona Cristobal Cobo en su ensayo "Aprendizaje de código abierto", el impacto del open source en la educación representa varias ventajas, ya que los avances en los desarrollos digitales de código abierto han permitido:

"1) La disponibilidad de recursos educativos de bajo costo o gratuitos, lo que genera un notable beneficio social y pedagógico para estudiantes y profesores; 2) ahorros significativos en infraestructura tecnológica, en particular para aquellos países con economías más vulnerables, y 3) impulso de una cultura del libre intercambio de materiales educativos, válido en instituciones de educación básica y superior, así como en centros de investigación científica de punta"[19]

En lo que respecta al punto número 1 y 2 de la cita anterior, es innegable que uno de las principales atractivos del software libre es el bajo costo[20]que implica su implementación, y las ventajas que esto puede traer en cuanto a medios educativos de refiere, sobre todo si se trata de educación pública. Sin embargo, esto es no es contundente y aunque resulta importante, también es superficial, dado que esto puede ser fácilmente alcanzable por las empresas de programas privativos. Tenemos el ejemplo de Microsoft que "regala" licencias en entornos universitarios[21].

Es innegable también que muchas de las instituciones de educación pública, incluyendo a la Escuela Nacional de Artes Plásticas, y la UNAM tiene instalado en sus equipos copias no autorizadas del software, esto, de alguna u otra manera fomenta en los estudiantes el uso de copias no legales sin reparar que se trata de un delito, que si bien, parece no ser importante por que en México se realiza con bastante frecuencia y sin castigo, no deja de tratarse de una actividad penada por la ley:

La compra de copias de software no autorizado, es una práctica común en México.

"La posibilidad de copiar y difundir el software sin recurrir a copias ilegales, es una forma de evitar que la propia institución educativa lleve a sus alumnos y docentes a violar la ley. Suele suceder, además que esto ocurra sin que los docentes, alumnos o familiares vinculados a la escuela tengan siquiera conciencia de que están cometiendo una actividad penada por la ley"[22]

Enseñar con software libre, impulsaría una cultura del libre intercambio de materiales educativos; promovería el uso de una herramienta legal y protegería a la propia institución y a los estudiantes, de cometer una acción fuera de los estatutos legales sólo por enseñar y usar una herramienta necesaria para el ejercicio profesional.

4.2.1. Motivar la generación de conocimiento, un software que fomenta la colaboración

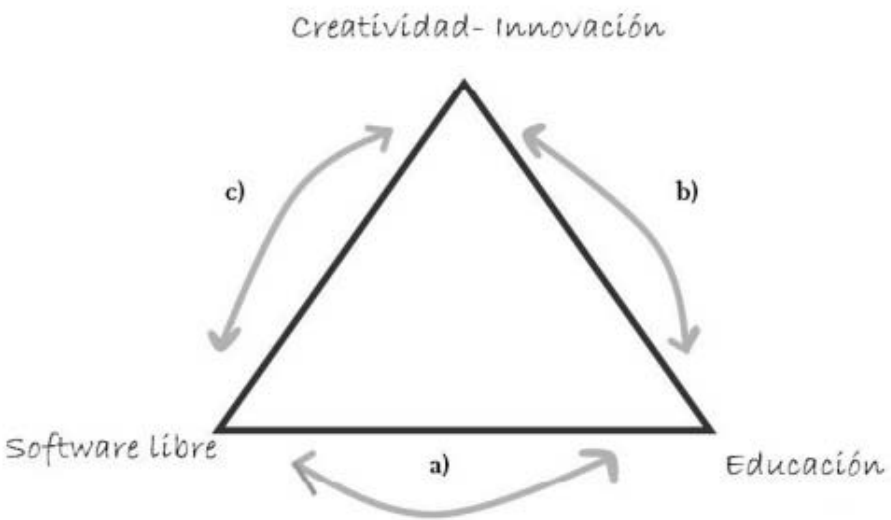
Sabemos ya a la altura de este último capítulo, que el software libre no solo supone una tecnología con el código abierto para que todos puedan colaborar en su creación y constante crecimiento, sino que también, habla de una cultura libre, del libre intercambio de conocimientos y por supuesto, de un crecimiento en masa a partir de la participación colaborativa, tal como lo menciona Cobo:

"(la) Educación universal como el open source se nutren de aquella premisa que postula que el intercambio de conocimientos genera nuevos saberes"[23]

Es decir, la cultura de la participación y de la colaboración que promueve el software libre, no sólo resulta beneficiosa al momento de mejorar un programa, sino que también, fomenta en los estudiantes la cultura de la colaboración y de la participación, inculcando la importancia del enriquecimiento del saber humano:

"El paradigma del Software Libre invita a que se genere cooperación, colaboración y reconocimiento de las diferencias como una forma de enriquecimiento y fortalecimiento mutuo, valores que deben ser impartidos desde la escuela a los estudiantes de manera que se generen estilos de vida beneficiosos para la sociedad en conjunto."[24]Continúa diciendo: "Una cultura orientada a compartir el conocimiento, a abrir canales de intercambio, es una cultura que tiene más posibilidades de crear, de innovar y de crecer, esta es justamente la filosofía del S.L."[25]

El propio Cristobal Cobo ha propuesto un triángulo donde se representa la relación entre software libre, creatividad y educación:



Círculo vistuoso entre creatividad, educación y Software Libre, según Cobo.

En donde ambas figuras: a) software libre – educación (aunque en escalas diferentes) se nutren de la premisa que postulan "que el intercambio de conocimientos genera nuevos saberes"[26]

Respecto a esto, el artista visual argentino Fabricio Caiazza comenta en entrevista:

"(el S.L.) Es un software colaborativo y muchos de los aspectos pedagógicos de las escuelas contemporáneas contemplan la posibilidad del trabajo colaborativo para la construcción de elementos pedagógicos didácticos. Este software es coherente con éste tipo de prácticas."[27]

Fabricio Caiazza, artista argentino usuario de software libre. Foto: Irene Soria.

Continuando con esta idea, Cobo retoma a Meiszener, Glott y Sowe que dicen que:

"Las comunidades de software libre proporcionan y distribuyen de una manera sostenible el conocimiento necesario para la producción de software de calidad. La forma de entender la creación de conocimiento en estas comunidades es muy distinta a la de los productores de software de propiedad (...) Los expertos y los usuarios crean el conocimiento en colaboración, la ayuda se proporciona mediante sistemas de apoyo "de usuario a usuario", y la sostenibilidad y la calidad también quedan garantizadas mediante la participación de la comunidad."[28]

Pensemos por ejemplo, que al enseñar Photoshop como programa de manipulación de imágenes, se enseñe a la par Gimp; como una alternativa libre, que tal como lo menciona Casado, Marín, Beneito, entre otros, Gimp es el resultado de la suma de muchas individualidades y animar a los estudiantes a participar en el proyecto refuerza este espíritu de trabajo en común; al igual que el uso de Blender, la herramienta libre para modelado y animación en 3D, -dicen los autores- que la participación de los estudiantes en comunidades abiertas y en foros de discusión, algo que el entorno de Blender facilita; fomenta el espíritu de trabajo colaborativo y posibilita diseñar intervenciones pedagógicas en ese sentido[29]

Podríamos decir hasta este punto, que el software libre en la educación acerca al estudiante a la idea de colaboración y generación del conocimiento por parte de una comunidad, lo

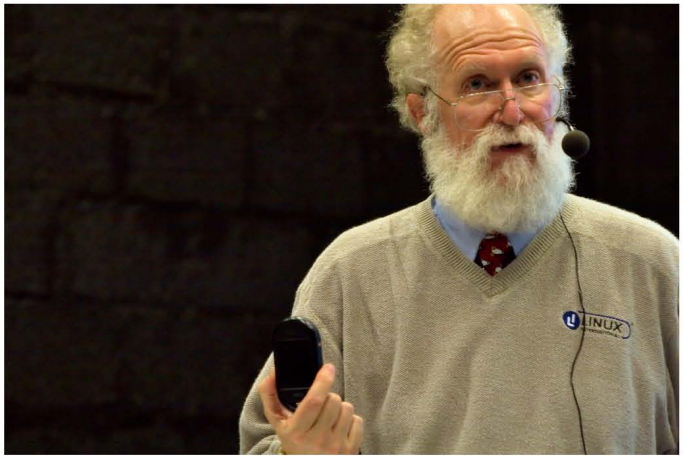
cual ayudaría notablemente a su formación como profesional. Ahora bien, además de ello, es posible que el estudiante entienda que no hay límites en cuando a software se refiere, que puede ser capaz de crear sus propias soluciones y que puede profundizar en el uso de un programa hasta dónde él lo desee, tal como se explica a continuación:

4.2.2. No hay límites

En el 2001, La UNESCO comenzó a apoyar al movimiento del software libre, pero no fue hasta el 2007 que edita un manual titulado: Guía práctica sobre Software Libre. Su selección y aplicación local en América Latina y el Caribe. En él, un apartado sobre el software libre y educación recomienda el uso de este tipo de software en cualquier nivel educativo. El poder acceder al código fuente es un estímulo permanente para la apropiación de las nuevas tecnologías y la innovación .[30]

El alumno puede entender que el software puede adaptarse a sus propias necesidades, y no limitarse a las opciones predeterminadas de uno privativo, pudiendo generar incluso su pro-

Guía práctica sobre Software Libre. Su selección y aplicación local en América Latina y el Caribe. Donde la UNESCO recomienda el uso de este tipo de software en cualquier nivel educativo.



John “Madog” Hall, presidente y director ejecutivo de Llinux International. Foto: BY-SA @ChrisAC

“(...)esto es una forma real y tangible de acercar las nuevas tecnologías a nuestra gente, acercando las posibilidades, en lugar de mostrar al software como la obra de algún ilumina-do que seguramente vive en un país desarrollado y necesita una enorme infraestructura para su trabajo. Esto no es así, y el software libre es una muestra tangible de eso”[31]

Algo similar dice John “Madog” Hall, presidente y director ejecutivo de Llinux International, en entrevista a Master Magazine durante las Jornadas Regionales de Software libre en la ciu-dad de Buenos Aires en el 2008. En dicha entrevista, habla de cómo las Universidades pue-den fomentar la capacidad de resolver problemas con el uso de programas libres:

“Con el software libre no sólo se puede aprender a resolver problemas, sino que también puede aprender cómo el software resuelve los problemas. Y si además tienes la habilidad, puedes hacer que el software resuelva el problema de mejor forma”[32]

4.2.3. No existe una sola herramienta correcta

Si bien, los diseñadores y artistas debemos tener un conocimiento teórico sólido de nuestra disciplina, si se nos enseña a analizar una imagen de manera profunda, a ser analíticos en cuanto a estrategias de comunicación visual se refiere; sí aprendemos de figuras retóricas y

técnicas de composición, como creadores digitales quizá deberíamos comenzar a tener un conocimiento más profundo acerca de las herramientas que ocupamos, de nuestro medio de producción y por lo tanto, del software que usamos; saber cómo funciona, e incluso, poder replicar su funcionamiento, conocer los algoritmos que llevan a un programa a realizar la acción que queremos. En pocas palabras: "Enseñar o desarrollar habilidades, no a usar un producto"[33]

Respecto a esto, Martín Eschoyez, diseñador gráfico y profesor argentino dice*:* "Yo no trato de enseñar un software, lo que yo trato de enseñar, son conceptos, y el S.L. a mi me permite bajar esos conceptos a la máquina. Trato de hacer que los estudiantes entiendan cuál es la filosofía que hay detrás para que ellos sean conscientes a la hora de elegir."[34]

Lila Pagola, por otra parte comenta que*:* "lo que aporta el uso del S.L. primero es mostrarles (a los alumnos) que **no existe una sola herramienta correcta** y que el lugar que le toca al maestro no es revelar los trucos que nadie sabe y que no te enseñan en el manual (...) no tiene mucho sentido transmitir cosas específicas, recetas de cómo hacer esto o aquello, si-

Lila Pagola, en entrevista, Junio 2011, Córdoba Argentina. Foto: Irene Soria

no tener una comprensión un poco más estructural de qué es lo que hace un tipo de software o aquel otro (...) también sirve para que los alumnos ganen flexibilidad en relación con las herramientas, que no vean a Photoshop como la única forma de resolver ciertos problemas sino que se den cuenta que en realidad hay conceptos generales en la edición de imágenes de pixeles que cada programa implementa con similitudes o diferencias".[35]

Tal como lo dice Cobo, "en la era de la información y el conocimiento, más importante que la capacidad de retención de datos, es el desarrollo de destrezas y habilidades para relacionar contenidos, adaptarlos y aplicarlos en diferentes contextos".[36]Es por eso que en un mundo globalizado, es decir, donde la información corre a una gran velocidad, es necesario tener una perspectiva diferente en las formas de enseñar a los diseñadores, una nueva forma de transmitir el proceso de producción digital y así, enfrentar los cambios que trae el futuro. Es necesario fomentar el criterio, la búsqueda, la creatividad, y con ello una conciencia más profunda de las herramientas digitales que utilizamos para nuestro ejercicio profesional, que se han vuelto ya, en el medio de expresión de la nueva era.

En la era de la información y del conocimiento, más importante que la capacidad de retención de datos, es el desarrollo de destrezas y habilidades para relacionar contenidos, adaptarlos y aplicarlos en diferentes contextos

"Aunque este es un tema en gestación, los indicadores muestran que el desafío actual no está únicamente en incorporar más tecnología en las escuelas (hardware y software) sino en desarrollar una "cultura del open source" frente al uso de la información y el conocimiento que contribuya a la formación de nuevas competencias digitales y cognitivas para reforzar las bases de la educación de nuestros días"[37]

En entrevista, José Serralde, habló acerca del proceso de creación digital, el cual, debería poder estudiarse, enseñarse y por lo tanto replicarse, (hacía una analogía con La Piedad de Miguel Anguel). En el caso por ejemplo, de los fotógrafos de la vieja escuela, con el uso de una tecnología análoga para la toma y revelado de sus fotos. Es posible que ellos conocieran el funcionamiento de su cámara reflex, cómo funciona su lente, el flash, y en el caso del revelado, conocen los químicos para llevar acabo el proceso que revela la imagen y saben cómo es que actúan los químicos en el papel fotográfico, por consiguiente, dicho proceso, que es conocido a profundidad en la teoría y en la práctica, puede ser enseñado y por lo tanto realizado con diversas modificaciones, la pregunta sería, ¿no debería de pasar lo mismo con el software, dado que es la herramienta primordial para la creación digital?

4.3. Un ejemplo de enseñanza con software libre

La Universidad Oberta de Cataluña (UOC) apuesta por el uso del software libre, tal y como lo dice su vicerrector de tecnología, Lloreng Valverde "El software libre es una apuesta institucional, pero también responde a una petición de la comunidad"[38] y también de los proyectos relacionados con el s.l. que hay en marcha, así como del hecho de ofrecer un master oficial en S.L.. que pertenece a los estudios de esta Universidad".[39]

Dentro de las asignaturas que enseñan tecnología digital para creación en la UOC, se sentó la prioridad de enseñar software privativo, pero a la par y como elemento complementario, también la enseñanza de software libre, y tal como lo dicen Casado, Marín, Beneito, Corcoles, Giménez y Porta, el primer objetivo de esta propuesta es evitar una relación unilateral entre un procedimiento y un software en concreto.

Por ejemplo, si se enseñan sólo los procesos de edición con un programa en concreto, el estudiante puede llegar a la conclusión errónea que procedimientos como el ajuste por niveles, por curvas o las máscaras de capa son exclusivos de Photoshop, cuando en realidad se trata de procedimientos comunes a la mayoría de los softwares de edición:

"Observar los paralelismos entre Photoshop y Gimp induce a pensar en los procedimientos de trabajo en una forma más abstracta o generalista, desligada en fin de un programa concreto y asociada al retoque o edición digital en general"[40]

Sucede lo mismo con la enseñanza de las herramientas de modelado y animación 3D, por un lado, la opción privativa 3D Max que constituye un paradigma para la capacitación de los estudiantes como futuros profesionales, Blender, la opción libre, permite desarrollar en base a un uso paralelo, estrategias educativas que posibiliten centrar la atención del estudiante en los procesos fundamentales de la animación digital.[41]

Enseñar a los alumnos de artes y diseño el uso de programas tanto privativos como libres, les daría la posibilidad no sólo de elegir cual herramienta usar en base a sus características y diferencias, como lo mencionamos casi al inicio de este capítulo, sino que podrá entender de una manera integral, el procedimiento por si mismo, diferenciarlo en base a cada software y contar con más herramientas para su futuro ejercicio profesional.

Dentro de este contexto, la UOC probó con otras tecnologías en otros ámbitos de la enseñanza, justo para propiciar la comparación entre ambas soluciones, la privativa y la libre, uno de esos ámbitos fue la educación a distancia, donde un grupo de estudiantes hicieron un curso durante un mes en el que se usaba la plataforma privativa .net, y según los autores, el resultado fue descorazonador por las dificultades que encontraron para instalar el entorno en local, o para encontrar un servidor gratuito que les permitiera usarlo a aquellos que no fueron capaces de instalarlo en su equipo.

José Santorcuato,profesor en la Universidad Católica de Chile, trabajó para el proyecto de Microsoft, Escuelas del futuro (Innovative Schools). Una de las doce escuelas en el mundo se encontraba en Chile. Él era el encargado de enseñar tecnología, y según sus palabras, "todo resultó terrible, todo estaba bloqueado, no se podían instalar programas, los contenidos de youtube bloqueados, sourceforge igual, era una dictadura. El proyecto fracasó. Ahí entendí que no había otra opción que el software libre para el desarrollo, la educación y la creación. Ahí, migré"[42].

José Santorcuato, artísta electrónico y profesor chileno, usuario de software libre. foto: Irene Soria

Así pues, y continuando con la experiencia de la UOC, según su vicerrector de tecnología, dicha universidad se encuentra en un momento en el que el acceso a ella y el trabajo de sus estudiantes debe de ser multicanal, con movilidad : "No podemos permitir que sólo haya un determinado sistema operativo y unas herramientas muy concretas para poder hacer este trabajo"[43]. De ahí que la Universidad tenga una gran apertura al S.L. incursionando incluso en una maestría en el área, pues responde a una petición de su propia comunidad.

4.4. La parte ética

No hay que perder de vista que, como hemos explicado a lo largo de esta tesis, el software libre no sólo se limita al desarrollo de programas de computo, sino que, como movimiento social, ha derivado en el desarrollo de otros tipos de saberes. Así pues, el acceso al código se ha convertido en un nuevo tipo de acceso al conocimiento y como tal, puede percibirse como parte de una sociedad que se basa en el intercambio y compartición de ese conocimiento.[44]"El software libre, entonces, puede convertirse en un ejemplo 'vivo' de compartir y colaborar alrededor de conceptos tan abstractos como el conocimiento."[45]

Sin embargo, existen otros discursos que defienden la educación con software libre sin estar centrados en el acceso al código, ya que no siempre se tiene la posibilidad, necesidad o interés de intervenirlo, es decir, discursos que ya no se centran en la apertura código fuente. Dichas posturas se basan en que la enseñanza con S.L. se acompañe de una explicación de la ética que trae consigo su filosofía, tal como lo dice Alejandro Miranda:

"Si la sustitución de programas privativos por libres no se acompaña de la comprensión y apropiación de la propuesta ética, en el mejor de los casos, se reduce el uso de herramientas "gratuitas".[46]

La parte ética se ve reflejada, entre otras, en la capacidad de compartir, "en este caso software, pero se apuesta a que puede ampliarse y encadenarse a otras áreas dando una lección cívica llevada a la práctica"[47]

Se trata, además, de enseñar con herramientas que estén al alcance de todos, que garanticen que toda la comunidad haga el mismo trabajo en igualdad, sin estar condicionados a tener las posibilidades o no para adquirir de manera legal o ilegal, una herramienta que se ha vuelto casi indispensable para todo proceso, tanto creativo como industrial, científico o de oficina, y que no esté restringido por condicionantes económicos o por un determinado tipo de herramientas. Esta garantía, de poder adquirir una herramienta de manera legal y al alcance de muchas mas personas, sólo lo puede garantizar el uso del software libre[48]

"La decisión de trabajar con software libre es también una decisión ética, la expresión del deseo de vivir en un mundo organizado de una forma distinta, donde las barreras artificiales que benefician sólo a algunos pocos, son eliminadas."[49]

Enseñar software libre como herramienta para la creación, no solo le daría al alumno una posibilidad más, para que al menos, elija la herramienta que mejor le convenga; sino que también, "permitiría resaltar la relevancia en la educación de las implicaciones éticas y cívicas de las propuestas culturales libres (..) desde un movimiento contracultural, en un mundo en que aparentemente ha sido cooptado por el dominio de empresas que monopolizan el acceso y divulgación del conocimiento".[50] Permitiría al futuro profesional a ser consciente de la nueva era de la que forma parte, y le promovería el uso de material libre y así, fomentar una cultura de la libre circulación de ideas y material cultural para el bien de su comunidad y subsecuente desarrollo.

Tux, la mascota Linux, rodeado de los logos sistemas operativos libres e interfaces graficas de usuario libres

Citas del capítulo 4

1Serralde José María, extracto de entrevista (México D.F. en Mayo 2010).

2Eschoyez Martín, extracto de entrevista (Córdoba Argentina, Junio 2011).

3Me refiero a la salida de una versión por año o condicionar un software a un tipo de hardware como en el caso de OSX que sólo puede correr en equipos Mac o en programas que requieren de algunas especificaciones de Hardware para poder funcionar correctamente.

4FLOSS: Free, Libre, Open Source, Software que traducido al español: Software Libre de Fuentes Abiertas. También puede ser llamado FOSS y por lo tanto, es un sinónimo de Software Libre

5Aymeric Mansoux y Marloes de Valk , "Prefacio" en **Floss and art** ed. Aymeric Mansoux y Marloes de Valk (Francia, GOTO10 y OpenMute, 2008) 12

6 Mansoux y de Valk **Floss and art**, 12.

7El software libre en la docencia multimedia. UOC, del libro: Proceedings of the floss internacional conference 2007

8Floss and art pag 6

9Josephine Berry Slater, "Bare Code: Net art and the free software movement" en **Engineering Culture: Control and Commitment in a High-Tech Corporation** ed. Gideon Kunda (E.U., Temple University Press, 2006) 136.

10Ver capítulo 2 sobre el uso del software en la creación.

11Eschoyez Martín, extracto de entrevista (Córdoba Argentina, Junio 2011).

12Pagola Lila, extracto de entrevista (Córdoba, Argentina, Junio 2011).

13Dr. Ricardo J. Castello , Cr. Eduardo J. Gauna , Cra. Sandra Arónica . Software Libre, modelo de análisis de factibilidad económica-financiera. Informe del proyecto de investigación del 2004. Marzo 2005.

14"Russian schools move to Linux" **BBC News** (Octubre 2007 [citada en diciembre 2011]) disponible en: http://news.bbc.co.uk/2/hi/technology/7034828.stm

15"German universities migrate ti Linux" **Computer Weekly** (Agosto 2007 [citada en diciembre 2011]) disponible en: http://www.computerweekly.com/news/2240082699/German-universities-migrate-to-Linux

16Focus Editors "50 places linux running you might not expect" **Focus** (Marzo 2010 [citada en diciembre 2011]) disponible en: http://www.focus.com/fyi/50-places-linux-running-you-might-not-expect/

17Dr. Ricardo J. Castello , Cr. Eduardo J. Gauna , Cra. Sandra Arónica . **Software Libre, modelo de análisis de factibilidad económica-financiera. Informe del proyecto de investigación del 2004**. Marzo 2005. pag 4

18Dr. Ricardo J. Castello , Cr. Eduardo J. Gauna , Cra. Sandra Arónica . Software Libre, modelo de análisis de factibilidad económica-financiera. Informe del proyecto de investigación del 2004. Marzo 2005. pag 4

19Cristobal Cobo, "Aprendizaje de Código Abierto" en: **Plan Ceibal, Uruguay: una computadora por cada niño, los ojos del mundo en el primer modelo OLPC a escala nacional**. ed. Roberto Balaguer (Uruguay:Pearson 2009) 127

20Recordemos que aunque libre, no significa que sea gratuito, pero mucha de las veces lo es.

21Alejandro Miranda "Educación y Software Libre", en: **Construcción colaborativa del conocimiento, (**Universidad Nacional Autónoma de México, Instituto de Investigaciones Económicas, 2011) 235.

22Fernando da Rosa y Federico Heinz , **Guía práctica sobre Software Libre. Su selección y aplicación local en América Latina y el Caribe**. (Uruguay: UNESCO 2007), 54.

23Cristobal Cobo, "Aprendizaje de Código Abierto" en: **Plan Ceibal, Uruguay: una computadora por cada niño, los ojos del mundo en el primer modelo OLPC a escala nacional**. ed. Roberto Balaguer (Uruguay:Pearson 2009) 127

24July E. Jiménez O. Y otros, "Software libre en la educación" **Apropiación Social de las TICs** , ([citado en noviembre 2010]) ed. Colombia aprende, la red del conocimiento. Disponible en: www.colombiaaprende.edu.co/html/mediateca/1607/articles-108475_archivo.pdf

25Juan Cristóbal Cobo, "Conocimiento, creatividad y software libre: una oportunidad para la educación en la sociedad actual" **UOC Papers**. N.o 8. UOC. (2009 [citado en 2010]) disponible en: http://www.uoc.edu/uocpa-

26 Cobo, Aprendizaje... pag 127.

27Caiazza Fabricio, extracto de entrevista (Rosario Argentina, Mayo 2011).

28Andreas Meiszener, Rüdiger Glott, Sulayaman K. Sowe, "Preparando la generación red: Enseñanzas extraídas del software libre y sus comunidades" Global University Network for Innovation, (Septiembre 2008 [citado en Noviembre del 2010]) ed. Universidad Politécnica de Catalunya, UNESCO, United Nations University, disponible en: http://web.guni2005.upc.es/news/detail.php?chlang=es&id=1251

29Carlos Casado Martínez, Marín Amatller y otros "El software Libre en la docencia multimedia, en: Procedings of the FLOSS International Conference 2007 ed. Rafael Rodríguez Galván y Manuel Palomo Duarte, (Cádiz, España: Universidad de Cádiz, 2007) 276.

30da Rosa y Heinz, "Guía práctica," 53.

31da Rosa y Heinz, "Guía práctica," 53

32Maccur, "Entrevista con John 'Maddog' Hall" Master Magazine (Enero 2009 [citada en octubre 2010]) disponible en: http://www.mastermagazine.info/articulo/13544.php

33Mansoux y de Valk , Floss and art, 12

34Eschoyez Martín, extracto de entrevista (Córdoba Argentina, Junio 2011).

35Pagola Lila, extracto de entrevista (Córdoba Argentina, Junio 2011).

36Cobo, "Aprendizaje de," 128.

37Cobo, "Aprendizaje de," 131.

38Mariel Sangrà, "El software libre entra a la aulas", Sala de prensa de la Universidad Oberta de Cataluña (Septiembre 2006 [citada en Octubre del 2010]) ed. Universitat Oberta de Catalunya, disponible en: http://www.uoc.edu/portal/catala/la_universitat/sala_de_premsa/reportatges/2006/programari_lliure.html

39Casado, Amatller y otros "El software Libre en," 276.

40Casado, Amatller y otros "El software Libre en," 275.

41Casado, Amatller y otros “El software Libre en,” 276.

42Santorcuato José, entrevista via chat el 21 de octubre del 2010 y reforzada con la entrevista en vivo en Santiago de Chile en Mayo del 2011.

43Sangrà “El software libre entra,”

44Miranda, Educación y...pag, 235

45Miranda, Educación y...pag, 235

46Miranda, Educación y...pag, 234

47Miranda, Educación y...pag, 235

48Sangrà “El software libre entra,”

49Pedro Soler, “Artist and Free Software, an Introduction” en Floss and art ed. Aymeric Mansoux y Marloes de Valk (Francia, GOTO10 y OpenMute, 2008) 16

50Miranda, Educación y...pag, 245

<title>

Conclusiones

</title>

En base a lo documentado en el capítulo 1, encontramos que el diseño gráfico y la creación gráfica digital estuvieron rodeados desde sus inicios de mitos que permanecen hasta hoy en día, (Mac, Illustrator, Photoshop, etc asociadas como única herramienta eficaz para llevar acabo los procesos de producción gráfica digital) y que la conjunción de tres tecnologías: Post Script, Mac y el Page Maker marcaron el rumbo del diseño gráfico hecho por computadora, abriendo la posibilidad de que cualquier persona, sin necesidad de ser profesional del diseño, hicieran diseño de escritorio o Desktop publishing.

Sin embargo, estas herramientas que en un principio facilitaron y aceleraron el trabajo del diseñador y eventualmente del artista visual digital, a la larga, representaron una caja negra que imposibilita al creador conocer el proceso total de su obra. Mientras se proporciona nuevas maneras de crear, también, la caja negra, las limita a sus capacidades técnicas. Así el profesional que usa el software ve limitado la creación de objetos culturales a las capacidades de la herramienta que utiliza.

Este hecho, aleja al creador de su propia obra, impidiéndole replicar ese proceso en un programa distinto, pues desconoce el funcionamiento del aparato con el que fue hecho. Por lo tanto, es necesario el esclarecimiento de la caja negra que representa el software como herramienta para la creación, para que exista un conocimiento más profundo de la misma y por lo tanto un control más sustancial por parte del profesional y con ello su acercamiento a la obra. Así mismo, éste conocimiento permitirá desmitificar la idea del aparato que lo hace todo, el software que resuelve de manera "mágica" los problemas del diseñador con sólo aprender a "usarla" dominado la técnica. Permitiría eventualmente, una reflexión acerca de la herramienta que utiliza.

Así pues, observamos que, el software libre y su posibilidad de código abierto, se vuelve entonces una "caja transparente" que permite que podamos desmitificar el aparato a través del ya mencionado, esclarecimiento de la caja negra que representa el software privativo. El software libre por lo tanto, nos permitiría estar mas cerca de nuestra obra, conocer a profundidad cómo fue hecha y con ello replicar el proceso, acercarse y apropiarse del proceso del objeto cultural.

Comprobamos de manera parcial la hipótesis de que existe laposibilidad de prescindir del uso de cualquier tipo de software privativo para la creación gráfica digital de manera exitosa. Fue posible crear sólo con software libre en los siguientes casos: primero, de manera personal basado en mi experiencia (ejerciendo profesionalmente la disciplina de diseño gráfico con S.L. por más de 3 años) y segundo, en el caso de otros profesionales que fueron entrevistados y que ejercen su disciplina usando sólo programas libres. Sin embargo, en algunos casos no resulta viable su uso exclusivo, ya que algunas acciones aún no se pueden resolver de manera eficaz y de calidad con software libre. Tal es el caso de la edición de video, la cual aún no está lo suficientemente avanzada, o algunas acciones específicas de diseño editorial, como la inserción de los pies de página. También hay que tomar en cuenta que los avances y mejoras de un programa dependen en gran medida a la su comunidad de usuarios, por lo que ser usuarios de S.L. nos convierte en entes activos capaces de cambiar y aportar para que nuestra herramienta sea cada vez mejor, y resulta, a la larga ser parte de nuestra responsabilidad; se trata de un cambio de paradigma del usuario-cliente cautivo. Así mismo, eso nos da la certeza que cualquier problema o deficiencia en el S.L. cambiará eventualmente, cosa que podría no ser así en el caso del S.P. donde se supeditan los cambios a las decisiones de una "única" empresa.

De igual forma, es importante considerar que cualquier cambio o uso inicial de este software será complicado en términos de adecuación a la nueva herramienta (tal como sucede cuando se usa Windows con regularidad y Osx nos resulta nuevo y diferente).

Dentro del grupo de 12 entrevistados y en base a mi experiencia personal, detecto que gran parte de los usuarios de software libre para la creación gráfica lo hacen por convicción y no necesariamente por encontrar "mejores" soluciones técnicas. Se acerca a él en base a una búsqueda y cuestionamiento de la herramienta privativa que utiliza.

Comprobamos que es recomendable el uso del software libre, debido a sus múltiples beneficios, dentro de los que se encuentran: bajos costos, uso de estándares (lo cual permite la permanencia en el tiempo de la obra digital, garantizar la posibilidad de abrir un archivo aunque un día el programa deje de existir), seguridad, posibilidad de trabajo en equipo, afinidad con conceptos como libertad y colaboración; también permite mayor control de la obra en caso de los artistas digitales. Representa una alternativa viable para ONG's, asociaciones civiles, educación, creación artísticas, proyectos independientes, cultura, etc. Y en general, por que tal como se explicó en el capítulo 2 representa menos obstáculos para la libre creación y circulación del objeto cultural.

Fue posible comprobar en el capítulo 3 la hipótesis de que el software libre permite ejercer profesional, creativa y eficazmente nuestra disciplina.Encontramos que la posibilidad del S.L. de modificarse y estudiarse, hace de éste una herramienta que puede ser conocida a fondo y con ello, estimular la curiosidad para descubrir nuevas formas de resolver problemas inherentes al programa; además, se promueve la colaboración como fuente primordial de generación de ideas en donde una cultura orientada a compartir el conocimiento, que no cierra el intercambio, se convierte en una cultura que tiene más posibilidades de crear, innovar y de crecer. La colaboración, el estímulo de curiosidad, la libertad, el intercambio de conocimientos, así como la posibilidad de generar soluciones innovadoras debido a la apertura del código. El software libre me permitió ser más consciente de mi diseño y me invitó a explorar nuevas posibilidades de creación.

En base a las entrevistas de diseñadores y artistas y reflexiones posteriores a la finalización de este trabajo, puedo deducir que el software libre tiene mayores posibilidades de penetración en el terreno de la creación artística que en la de diseño gráfico, debido a que en el arte existe un ambiente propicio para la experimentación y en ocasiones, necesita de la solución de problemas muy específicos que a su vez requieren de un control mucho mayor de la herramienta en cuestión. Además que ambos, S.L y arte, sostienen varias premisas a fines, entre ellas el concepto de libertad. Además, de lo accesible que resulta debido a su bajo costo, ayudando a que los propios artistas disminuyan sus gastos de producción. En cambio, en el diseño gráfico, las posibilidades de ven limitadas a resolver un diseño de manera rápida y en el menor tiempo posible, pues muchas veces se está a las expensas de un cliente y de sus tiempos. Es difícil, entonces, que el diseñador gráfico se acerque a una nue-

va herramienta dentro del vertiginoso proceso de producción que mantiene en general. Eso me lleva a decir que el software libre es para todos pero no todos son para el software libre. Habrá quienes no están interesados en su uso o que probablemente no encuentren interés en la filosofía que promueve su uso y mucho menos estén interesados en pertenecer a una comunidad que mejora su propia herramienta. Será una herramienta que debe de ir acompañada de una comprensión y afinidad con la propuesta ética y filosófica que subyace, y de ese modo, ser usada por convicción e ideología.

Quizá en la medida en la que un diseñador decida que el objeto cultural que ha creado requiere de una conservación, o el proyecto requiera de un libre flujo de información a través del fomento de uso de estándares, entonces, pueda interesarle migrar de herramienta privativa a una libre.

Sin embargo, esto no siempre ocurre así y el diseñador, quizá contrario al artista, no buscará, al menos no de manera inmediata, cambiar sus herramientas diariamente usadas por unas nuevas en miras de ser más coherente con su trabajo. Es por eso que, para que el diseñador se acerque al uso del software libre, encuentro que la única manera eficaz será la de enseñarlo en su etapa de formación profesional (o inclusive su reconocimiento anterior, en la educación primaria y secuendaria).

Para usar software libre se requiere de una motivación, pero también, puede ser el caso que se le de a conocer al estudiante de diseño a la par del software privativo y se le de así la oportunidad de **elegir** qué herramienta digital le viene mejor. Además de que, con ello, se le estaría enseñando a entender procesos (aplicados a distintos programas) y no el uso específico de un determinado y único software. Limitar al estudiante de diseño a usar "un producto" en lugar de desarrollar habilidades y hacer que entenda el funcionamiento y comportamiento de las herramientas, equivaldría a decir que en las clases de fotografía sólo se usará una marca y modelo de cámara sin opción a otra, todos los alumnos tendrían que comprar una Canon A3, por ejemplo por ser la "más usada" la de "mejor calidad" y las "mas conocida".

Así mismo, observamos que el uso del software libre es muy viable en el entorno educativo ya que fomenta la colaboración y el trabajo en equipo, así como la creatividad y procesos críticos y puede ser adaptado a cada necesidad (traducción de programas a idiomas de algunas minorías, ayuda a discapacidad, personalización según el alumno, etc)

Enseñar el software libre a la par del software privativo para la creación visual digital, resulta una solución eficaz para el problema de la caja negra que representa el software privativo para el creador, el poco conocimiento del proceso del diseñador en su obra y el desconocimiento de su herramienta. La utilización y enseñanza del software libre, no se limita sólo al uso de una herramienta alternativa, sino que permite una reflexión en cuanto a nuestra manera de crear digitalmente se refiere, ofrece una solución a la problemática planteada en el capítulo 1. El software libre no sólo es bueno y útil para la creación de objetos culturales, también lo es para acercar al diseñador gráfico y artista visual, a conocer el proceso de su obra y hacerlo mucho más crítico respecto a esto.

Además de que el crecimiento exponencial del software libre en los últimos años, hace suponer que su crecimiento seguirá en aumento y eso nos lleva a volcarnos en un nuevo eje del conocimiento que representará mas posibilidades y entornos creativos donde el profesional de la creación gráfica, artista y comunicador visual, puede desarrollarse y encontrar un terreno fértil de creatividad.

Por último, pero no menos importante, el uso del software libre trae consigo el acercamiento al movimiento social que lo subyace, el cual supone el modelo de una construcción colaborativa del conocimiento, el beneficio común, el avance en comunidad y la no restricción de libertades.

El uso del software libre me llevó no sólo a conocer una nueva y diferente herramienta digital, sino que me acerco a la práctica de la cultura de la libre distribución, del licenciamiento permisivo, el uso de Creative Commons para licenciar mis trabajos, la incursión en el diseño colaborativo, la cita estricta de las fuentes y sobre todo, a la idea de que el conocimiento es construido y generado por todos.

<title>
Anexos
</title>

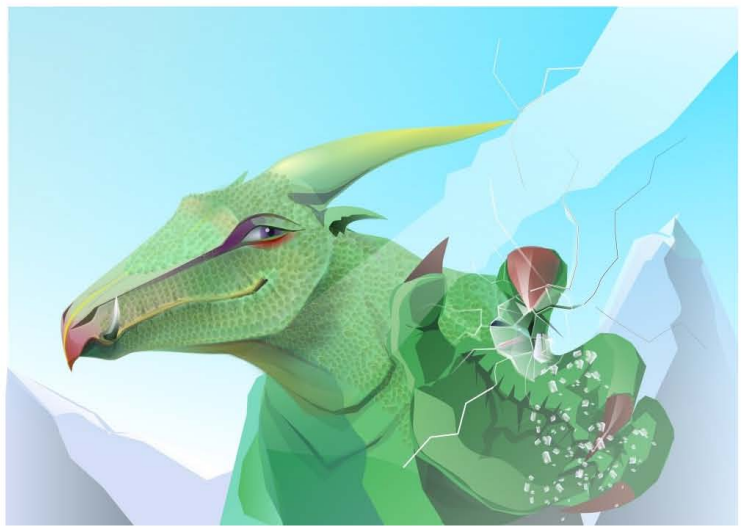
6.1. Aplicaciones libres para el diseño

Diseño e ilustración vectorial

Aplicaciones Privativas: Adobe Illustrator, Corel Draw.

Aplicaciones Libres: Inkscape (www.inkscape.org [1]), SK1 (http://sk1project.org/ [2])

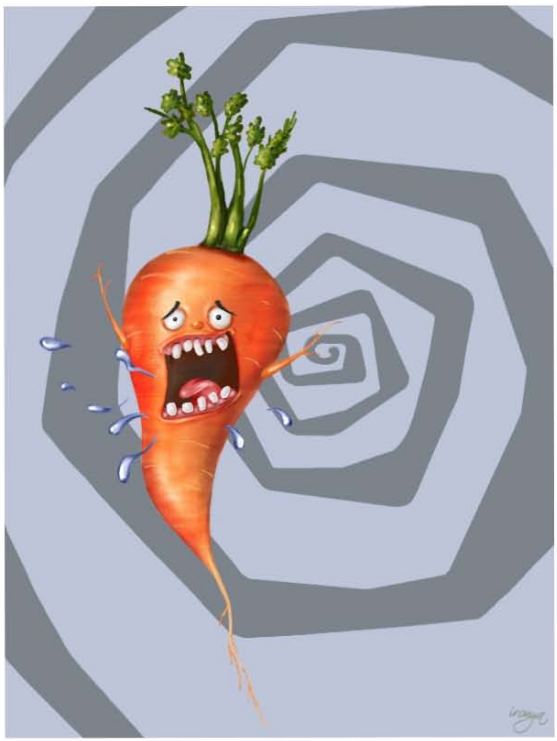
INKSCAPE: La alternativa libre a ILLUSTRATOR es un avanzado programa para la creación y edición de imágenes vectoriales. Como usuaria frecuente de INKSCAPE, puedo decir que resulta ser un programa muy intuitivo para usuarios que nunca han utilizado software de generación de vectores, y un tanto confuso al principio para diseñadores o artistas familiarizados con ILLUSTRATOR, sin embargo, una vez que se ha superado la primera impresión del programa, resulta ser muy sencillo e incluso, más sencillo de manipular que su homologo privativo. Resuelve muchos problemas de una manera más eficaz, como la importación de manera selectiva de algunos vectores a PNG, o bien, del diseño completo, así como la manipulación de su resolución y tamaño de manera también muy cómoda y fácil. Un archivo SVG creado por INKSCAPE puede ser visualizado en cualquier navegador de Internet. También permite salida de archivos a PDF, SVG, EPS. El principal problema es que no tiene soporte CMYK por lo que hay que importar primero un PNG en RGB para luego convertirlo a CMYK en el programa GIMP



Obras realizadas usando Inkscape. De ariba a abajo, de izquierda a derecha; Sin título de Andrea Sagardoy, Sin título de Luciano Lourenço y Sin título, Konstantin R.

Obras realizadas usando Inkscape. De abajo hacia arriba y de izquierda a derecha; Drums, de Guilles Pinard, Avión, de Guilles Pinard, Foco, de Joacl Istgud

SK1: Es un programa de ilustración de código abierto. Actualmente GNU / LINUX es la plataforma principal de desarrollo, pero ya se ha desarrollado para Win32 y Mac OS X. sK1 permite algunas funciones profesionales de edición, tales como color CMYK, separaciones, la gestión de color ICC y la prensa preparado para la salida PDF.



Maja Kocón, obra realizada con Gimp

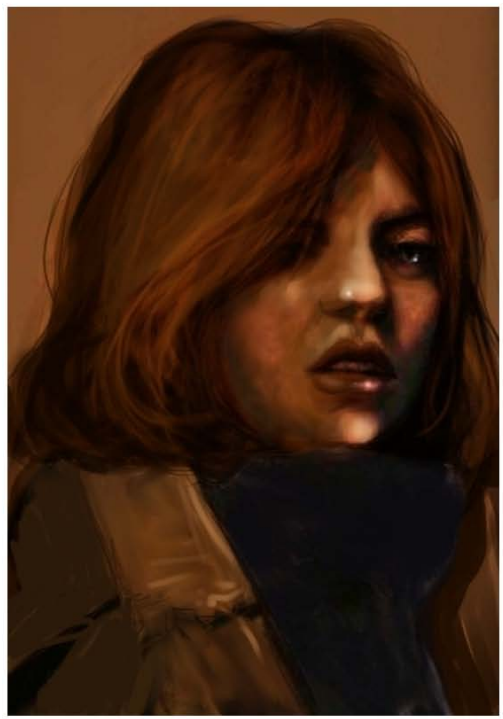
Tratamiento de imágenes

Aplicaciones Privativas: Adobe Photoshop

Aplicaciones Libres: GIMP (www.gimp.org [3]), Cinepaint (www.cinepaint.org). Separate+ (plug in de separación CMYK para GIMP)

GIMP: Gimp o Gnu Image Manipulation Program, es un programa para edición de gráficos en mapa de bits y fotografías digitales, la alternativa libre a PHOTOSHOP, aunque su interfaces son muy diferentes entre si al igual que la forma de resolver problemas gráficos. Algunos inconvenientes que he experimentado en su uso, es que no existe aún una manera eficaz y rápida para deformar una imagen de tal forma que ésta se adecué a una superficie ya deformada, el llamado WARP en Photoshop. Para ello, se tiene que recurrir a deformarlo en perspectiva y luego el filtro de deformar por curva, pero resulta muy tardado pues no se tiene un control suficiente a dicha deformación. De igual forma, para convertir una imagen en color CMYK se tiene que instalar el plug in separate+. Si se es un usuario habitual de Photoshop, el hecho de que GIMP sea multiventanas, seguramente le parecerá poco funcional.

Sin título, de Blacktoon, obra realizada con Gimp



Sin título, de Arwassa, obra realizada con Gimp

CINEPAINT: Programa de código abierto para pintura y retoque de imágenes de mapa de bits de las películas. Se trata de un FORK de la versión 1.04 de GIMP. Tuvo cierto éxito como una de los primeras herramientas libres desarrollado para efectos visuales en el cine y el trabajo de animación.

Episodio I, la guerra de los clones y Scooby Doo, películas que usaron Cinepaint en parte de su realización

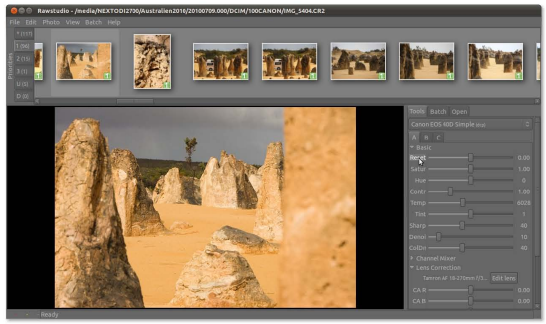
Revelado Raw

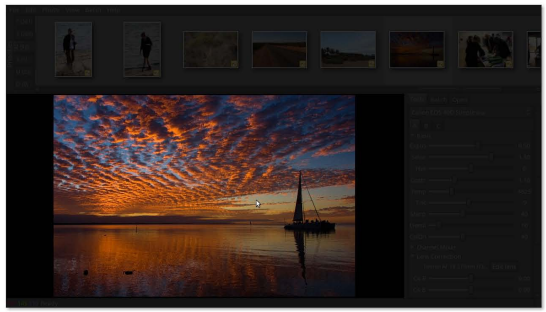
Aplicaciones Privativas: Adobe Photoshop, Aperture

Alternativas Libres: RawStudio (www.rawstudio.org [4]) RawTherapee (www.raytherapee.com [5]), QTPFSGUI (Luminance HDR), Ufraw (plug in para GIMP)

RAWSTUDIO: Programa libre para leer y manipular imágenes RAW desde cámaras digitales. Convierte archivos RAW a JPEG, PNG o TIF y desde la aplicación puede subirse fotos directamente a Picasa, Flickr y Facebook.

RAWTHERAPEE: De igual forma, se trata de un programa para tratamiento de imágenes formato RAW. Permite obtener la mayor cantidad de detalles gracias a los algoritmos: Amaze, DCB, rápido, AHD, EAHD, HPHD y VNG4. Posee un manejo avanzado del color del balance de blancos para HSV (Tono-Saturación-valor) y las curvas de la gestión del color. Como métodos de eliminación de ruido: luminancia, cromancia. Máscara de enfoque, RL desconvolución, el contraste por los niveles de detalle.





En la parte superior, captura de pantalla del programa Rawstudio, a la izquierda, captura de pantalla del programa Rawtherapee

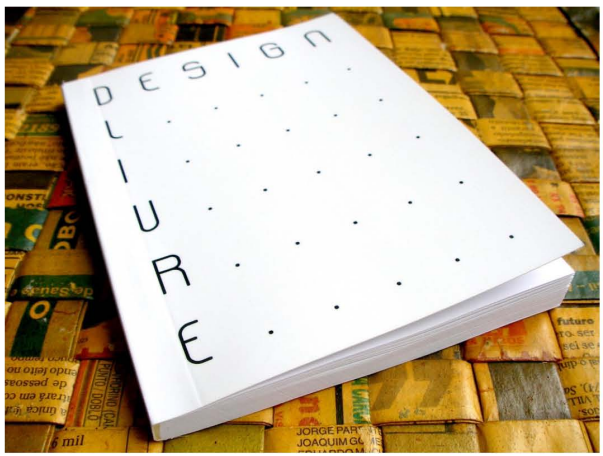
Diseño Editorial

Aplicaciones Propietarias: Adobe Indesign, Quark Express

Alternativas Libres: Scribus [6]

SCRIBUS: Homólogo de INDESIGN es un programa libre para diseño editorial muy avanzado que permite trabajar tanto texto, imágenes y gráficos vectoriales. Es sumamente útil para el armado de cajas tipográficas y diseño de publicaciones. El armado editorial de esta tesis esta realizado en este programa. Aún presenta problemas con las notas al pie de página pero tiene un buen manejo de CMYK.

Desig livre. http://designlivre.faberludens.com.br/ [7]. Libro sobre proyectos colaborativos.



Nüüs vu Mullis
http://ngiger.dyndns.org/vvm/nüüs_vu_mullis.pdf [9]

CD, Les jetés de l'encore / Bologne
http://www.lesjetesdelencre.com [8]

Choices Magazine http://choicesmagazine.co.uk/

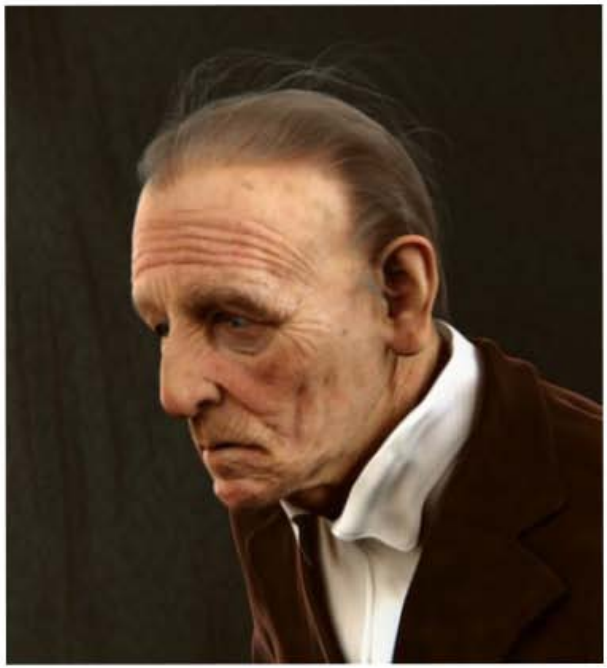
Modelado y Animación 3D

Aplicaciones Privativas: Maya, 3D Studio Max, Vray

Alternativas Libres: Blender [10],Yafaray [11], LuxRender [12]

BLENDER: Uno de los programas libres más avanzados y exitosos entre la comunidad de creadores gráficos digitales. Permite el modelado, animación y creación de gráficos 3D. Particularmente, blender ha tenido un desarrollo considerable debido a la interacción entre artistas y programadores, además de que en un inicio era un programa privativo y ya su desarrollo se encontraba muy avanzado.

El cortometraje animado Big buk bunny, hecho en su totalidad con software libre, entre ellos, el programa de modelado y animación 3D, Blender.



De arriba a abajo y de izquierda a derecha; Old Guy, de Kamil (maqs) Makowski, Dea della Fertilita, de Davide Maimone y Frakas Dream, de Pablo Vazquez

Otras aplicaciones:

Administración de Tipografías

Aplicaciones Propietarias: ATM, Suitecase, Font Explorer

Alternativas Libres: Fonty Python [13], FontMatrix [14]

Pintura Digital

Aplicaciones Propietarias: Photoshop, Painter, Art Rage

Alternativas Libres: Gimp Paint Studio [15], MyPaint [16], Krita [17], Alchemy [18]

Motion Graphics

Aplicaciones Propietarias: After Effects

Alternativas Libres: Blender [19]+ XCF Layers Suite [20]

Diseño Web

Aplicaciones Propietarias: Adobe Dreamweaver, Flash

Alternativas Libres: Aptana Studio [21], Quanta+ [22], frameworks Ajax (mootools, jQuery, Prototype)

Selección de color y software libre

Para entender mejor la razón por la cual algunos programas libres no ofrecen de manera nativa el uso del CMYK, explicaré de manera muy breve algunas cuestiones de gestión de color en el entorno digital.

Sabemos que las pantallas de las computadoras trabajan en RGB (Red, Green, Blue) que es una mezcla aditiva de emisión de luz, mientras que en los impresos se usa el CMYK (Cyan, Magenta, Yellow, Black), que se trata de una mezcla sustractiva de recepción de luz, esto bajo la lógica que la mezcla de cyan, magenta y amarillo obtiene negro, así mismo que con un porcentaje de cada color, daría el tono buscado. En teoría la mezcla de 100% 100% 100% de CMY debería dar negro, sin embargo, en la práctica, las tintas vienen con impurezas de fábrica y la mayoría de las veces, la mezcla sólo da un tono café oscuro pero no un negro puro, es por eso que se adicionó el Black al modelo CMYK, además que el negro reemplazando a la mezcla de 3 colores, resulta mucho más económico a la hora de imprimir1.

El CMYK existe por la limitación y el problema de no lograr el negro perfecto y una gama de grises negros. Si a esto le aunamos a que cada máquina responde de manera distinta y que el CMYK depende mucho del dispositivo de trabajo de imágenes para la impresión, el tema se hace más complejo. Como lo diría Guillermo Espertino, diseñador gráfico argentino en su blog: "En el principio del diseño por ordenador no existía la gestión de color como hoy la conocemos y por ese motivo la única manera de trabajar para impresión era especificando los valores de CMYK que se iban a imprimir a mano.

De ahí viene la tradición de trabajar en CMYK para impresión. Antes no había una manera del todo fiable de convertir entre espacios de color (y si la había era costosísima, y no había expertos a quien preguntarle)"2.

Los programas para gráficos incorporaron el modo CMYK como una forma de facilitar ese trabajo dada la práctica de trabajar todo en este modelo, el cual ya estaba demasiado arrai-

gado para que los usuarios consideraran otra cosa. Sin embargo, el sólo convertir un diseño a CMYK no basta para que los colores sean respetados según el equipo de impresión, por que existen además los "perfiles de color" que son estándares de facto usados por la industria en un intento por homogeneizar y "ponerse" de acuerdo con los modos de color CMYK de las impresiones. Así pues, si la imprenta pide que el diseño esté en CMYK, pero en muchos casos no nos dice en "qué tipo de CMYK", es muy probable que la propia imprenta no sepa qué perfil de color utiliza.

Por ejemplo, ¿qué pasa cuando el perfil con el que trabajamos y que muchas veces no conocemos, no corresponda al perfil de color utilizado por el impresor, que también, en muchos casos, tampoco lo sabe?. Para que la imprenta sepa con qué perfil trabajamos este tiene que estar incrustado en el archivo, de no ser así, el impresor desconocerá la combinación de colores que estamos buscando. Si el archivo tiene el perfil incrustado, la imprenta tendrá la opción de convertir a su propio perfil o bien, desechar el perfil incrustado y asignarle su propio perfil de trabajo, de la segunda forma, nunca podremos estar seguros de cómo será el resultado. La primera opción que es la mas recomendable, hará que el sistema de gestión de color adapte los colores de perfil de origen y destino de la forma más adecuada para lograr la apariencia correcta dentro de los parámetros posibles.

En base a lo anterior, podría sustentar que una buena práctica sería trabajar con lo que se conoce como "late binding", es decir, dejar para el final la conversión a CMYK.

La tendencia actual, acentuada por el creciente uso de sistemas DI (direct imaging) y CPT[3], es dejar el trabajo de separación al RIP[4], y que éste se encargue de la conversión a CMYK. En ese caso se sugiere trabajar directamente en RGB, sin preocuparse en absoluto por convertir a CMYK. El hardware y software del dispositivo de salida se ocupan de convertir las imágenes según el perfil correcto, y se logra la mejor reproducción posible de nuestro original RGB en el impreso[5].

Es por eso, que en el caso de mi experiencia convirtiendo al final mi diseño a CMYK por medio de un plugin, no representó problema alguno a la hora de la impresión. Se obtuvieron las 1000 copias de cada una de las infografías, y se distribuyeron sin problema.

Citas de los Anexos

1Guillermo Espertino "Introducción a la gestión de color" Ohweb (16 de Enero del 2010 [citada en Enero 2012] ed. Guillermo Espertino) disponible en http://www.blog.ohweb.com.ar/?p=190

2Extraído de entrevista vía mail a Guillermo Espertino.

3Computer to Plate o simplemente CtP es una tecnología de artes gráficas por medio de la cual las placas de impresión Offset son copiadas por máquinas manipuladas directamente de una computadora

4Raster image processor

5Guillermo Espertino "RGB y CMYK, late binding vs. Early binding" Ohweb (20 de abril del 2010 [citada en Enero 2012] ed. Guillermo Espertino) disponible en http://www.blog.ohweb.com.ar/?p=220

Links

[1] http://www.inkscape.org/

[2] http://sk1project.org/

[3] http://www.gimp.org/

[4] http://www.rawstudio.org/

[5] http://www.raytherapee.com/

[6] http://www.scribus.net/

[7] http://designlivre.faberludens.com.br/

[8] http://www.lesjetesdelencre.com/

[9] http://ngiger.dyndns.org/vvm/nÃ¼Ã¼s_vu_mullis.pdf

[10] http://www.blender.org/

[11] http://www.yafaray.org/

[12] http://www.luxrender.net/

[13] http://otherwise.relics.co.za/wiki/Software/FontyPython/

[14] http://fontmatrix.net/

[15] http://code.google.com/p/gps-gimp-paint-studio/

[16] http://mypaint.intilinux.com/

[17] http://www.koffice.org/krita/

[18] http://al.chemy.org/

[19] http://www.blender.org/

[20] http://alexvaqp.googlepages.com/xcf_layers_suite

[21] http://www.aptana.org/

[22] http://quanta.kdewebdev.org/index.php

<title>

BIBLIOGRAFÍA

</title>

Libros:

Alsina, Pau. "La intersección entre las artes y las tecnologías de la información y comunicación". En: E-cultura Les noves tecnologies aplicades a la gestió de la cultura, editado por Melba G. Claudio González. Barcelona: Associació de professionals de la gestió cultural de catalunya, 2006.

Benjamin, Walter. El autor como productor. México D.F: Itaca, 2004.

Barrón, Frank. Creativity person and creative process. New York: Holt, Rinehart, & Winston 1970.

Berry Slater, Josephine. "Bare Code: Net art and the free software movement". En Engineering Culture: Control and Commitment in a High-Tech Corporation ed. Gideon Kunda .E.U.: Temple University Press, 2006.

Casado Martínez, Carlos, Marín Amatller y otros."El software Libre en la docencia multimedia". En: Procedings of the FLOSS International Conference 2007, editado por Rafael Rodríguez Galván y Manuel Palomo Duarte. España: Universidad de Cádiz, 2007.

Castello Ricardo, Gauna Eduardo, Arónica Sandra . "Software Libre, modelo de análisis de factibilidad económica-financiera". Informe del proyecto de investigación del 2004. Editado por Universidad de Córdoba, 2005.

Cerezo, José María. Diseñadores en la Nebulosa, el diseño gráfico en la era digital. Madrid: Biblioteca nueva, 1999.

Cristobal Cobo. “Aprendizaje de Código Abierto”. En: Plan Ceibal, Uruguay: una computadora por cada niño, los ojos del mundo en el primer modelo OLPC a escala nacional, editado por Roberto Balaguer. Uruguay: Pearson 2009.

Cobo, Cristóbal. "Conocimiento, creatividad y software libre: una oportunidad para la educación en la sociedad actual" UOC Papers. N.o 8. UOC. (2009 [citado en 2010]) disponible en: http://www.uoc.edu/uocpapers/8/dt/esp/cobo.pdf

da Rosa Fernando y Federico Heinz. Guía práctica sobre Software Libre. Su selección y aplicación local en América Latina y el Caribe. Uruguay: UNESCO 2007.

Flusser, Vilem. Hacia una Filosofía de la fotografía.México: Trillas, 1990.

Lessing, Lawrance. Cultura Libre. Traducción de Antonio Córdoba/Elástico. E.U: Penguin Books, 2004.

Machado, Arlindo. “Repensando a Fluser”. En: El paisaje Mediático. Sobre el desafío de las poéticas tecnológicas. Buenos Aires: Libros del Rojas. 2000.

Mansoux, Aymeric y Marloes de Valk . “Prefacio”. En Floss and art, editado por Aymeric Mansoux y Marloes de Valk. Francia: GOTO10 y OpenMute, 2008.

Martínez Meave, Gabriel y otros. Diseño, tipografía y lenguaje.México D.F: Editorial Designio, 2004.

Megs, Philip y Purvis, Alston W. Historia del Diseño Gráfico. México: RM Verlag, 2009.

Miranda, Alejandro. “Educación y Software Libre”, en: Construcción colaborativa del conocimiento, México D.F.: Universidad Nacional Autónoma de México, Instituto de Investigaciones Económicas. 2011.

Pagola, Lila. “Software libre, una caja abierta y transparente”. En: Instalando: Arte y cultura digital, editado por Troyano: Ignacio Nieto, Italo Tello, Ricardo Vega . Chile: Lom Ediciones, 2007.

Pelta, Raquel. Diseñar hoy. Barcelona: Paidos Diseño 01, 2004.

Rhodes, Mel. “An Analysis of Creativity”. The Phi Delta Kappan. Vol. 42, No. 7 (Abril 1961).

Rodriguez Estrada, Mauro. Manual de Creatividad. Los procesos psíquicos y desarrollo. México: Trillas 2006.

Sánchez Ramos, Ma. Eugenia. “La revolución digital en el diseño gráfico” Actas de Diseño, Vol. 7 Año IV (Julio 2009).

Silva López, Blanca. Creatividad y pensamiento crítico. México: Trillas, 1998.

Soler, Pedro. “Artist and Free Software, an Introduction”. En Floss and art, editado por Aymeric Mansoux y Marloes de Valk. Francia: GOTO10 y OpenMute, 2008.

Smiers, Joost. Un mundo sin Copyrigth, arte y medios en la globalización. Traducido por Julieta Barba y Silvia Jawerbaum. Barcelona: Gedisa. 2006.

Stallman, Richard. Software libre para una sociedad libre. Madrid: Traficantes de sueños, 2004.

Sutherland, Ivan. Sketchpad: A mal-machine graphical communication system. Editado por University of Cambridge Computer Laboratory. Disertación de grado, Massachusetts Institute of Technology , 1963.

Vega Mora, Ricardo. “Tecnología, panoramas y paradigmas: cambios en la producción digital de la

era digital”. En Buscando Señal, lecturas sobre nuevos hábitos de Consumo Cultural, editado por Mariano Barbieri. Córdoba: Ediciones del Centro Cultural España Córdoba, 2009.

Internet

“Timeline of Computer History-1983”. Computer History Museum. (2006 [citado el 7 de Junio del 2009]): disponible en http://www.computerhistory.org/timeline/?year=1983

“Adobe Photoshop”. Wikipedia, La enciclopedia libre (28 mayo 2009 [citado el 6 de junio del 2009]): disponible en http://es.wikipedia.org/wiki/Adobe_Photoshop.

"Visión general del proyecto GNU". Historia del proyecto GNU. (Enero 2008, [citada el 6 de junio del 2009]), GNUOpening Systems: disponible en http://www.gnu.org/gnu/gnu-history.es.html [1]

McAllister, Neil. "Devices provide a fertile new ground of Linux". Computerworld. (18-Sep-2006 [citado el 7 de Julio del 2009]): disponible enhttp://computerworld.co.nz/news.nsf/tech/2E968937ADA351D8CC2571EA0013F01B

Kroah-Hartman, Greg, Jonathan Corbet y Amanda McPherson. "Linux Kernel Development, How Fast it is Going, Who is Doing It, What They are Doing, and Who is Sponsoring It". (August 2009 [citado en agosto del 2010] The Linux Fundation): dispobible en www. [2]linux [3]foundation.org/sites/main/files/publications/whowrites [4]linux [5].pdf [6]

Red Hat. "Open Source Activity Map – 2008". (2009 [citada en octubre 2009] Red Hat) editado por Red Hat Enterprise Linux: disponible en http://www.redhat.com/about/where-is-open- [7]source/activity/ [8]

Serralde, José. "Opera, norteños y software libre (I)". Sonoblog, música, tecnología, cultura libre y sonidos por dentro.(10 de marzo del 2010 [citado en septiembre 2010]) editado por José Serralde: disponible en http://blog.joseserralde.org/opera-video-nortenos-y-software-libre [9]

"Open Standars". Document Freedon Day.(Primavera 2010 [citado en septiembre 2010]) editado por FSFE: disponible en http://documentfreedom.org/2011/os.en.html

Puccio Gerard. "Two Dimensions of Creativity: Level and Style". The International Center for Studies Increativity. (1999 [citada en septiembre 2010]) editado por Bufalo State, State Universuty of N.Y.: disponible en http://www.buffalostate.edu/orgs/cbir/readingroom/html/Puccio-99a.html

Maccur. "Entrevista con John 'Maddog' Hall". Master Magazine. (Enero 2009 [citada en octubre 2010]): disponible en http://www.mastermagazine.info/articulo/13544.php [10]

Mariel Sangrà. "El software libre entra a la aulas". Sala de prensa de la Universidad Oberta de Cataluña. (Septiembre 2006 [citada en Octubre del 2010]) editado por Universitat Oberta de Catalunya: disponible en http://www.uoc.edu/portal/catala/la_universitat/sala_de_premsa/reportatges/2006/programari_lliure.ht [11]ml [12]

Jiménez, July y otros. “Software libre en la educación”. Apropiación Social de las TICs.([citado en noviembre 2010]) editado por Colombia aprende, la red del conocimiento: disponible en www.colombiaaprende.edu.co/html/mediateca/1607/articles-108475_archivo.pdf [13]

Meiszener, Andreas, Rüdiger Glott, Sulayaman K. Sowe. “Preparando la generación red: Enseñanzas extraídas del software libre y sus comunidades” Global University Network for Inovation. (Septiembre 2008 [citado en Noviembre del 2010]) editado por Universidad Politécnica de Catalunya. UNESCO, United Nations University: disponible en http://web.guni2005.upc.es/news/detail.php?chlang=es&id=1251 [14]

“Historia del Post Script”. Digitecnia(Citada el 30 de marzo del 2011): disponible en http://www.icono-computadoras-pc.com/postscript.html [15]

Shaughnessy, Adrian. “Laptop Aesthetics”. Eye Magazine. 49, 2003. (Otoño 2003 [citada en mayo del 2011]): disponible en http://www.eyemagazine.com/feature.php?id=71&fid=471

Flores, José. “Mozilla Firefox, traducido a lenguas prehispánicas”. Alt1040.(16 de Agosto de 2010 [citada en Julio del 2011]) editado por. Hipertextual: disponible en: http://alt1040.com/2010/08/mozilla-firefox-traducido-a-lenguas-prehispanicas [16]

Levien, Ralph. “Ralph Levien's Thesis”. Sitio web personal de Ralph Levien (Septiembre 2009 [citada en Julio del 2011]) editado por Ralph Levien: disponible en: http://www.levien.com/phd/phd.html [17]

“Illustrator CS5, especificaciones técnicas”. Adobe(citada en Julio del 2011): disponible en http://www.adobe.com/es/products/illustrator/tech-specs.html [18]

Stone, Brad. “Amazon Erases Orwell Books From Kindle”. The New York Times. (17 de Julio 200[citado el 8 de agosto del 2011]) Sección: Companies, Technology: disponible en http://www.nytimes.com/2009/07/18/technology/companies/18amazon.html [19]

Lixus Devices. “SFLC, BusyBox, Monsoon dismiss GPL lawsuit”. Eweek Linux Devices(30 de octubre del 2007 [citado en Agosto del 2011]) editado por Linux Devices: dispobible en http://www.linuxfordevices.com/c/a/News/SFLC-BusyBox-Monsoon-dismiss-GPL-lawsuit/ [20]

Roosendaal, Ton. “About”. Sintel, the Durian Open Movie Proyect. (citada en Agosto del 2011). Sintel edi-

tado por Blender Open Movie Proyect: disponible en http://www.sintel.org/about/ [21]

"Russian schools move to linux" BBC News(Octubre 2007 [citada en diciembre 2011]) disponible en: http://news.bbc.co.uk/2/hi/technology/7034828.stm [22]

"German universities migrate ti Linux" Computer Weekly(Agosto 2007 [citada en diciembre 2011]) disponible en: http://www.computerweekly.com/news/2240082699/German-universities-migrate-to-Linux [23]

Focus Editors "50 places linux running you might not expect" Focus(Marzo 2010 [citada en diciembre 2011]) disponible en: http://www.focus.com/fyi/50-places-linux-running-you-might-not-expect/ [24]

Documentales y videos:

HBO. Nom de Code: Linux. Documental en línea. Dirigido por Hannu Puttonen. Francia: ADR Productions ARTE, 2001.

Stephan, Benjamin y Lutz Vogel. "Trusted Computing". MPG4. Dirigido por: Benjamin Stephan y Lutz Vogel. E.U.:LAFKON, 2005. Disponible en: http://www.lafkon.net/tc/

Entrevistas:

Caiazza, Fabricio (Rosario, Argentina)

Eschoyes, Martín (Córdoba, Argentina)

Espertino, Guillermo (San Jorge, Argentina)

García Alfaro, Javier (Rosario, Argentina)

González, Claudia (Santiago de Chile)

Martino, Inne (Rosario, Argentina)

Nieto, Ignacio (Santiago de Chile)

Pagola, Lila (Córdoba, Argentina)

Sagardoy, Andrea (San Jorge, Argentina)

Sanches, Felipe (Sao Paulo, Brasil)

Santorcuato, José (Santiago de Chile)

Serralde, José María (México D.F.)

Vega, Ricardo (Santiago de Chile)

Links

[1] http://www.gnu.org/gnu/gnu-history.es.html

[2] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[3] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[4] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[5] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[6] http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf

[7] http://www.redhat.com/about/where-is-open-

[8] http://www.redhat.com/about/where-is-open-source/activity/

[9] http://blog.joseserralde.org/opera-video-nortenos-y-software-libre

[10] http://www.mastermagazine.info/articulo/13544.php

[11] http://www.uoc.edu/portal/catala/la_universitat/sala_de_premsa/reportatges/2006/programari_lliure.html

[12] http://www.uoc.edu/portal/catala/la_universitat/sala_de_premsa/reportatges/2006/programari_lliure.html

[13] http://www.colombiaaprende.edu.co/html/mediateca/1607/articles-108475_archivo.pdf

[14] http://web.guni2005.upc.es/news/detail.php?chlang=es&id=1251

[15] http://www.icono-computadoras-pc.com/postscript.html

[16] http://alt1040.com/2010/08/mozilla-firefox-traducido-a-lenguas-prehispanicas

[17] http://www.levien.com/phd/phd.html

[18] http://www.adobe.com/es/products/illustrator/tech-specs.html

[19] http://www.nytimes.com/2009/07/18/technology/companies/18amazon.html

[20] http://www.linuxfordevices.com/c/a/News/SFLC-BusyBox-Monsoon-dismiss-GPL-lawsuit/

[21] http://www.sintel.org/about/

[22] http://news.bbc.co.uk/2/hi/technology/7034828.stm

[23] http://www.computerweekly.com/news/2240082699/German-universities-migrate-to-Linux

[24] http://www.focus.com/fyi/50-places-linux-running-you-might-not-expect/

Esta tesis se terminó de imprimir en Junio del 2012 en la Ciudad de México.

Fueron usadas las siguientes tipografías, extraídas del catálogo de Google Web Fonts, todas publicadas bajo licencia SIL Open Font License 1.1.:

Abel, por MADType
Jura, por Daniel Johnson
Marvel, por Carolina Trebol
Play, por Jonas Hecksher

No se usaron programas privativos: como Microsoft Word, Adobe Indesign, Photoshop o Illustrator. En su lugar, se usaron los siguientes programas libres:

Libre Office Writter, Scribus, Gimp e Inkscape.