Universidad de las Ciencias Informáticas

Facultad 9

Análisis de herramientas libres para el desarrollo de productos educativos multimedia

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: David A. Mestre Dumenigo

Tutor: Ing. Abel Ernesto Lorente Rodríguez

Ciudad de La Habana, Julio, 2007
"Aniversario 49 de la Revolución"

Declaración de autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

David A. Mestre Dumenigo

Ing. Abel E. Lorente Rodríguez

_____

Firma del autor

_____

Firma del tutor

Agradecimientos

Esta tesis representa una etapa muy enriquecedora de mi vida. En toda mi experiencia universitaria y la conclusión de este trabajo, ha habido personas que merecen las gracias porque sin su valiosa ayuda no hubiese sido posible llegar hasta aquí.

A mis padres, Mandy y Tere, les agradezco su apoyo, su guía y confianza en la realización de mis sueños.

A mi hermano Adrián, con mucho cariño, por los momentos que hemos compartido juntos.

A mi tutor Ing. Abel E. Lorente, por la gran ayuda profesional y humana que me ha brindado en todo momento.

Al equipo de realización del proyecto GALM por su aporte profesional a este trabajo.

A mis amigos por su confianza y lealtad.

A Dios por llenar mi vida de dichas y bendiciones.

En fin, a todo el que ha dejado una huella de amor en mi formación, y cuyos nombres sería imposible mencionar, pues sería injusto olvidar alguno de ellos.

De agradecer, no dejaré jamás.
José Martí

Dedicatoria

A mis padres Mandy y Tere, y a mi hermano Adrián, a quienes agradezco siempre por su amor, cariño y comprensión. En todo momento los llevo conmigo.

Opinión del tutor

Sobre el ***trabajo de diploma*** presentado para optar por el título de ***Ingeniero en Ciencias Informáticas.***

Título: Análisis de herramientas libres para el desarrollo de productos educativos multimedia

Autor: David A. Mestre Dumenigo

Luego de realizar una revisión del trabajo presentado y teniendo en cuenta las características del documento reglamentado para este tipo de acto, se puede expresar que el mismo tiene una buena presentación gráfica así como una buena estructuración de su contenido, logrando un buen entendimiento del contenido expresado en el mismo.

Durante su ejecución el autor cumplió con los objetivos propuestos y se destacó por su alto sentido crítico y responsabilidad. Su independencia, laboriosidad y creatividad también se puso de manifiesto. Es válido señalar que el estudiante a partir de enero y hasta mayo del presente año se desempeñó como investigador principal sin apenas asesoría y fue el soporte del equipo de trabajo demostrando su capacidad en el rol asignado.

La selección del método científico, así como su aplicación es muy acertada. La documentación resultante es muy buena en estructura y contenido.

El resultado obtenido por el estudiante puede ser considerado como muy bueno y de gran utilidad para llevar a cabo de migración de la producción de software educativo multimedia a software libre, además ha demostrado su capacidad y habilidad para resolver problemas propios de la profesión de forma muy adecuada por lo que, junto a las consideraciones expresadas anteriormente, considero que el estudiantes está apto para ejercer como Ingeniero en Ciencias Informáticas.

Considero también que los resultados de la investigación deben ser publicados y presentados en eventos científicos.

Teniendo en cuenta lo antes expresado, se propone al Tribunal la calificación de 5 puntos.

_____ _____

Ing. Abel Ernesto Lorente Rodríguez Fecha

Resumen

El siguiente trabajo trata de un análisis que se realizó a un grupo de herramientas libres para el desarrollo de productos educativos multimedia. En el mismo se hace un estudio de varias alternativas existentes y se propone un grupo de ellas por su calidad y fiabilidad para trabajar. Estas herramientas se integrarán de tal forma que cada una de ellas tendrá una función específica para desarrollar aplicaciones multimedia.

Existen varias alternativas libres, pero muchas de ellas no han logrado ocupar un lugar privilegiado dentro del movimiento del software libre. Sin embargo otras nos sirvieron de gran ayuda y aporte para la siguiente investigación.

Palabras claves

- Software educativo multimedia
- herramientas libres

Índice

INTRODUCCIÓN 1
CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA 4
1.1 INTRODUCCIÓN 5
1.2 CONCEPTOS ASOCIADOS AL DOMINIO DEL PROBLEMA 5
1.2.1 Colecciones de producto de software educativo 5
1.2.2 Código abierto 6
1.2.3 Licencia de software 7
1.3 OBJETO DE ESTUDIO 9
1.3.1 Descripción general 9
1.3.2 Herramientas libres 10
1.3.2.1 MTASC 10
1.3.2.2 UIRA 11
1.3.2.3 Ktoon 13
1.3.2.4 Librerías 13
1.3.2.5 SWFTools 15
1.3.2.6 FAMES 16
1.3.2.7 Trabajo con Bases de datos 17
1.3.3 Situación problemática 18
1.4 ANÁLISIS DE OTRAS SOLUCIONES EXISTENTES 19
1.4.1 JClic 19
1.4.2 AJAX 23
1.4.3 Python 25
1.4.3.1 PythonCard 27
1.5 CONCLUSIONES 29
CAPÍTULO 2: SOLUCIÓN PROPUESTA 30
2.1 INTRODUCCIÓN 31
2.2 POSIBLES HERRAMIENTAS PARA EL IDE 31
2.2.1 Qt Designer 31
2.2.2 Primeros pasos con MTASC 35
2.2.3 Ming. Comandos básicos de dibujos 37
2.2.3.1 Algunas funciones 40
2.2.4 UIRA 40

2.2.5 SWFC *42*
2.3 BASES DE DATOS EN ENTORNOS LIBRES 48
2.3.1 Tendencias actuales *48*
2.3.2 PostgreSQL *49*
2.3.2.1 ¿Por qué PostgreSQL? 50
2.4 CONCLUSIONES 52
CAPÍTULO 3: RESULTADOS OBTENIDOS 53
3.1 INTRODUCCIÓN 54
3.2 BENEFICIOS TANGIBLES E INTANGIBLES 54
3.2.1 Beneficios económicos *55*
3.2.2 Defensa de la Patria *55*
3.2.3 Experiencias y aportes a la Universidad de las Ciencias Informáticas *56*
3.3 VENTAJAS DE LAS HERRAMIENTAS LIBRES 56
3.4 DESVENTAJAS DE LAS HERRAMIENTAS LIBRES 57
3.5 SELECCIÓN DE LAS HERRAMIENTAS 58
3.5.1 Resultados y beneficios de las herramientas *58*
3.5.2 Herramientas colaborativas *60*
3.5.3 Importancia de estas herramientas *60*
3.5.4 Aportes del software libre *61*
3.6 RESULTADOS CON LAS BASES DE DATOS 62
3.6.1 Migrando a entornos libres *62*
3.7 CONCLUSIONES 63
CONCLUSIONES 65
RECOMENDACIONES 67
BIBLIOGRAFÍA 69
REFERENCIAS BIBLIOGRÁFICAS 73
ANEXOS 75
ANEXO #1 CONDICIONES PARA QUE UN PROGRAMA SE CONSIDERE OPEN SOURCE 75
ANEXO #2: EJEMPLO DE ASWING CON MTASC 76
ANEXO #3: ALGUNAS FUNCIONES DE MING 78
ANEXO #4 82
ANEXO #5: ENTIDADES INVOLUCRADAS EN POSTGRESQL 84
ANEXO #6: COMANDO QUE PERMITE CONOCER LAS BD DEL SERVIDOR 85

GLOSARIO DE TÉRMINOS..... 87

Introducción

En la actualidad muchas de las grandes compañías hacen importantes gastos en la adquisición de software comercial. La industria del software evoluciona con el tiempo, esto implica que se tengan que adquirir nuevas versiones para un funcionamiento óptimo del producto. Empresas líderes en el sector como Adobe, Microsoft, Macromedia, entre otras; están en constante evolución de sus productos y de esta forma obligan a los usuarios a realizar inversiones significativas si quieren mantenerse en la élite con sus aplicaciones.

A mediados de la década de los 80 Richard Stallman formalizó las ideas básicas del movimiento del software libre. El software libre, tal y como lo conocemos hoy, dio sus primeros pasos con un manifiesto a favor de la libertad de expresión y un proyecto conocido como GNU, acrónimo de *GNU's Not Unix* (GNU No es Unix) y con él la Licencia Pública General (GPL) lo que se considera la base de desarrollo del software libre. De esta forma se divisó una visión diferente a la que se tenía del software.

En el software libre existen cuatro libertades, las cuales veremos más adelante. Estas libertades son imprescindibles para su desarrollo y son los pilares del proyecto GNU. Es necesario aclarar algo, para que estas libertades se cumplan necesitan un requisito indispensable: debe existir un libre acceso al código fuente. Dichas libertades son de gran ayuda tanto a los usuarios como a los desarrolladores, beneficios que de otro modo no tendrían y que son opuestos a los del tradicional *copyright.* A menudo se confunde el concepto de software libre con el de código abierto u *Open Source* y aunque puedan parecer lo mismo entre estos conceptos existen discrepancias.

En el caso de la Universidad de las Ciencias Informáticas cuenta con un gran número de usuarios, lo que implica que el coste llega a ser un factor con un peso específico muy importante, debido a que el software acostumbra a venderse partiendo del número de licencias de uso y no como una única entidad. A esto se suma el bloqueo, impuesto por los Estados Unidos desde principios del triunfo de la Revolución, lo que impide un mayor intercambio comercial con países con tradición en esta esfera y un desarrollo estable en la industria del software por motivos de varias restricciones.

Son muchas las posibilidades de un intercambio comercial con varios países en la industria del software, específicamente en el desarrollo de software educativos multimedia, de igual forma existen las dificultades de contar con la licencia de los productos que puedan desarrollar dichas aplicaciones. Tomando estas consideraciones nos urge investigar y tomar otras vías o alternativas que pueden resolver en gran medida esta situación que se nos avecina.

Actualmente se hace muy difícil desarrollar colecciones de productos de software educativos multimedia con herramientas libre, por tal motivo es necesario desarrollar un análisis de estas herramientas para su futura implementación en un Entorno de Desarrollo Integrado (IDE).

En nuestra Universidad se han realizado varios proyectos de software educativos, pero ninguno ha tenido la particularidad de desarrollarse con herramientas libres, principalmente esto se debe a un factor fundamental: el poco o total conocimiento que se tiene de estas alternativas libres. Los principales proyectos hasta ahora realizados se han desarrollado en el antiguo Macromedia Director (ahora perteneciente a Adobe) y Toolbook, ambos software propietarios de los cuales nuestro país no posee licencia.

Conociendo estas alternativas ampliamos nuestro campo en la comercialización de la industria del software sin ningún tipo de restricciones. A medida que estas herramientas ganen potencialidad, el ahorro para la economía en nuestro país será notable, no solo por las inversiones que prácticamente son nulas, sino por lo que todo ello significará en ingresos a Cuba.

Disímiles son las alternativas que cuentan con versiones actualizadas y su documentación necesaria para una mayor explotación de las mismas. Muchas de ellas se rigen por normas y estándares internacionales lo que trae consigo un trabajo sumamente profesional.

Este trabajo tiene como objetivo principal realizar un análisis de las aplicaciones libres que permitan la futura implementación de un sistema para la automatización del proceso de producción de productos educativos multimedia y de esta forma

documentar el mayor número posible de herramientas para la implementación de aplicaciones con estas características.

Aunque en cuestiones tecnológicas es difícil hacer predicciones, el acercamiento a la tecnología por parte de los usuarios es evidente no sólo en el campo de la informática. Una tarea primordial de este trabajo es verificar si existe documentación o se ha realizado algún trabajo investigativo sobre las herramientas existentes, así como analizar el estado del arte en el desarrollo de estas herramientas para la producción de software educativo. Hay que aprovechar este hecho para crear comunidades virtuales interesadas en la creación y la difusión del *Open Source*, puesto que es un tema actual.

Capítulo 1: Fundamentación teórica

Capítulo 1
Fundamentación teórica

1.1 Introducción

Muchas son las empresas que se dedican al desarrollo y comercialización de software. En la actualidad estas compañías distribuyen sus productos bajo licencia copyright, o sea, que se protege contra uso, copia o redistribución del mismo. Debido a lo anteriormente expuesto en el año 1984 Richard Stallman comenzó a trabajar en el proyecto GNU (GNU es un acrónimo recursivo que significa "GNU No es Unix") y luego un año más tarde fundó la *Free Software Foundation* (FSF) a la cual le introdujo el concepto de *copyleft.*

En el caso de software libre un monopolio de producto no es automáticamente un monopolio de empresa. En este caso si el producto es libre, cualquier empresa o incluso usuario puede trabajar con él, modificarlo y agregarle algunas mejoras de acuerdo con sus necesidades de tal forma que ayude a su evolución.

Existen diversas herramientas para el desarrollo de estos productos pero debido al poco conocimiento de estas alternativas libres nos vemos imposibilitados para la creación de un IDE y de esta manera perder paulatinamente espacio en el mercado mundial.

1.2 Conceptos asociados al dominio del problema

1.2.1 Colecciones de producto de software educativo

Un software educativo es un programa de computadora desarrollado para la educación, creado con la finalidad de ser usado como medio didáctico, de esta forma facilita los procesos de enseñanza - aprendizaje en sus modalidades tradicional y presencial a distancia.

Cuando nos referimos a colecciones de productos de software educativo multimedia hacemos referencia de forma general a programas informáticos que combinan textos, imágenes y sonidos, es decir, a una variedad de recursos que nos brinda la posibilidad

de obtener información de diversos medios de comunicación interconectados y que pueden ser controlados por el alumno mediante el ordenador. Este entorno permite al usuario (en este caso el alumno) interactuar activamente con la información.

Así como existen profundas diferencias entre las filosofías pedagógicas, así también existe una amplia gama de enfoques para la creación de software educativo atendiendo a los diferentes tipos de interacción que debería existir entre los actores del proceso de enseñanza aprendizaje: educador, aprendiz, conocimiento y computadora.(WIKIMEDIA FOUNDATION 2007f)

1.2.2 Código abierto

Durante el año 1998, Eric S. Raymond, Bruce Pernees y otros *hackers* involucrados en el desarrollo de software libre lanzaron la Open Software Initiative y propusieron el uso de término open source (código abierto) en contraposición al término free software (software libre) como término más atractivo al entorno empresarial. El término free software en el mundo anglófono creaba una situación incómoda debido a la doble acepción que en inglés tiene el término free (que puede significar gratuito o libre). La gran mayoría de empresas en Estados Unidos usan principalmente el término código abierto para evitar dar la percepción que el software libre es un recurso totalmente gratuito y para poner énfasis en el valor diferencial que representa el hecho de que el código fuente está disponible. Bruce Perens, de la Open Source Iniciative y antiguo coordinador de la distribución de Linux Debian, creó una lista de condiciones que debe cumplir un programa para poder ser considerado Open Source. Estas condiciones son muy similares y, de hecho están basadas, en las directrices de software libre de Debian. Estas condiciones también son aplicables a cualquier programa que sea software libre y pueden ayudarnos a matizar sus implicaciones:(CULEBRO JUÁREZ 2006)
[Ver Anexo # 1]

Sin embargo, hay que precisar, este proceso puede ser gratuito o bien representar algunos costes. Software libre no es igual a software gratis.

De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- La libertad de usar el programa, con cualquier propósito (libertad 0).
- La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades (libertad 1). El acceso al código fuente es una condición previa para esto
- La libertad de distribuir copias, con lo que puedes ayudar a tu vecino (libertad 2).
- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. (libertad 3). El acceso al código fuente es un requisito previo para esto.

Ninguna de estas libertades es una obligación. Es decir, nunca el usuario está obligado a aplicar las cuatro cosas, pero tiene la libertad de hacer cualquiera cuando quiera.

La filosofía del *Open Source* centra su atención en la premisa de que al compartir el código, el programa resultante tiende a ser de calidad superior al software propietario, es una visión meramente técnica. Por otro lado, el software libre funciona bajo un ideal: el software propietario, al no poder compartirse, es poco ético dado que prohibir compartir entre seres humanos va en contra las leyes naturales.(WIKIMEDIA FOUNDATION 2007c)

1.2.3 Licencia de software

Una licencia de software (en inglés *software license*) es el permiso que se le concede al titular del derecho del autor. Existe la Licencia Pública General de GNU que su objetivo principal es garantizar la libertad de compartir y modificar software libre, para asegurar que el software es libre para todos sus usuarios.

Más formalmente, una licencia es considerada *Open Source* cuando ha sido aprobada por la *Open Source Initiative* (OSI), donde el criterio lo da la definición de *Open Source*. El software de dominio público (esto significa sin licencia), cumple todos estos criterios siempre y cuando todo el código fuente esté disponible, y esté reconocido por la OSI y se le permita usar la marca de la misma.(WIKIMEDIA FOUNDATION 2007d)

La licencia, que puede ser gratuita u onerosa, precisa los derechos (de uso, modificación o redistribución) concedidos a la persona autorizada y sus límites. Una licencia, puede señalar el plazo de duración, el territorio de aplicación entre otros aspectos que establezca su autor.

De forma general aunque hay muchas licencias se distinguen tres modelos principales de licenciamiento de software.

Licencias con *copyleft*. Son aquellas que ceden los derechos de copia, distribución y modificación del programa bajo las condiciones que definen al software libre pero que además exigen que cualquier versión modificada herede el mismo tipo de obligaciones y derechos que tenía el programa original. Estas licencias a menudo se llaman víricas por el efecto de contagio que tienen sobre trabajos derivados. El objetivo es garantizar que cualquier usuario conserve en el futuro las libertades originales que definen al software libre, y este futuro incluye los trabajos derivados del software original.(HERNÀNDEZ 2005)

Uno de los ejemplos de la licencia anterior es el del sistema operativo Linux, de esta forma se asegura que cualquier modificación al sistema se obliga a hacer público su código fuente.

Licencias de código abierto o permisivas**.** Aquellas que ceden el uso del programa bajo las condiciones que definen el software libre pero no obligan necesariamente a hacer públicas las mejoras que realicemos sobre el código. Con las licencias permisivas se pueden usar programas informático libre, hacer ampliaciones y crear un producto propietario sin compartir con el resto de la comunidad las mejoras introducidas.(HERNÀNDEZ 2005)

Así por ejemplo, Netscape creó su producto comercial propietario *Netscape Directory Server* basándose en el código fuente del servidor del software libre de la Universidad de Michigan sin necesidad de tener que publicar las mejoras, con lo cual sólo los clientes de Netscape podían beneficiarse de estas mejoras a pesar que la empresa había usado un programa libre.

Este tipo de licencia da más libertad a los programadores debido a que no impone ninguna restricción, permitiendo incluso desarrollar software propietario de un proyecto libre.

La licencia BSD es una de las más comunes de este tipo. Lo más destacable es que permite el uso y explotación de la aplicación, en forma de programa o código, sin ningún tipo de limitación. También protege al autor de los usos que terceros puedan hacer de su nombre para publicitar el producto y obliga a menudo a dar crédito a los autores originales.(HERNÀNDEZ 2005)

Por ejemplo, el objetivo de FreeBSD es proporcionar un sistema operativo sólido como una roca y tremendamente eficiente, que obtenga el máximo rendimiento de la máquina. En cambio, el proyecto NetBSD busca la portabilidad: no se limita a funcionar sobre los procesadores Intel o los PPC de los Apple, sino que es capaz de comportarse exactamente igual sobre una vertiginosa lista de máquinas menos frecuentes. Por último, OpenBSD presume, y con razón, de ser el sistema operativo más seguro del mundo. Dos miembros más recientes de la familia son Mac OS X y DragonFlyBSD.(SÁNCHEZ 2004)

1.3 Objeto de estudio

1.3.1 Descripción general

En la actualidad el *open source* nos facilita un grupo de herramientas con grandes posibilidades para desarrollar colecciones de productos educativo multimedia. Muchas de estas alternativas cuentan con el apoyo económico necesario para su desarrollo, ejemplo de ello es Ktoon, programa de animación en 2D creado por Toonka Films y muy difundido a nivel mundial.

Sin embargo otros proyectos no han corrido la misma suerte y solo han quedado en una primera fase, esto se debe a dos factores esenciales: falta de apoyo económico y escasez de personal calificado.

En visita realizada este año a Cuba el promotor y fundador del *Open Source*, Richard Stallman aseguró que le gustaría que no solo Cuba, sino todo el mundo migraran por

completo a software libre. «Para eso el mayor obstáculo es la inercia social. Pero Cuba tiene experiencia en luchar contra fuertes obstáculos. Entonces puede hacerlo».

1.3.2 Herramientas libres

Varias son las herramientas libres que existen y su funcionalidad cada día es mayor. Su aporte a la economía es significativo, precisamente por ser alternativas libres para la creación de proyectos de gran envergadura.

Términos relacionados

Existen muchos términos que en ocasiones se tienden a confundir por su similitud pero no significan lo mismo, a continuación veremos algunos de estos conceptos.

Freeware

En Internet se pueden conseguir varios programas, uno de ellos son los llamados *Freeware* que son programas gratuitos. En muchas ocasiones incluyen su código fuente aunque no es lo más usual. Estos programas suelen incluir licencia de uso permitiendo su redistribución pero con algunas restricciones.

La mayoría de estos programas se pueden obtener de un sitio oficial en Internet, normalmente para proporcionar otros programas o servicios. Contrariamente a lo que se cree, los programas de software libre no necesariamente son *Freeware.*

Shareware

Es más bien un método de distribución, debido a que los programas, generalmente sin fuentes, se pueden copiar libremente, pero no usar continuadamente sin pagarlos. Este sistema destaca entres sus ventajas principales que el usuario puede probar el software antes de adquirirlo, de esta forma se evita el riesgo de comprar algo que luego no sea útil.

1.3.2.1 MTASC

MTASC (Motion Twin Action Script Compiler) es un compilador *open source* para ActionScript 2.0 capaz de compilar 100 clases en menos de 5 segundos y es más estricto en muchos aspectos en los que el compilador de Flash no le da importancia.

Su creador, Nicolás Canesse, se ha ocupado de sacar cuatro versiones betas en a penas dos semanas, él ha creado el compilador teniendo en cuenta el *Standard ECMA*.

Todo comenzó con un compilador muy restrictivo, que no aceptaba muchas peculiaridades del compilador de Flash, con lo que la gente exigía más flexibilidad para compilar sus proyectos.

Poco a poco se ha avanzado en el proyecto y todo parece indicar que se puede dejar de depender del IDE de Flash para desarrollar sus aplicaciones con ActionScript. MTASC es independiente de plataforma, se puede compilar para la plataforma que se desee, lo único que se necesita es poder compilar el lenguaje y compilador de Ocalm.

El compilador funciona desde la línea de comandos y por lo tanto obliga a trabajar con Programación Orientada a Objetos (POO), o sea los proyectos tienen que ser un conjunto de clases. Existen dos variantes, se puede escoger trabajar con POO totalmente y exclusivamente con MTASC y la otra es parcialmente con el IDE de Flash y compilar las clases con MTASC.

Si se escoge trabajar totalmente con POO, desarrollar una aplicación desde Linux no se logra ver como hacerlo sin depender del IDE de Flash, incluso aunque solo fuese para instanciar la primera clase, sin embargo se puede trabajar con un sistema de clases en la que una de ellas se auto ejecute al empezar la aplicación.

RECORDAR

- SWfmill genera swf con librería partiendo de XML y elementos externos
- MTASC inyecta el código necesario

1.3.2.2 UIRA

Las alternativas a Adobe Flash en Linux son muy escasas y debido a este déficit de variantes se decidió crear un proyecto propio para ampliar estas tecnologías.

Partiendo del código de Flash4linux (F4l) se fueron rescribiendo varios módulos para solucionar diversos *bugs* y añadirles más funcionalidades. F4l siempre ha tenido problemas a la hora de crear Flash, se pueden hacer animaciones pero todavía tiene grandes dificultades para generar un archivo swf.

Pasaron unos seis meses y partiendo de ese código se obtuvo QFlash, el mismo puede crear animaciones básicas, añadir textos, polígonos, formas básicas, cuadro de texto, etc.

Entre agosto y septiembre del año 2005 Florian Delizy se puso en contacto con ambos proyectos: F4l y Qflash. Se llegó a un acuerdo general y decidieron crear un nuevo proyecto, y de esta forma surgió UIRA. Se decidió cambiar el nombre por tres razones fundamentales:

1. Los dos proyectos anteriores contienen en su nombre *flash* y los propietarios no querían problemas de *copyright.*
2. Flash4linux no contaba con un gran alcance, a sus desarrolladores les interesaba otras plataformas como Mac y Windows.
3. Flash4linux se quedaba corto porque no solo se quería crear flash.

Se decidió el nombre de UIRA por la mitología Maori y que es el Dios del trueno y del relámpago.

Actualmente el proyecto está en pleno desarrollo, por ahora solo se está trabajando en buscar un diseño óptimo que evite que se tenga que deshacer el trabajo por mal diseño. Aún no puede crear flash ya que se está programando librerías básicas, sistemas de *plugins* y preparando diagramas UML.

La parte gráfica se desarrolla con las librerías Qt4 y también usa el motor gráfico Amanith. Para el editor ActionScript se usa las librerías *scintilla*, actualmente este proyecto trabajo en conjunto con Ktoon para compartir parte del código.

1.3.2.3 Ktoon

Ktoon es una herramienta para desarrollar animaciones 2D realizada por animadores de Tonnka Films. Este proyecto está cubierto por licencia GPL usando G++, OpenGL y Qt como recursos de programación desde Kdevelop, es válido aclarar que por ahora solo está disponible para sistemas Unix.

Toonka Films tomó la iniciativa de desarrollar una herramienta que cumpliera con sus expectativas como creadores de contenidos animados. Toonka Films es una empresa colombiana de animaciones que pertenece al Parque Tecnológico del Software, ellos conocieron la cultura del *open source* y vieron una gran oportunidad para cambiar la historia de su empresa.

En Colombia existe una entidad del gobierno que se llama Conciencias que esta relacionada con la investigación, cada año está institución lanza convocatorias para el desarrollo de proyectos de investigaciones. Los ejecutivos de Toonka Films vieron una gran oportunidad de impulsar su proyecto ya que Conciencias financiaría gran parte del proyecto y se da el primer paso para la creación de Ktoon.

Ktoon se divide en dos módulos, el primero es de ilustración y un segundo módulo de animación. El de ilustración permite hacer los componentes gráficos, está muy vinculado a la edición de los componentes. El módulo de animación permite hacer edición pero no de las gráficas, sino del proyecto en general con respecto al tiempo.

1.3.2.4 Librerías

Las librerías no son más que un conjunto de procedimientos que se agrupan en un archivo con el propósito de ser aprovechadas por otros programas. En ocasiones se emplea el término biblioteca para referirse a una librería, ambos vocablos son correctos. Unas de las librerías más usadas en el software libre son Qt y Ming, a continuación veremos ambas librerías.

Qt

Qt es un producto creado por Trolltech AS, esta empresa, que tiene su sede principal en Noruega, se dedica a desarrollar librerías y herramientas para el desarrollo de

software. Qt es una librería para el desarrollo de interfaces gráficas, es multiplataformas: Linux, MacOs X, Solaris, HP-UX, MS-Windows XX, UNIX con X11.

Además, existe también una versión para sistemas empotrados. Podemos incorporar las Qt en nuestras aplicaciones open source debido a que se distribuye bajo una licencia libre GPL. La comercialización de Qt comenzó en el año 1996 y desde su inicio es usada en diversas aplicaciones en las cuales se incluye la interfaz gráfica para Linux KDE. Esta librería está escrita en código C++ y es orientada a objeto.

Características

Es una librería que se basa en los conceptos de *widgets* (objetos), Señales-Slots y Eventos, las señales y los *slots* es el mecanismo para que unos *widgets* se comuniquen con otros. Los *widgets* pueden contener cualquier número de hijos. El *widget "top-level"* puede ser cualquiera, sea ventana, botón entre otros. Algunos atributos como el texto de etiquetas se modifican de modo similar al lenguaje html.

Otras funcionalidades de Qt:

- Librerías básicas -> Entrada/Salida, Manejo de Red, XML
- Interfaces con bases de datos -> Oracle, MySQL PostgreSQL, ODBC
- Plugins, librerías dinámicas (Imágenes, formatos, etc.)
- Unicode, Internacionalización

En el mercado se pueden encontrar las siguientes distribuciones de Qt:

- **Qt Enterprise Edition** y **Qt Profesional Edition,** disponibles para el desarrollo de software con fines comerciales. Incluye servicio de soporte técnico y están disponibles ampliaciones.
- **Qt Free Edition** Es la versión para Unix/X11 para el desarrollo de software gratuito y de código abierto. Se puede obtener gratis sujeto a los términos de la *Q Public License and the GNU General Public License*. Para plataformas Windows también está disponible la versión **Qt non comercial**.
- **Qt Educational Edition** es una versión de la Qt Profesional Edition con licencia únicamente para fines educacionales.

- **Qt/Embedded Free Edition**.

Ming

Ming es una librería de C de código abierto (LGPL) que permite escribir películas SWF. Permite generar películas Flash sin ser necesaria la utilización de Adobe Flash además que es capaz de generar ficheros SWF mediante el desarrollo de código C++, PHP y otros lenguajes. Esta librería soporta las la mayoría de las características de Flash entre las que se destacan el soporte de audio mp3, formas, botones, gradientes, textos, acciones entre otras.

Aunque la librería Ming está escrita en código C plano, está diseñada para ser utilizada en un ambiente orientado a objetos; por consiguiente, en PHP se puede acceder a las funciones de dibujo de Ming mediante el objeto *SWFShape*.(WIKIMEDIA FOUNDATION 2007b)

Ming presenta varias ventajas. Tiene la posibilidad de trabajar con distintos tipos de objetos, permite usar C++ y generar fácilmente las películas flash sin usar Adobe Flash y tiene la facilidad de uso de los distintos métodos que ofrece.

1.3.2.5 SWFTools

SWFTools son un grupo de herramientas de código abierto para crear y manipular ficheros SWF. SWF es el formato utilizado por el software de animación Adobe Flash (anteriormente Macromedia Flash). SWFTools ha sido liberado bajo licencia GPL, y funciona en entornos Windows, Mac OS X, Linux y otros sistemas tipo Unix.

La herramienta principal es SWFCombine (SWFC), que recoge la descripción de la animación Flash en un lenguaje sencillo y genera el fichero de salida SWF. Es posible incluir scripts ActionScript en el fichero generado. SWFTools también incluye la biblioteca RFXSWF, permitiendo a programas de terceros generar ficheros SWF.(WIKIMEDIA FOUNDATION 2007g)

SWFTools es una colección de programas para la manipulación de ficheros Flash .SWF-files que a parte de SWFC incluye:

- **PDF2SWF:** genera un marco por página. Le posibilita completamente haber formateado texto, incluyendo tablas, fórmulas etcétera, dentro de su Flash Movie
- **SWFStrings Scan:** para datos del texto
- **JPEG2SWF:** describe y genera a un SWF *slideshow*
- **SWFExtract:** extrae *movieclips*, sonidos e imágenes de archivos SWF
- **RFXSWF Library:** biblioteca que puede servir para la generación de un SWF. Incluye soportes para *bitmaps*, botones, textos, fuentes, sonidos entre otros. También tiene soporte para ActionScript usando *Ming ActionCompiler*.

De todas estas herramientas mencionadas las que utilizaremos en nuestro proyecto son el SWFC y el SWFExtract.

1.3.2.6 FAMES

FAMES (*Flashout – ASDT – MTASC – Eclipse – Swfmill*) es una alternativa a Macromedia Flash libre y *open source,* se basa en Eclipse junto con unos *plugins* y otros programas.

Eclipse

Es un IDE para varios lenguajes, sobre todo para Java pero a través de *plugins* se puede lograr que compile de todo. Puede compilar mientras se escribe y tiene una robustez excelente.

Flashout y ASDT

Provee en los *plugins* para Eclipse

MTASC

Es el compilador, una rapidez y eficiencia de trabajo óptima

Swfmill

Es un *XML2SWF2XML,* explicándolo de otra forma es un traductor de XML a un Swf y los Swf en XML, la sintaxis es fácil, sirve para agregar objetos al Swf que posteriormente se pondrán en uso. Es el uso muy común la generación de bibliotecas del recurso que contienen las imágenes (PNG y JPEG), fuentes (TTF) u otras películas

de SWF para el uso con ActionScript MTASC-compilado, aunque pueden usarse los Swfmill para producir las estructuras de SWF simples y complejas. El Swfmill está en pleno desarrollo y varias características se le incorporan diariamente.

FAMES carece de un sistema gráfico por lo que no se recomienda para realizar animaciones que requieran mucho *timeline.* Pero sin embargo la rapidez a la hora de compilar hace que sea una herramienta preferida por muchos usuarios, además de lo anteriormente expuesto FAMES obliga a usar AS 2.0, así como olvidarnos un poco de la parte gráfica porque con MTASC se puede compilar sobre un Swf sin tocar la parte gráfica. De esta forma un diseñador y un programador podrían trabajar juntos sin la necesidad de intercambiar criterios, por un lado contaríamos con la parte gráfica en un Swf y por el otro la programación, al final el programador seria el encargado de recibir un Swf, compila y todo estaría listo.

1.3.2.7 Trabajo con Bases de datos

A principios de los años sesenta las aplicaciones informáticas acostumbraban a darse totalmente por lotes y estaban pensadas para una tarea muy específica relacionada con muy pocas entidades tipo. Cada aplicación utilizaba ficheros de movimientos para actualizar y/o consultar uno o dos ficheros maestros que en aquel entonces estaba sobre cinta magnética.

Luego apareció el término *on-line* y se fueron introduciendo las líneas de comunicación, los terminales y los discos, se fueron desarrollando aplicaciones que permitían a varios usuarios consultar los mismos ficheros. El acceso *on-line* y la utilización eficiente de las interrelaciones exigían estructuras físicas que dieran un acceso rápido, como por ejemplo los índices. Todos estos conjuntos de ficheros conectados, con estructuras complejas y compartidos por varios procesos de forma simultánea, recibieron al principio el nombre de *Data Banks*, y luego, a inicios de los años setenta, el de *Data Bases*, en español Bases de Datos.

PostgreSQL

PostgreSQL es un gestor de bases de datos orientado a objeto usado en entornos de software libre debido a que cumple los estándares SQL92 y SQL99, y también por el

conjunto de funcionalidades avanzadas que soporta, lo que lo sitúa al mismo o a un mejor nivel que muchos SGBD comerciales. PostgreSQL se distribuye bajo licencia BSD, este tipo de licencia lo vimos al principio de este capítulo lo que permite su uso, redistribución, modificación con la única restricción de mantener el *copyright* del software a sus autores.

Características

PostgreSQL tiene una serie de características que lo hacen capaz de competir con cualquier SGBD comercial:

- Está desarrollado en C, con herramientas Yacc y Lex.
- La API de acceso al SGBD se encuentra disponible en C, C++, Java, Perl, PHP, Python y TCL entre otros.
- Su administración se basa en usuarios y privilegios.
- Es altamente confiable en cuanto a su estabilidad.
- En PostgreSQL es posible definir un nuevo tipo de tabla a partir de otra previamente definida.

Limitaciones

Este tipo de SGBD tiene algunas limitaciones, entre las que se destaca principalmente:

- No soporta *tablespace* para definir donde almacenar los datos.
- El soporte a orientación a objetos es una pequeña extensión que ofrece prestaciones como la herencia, no un soporte completo.

1.3.3 Situación problemática

Son muchas las empresas y los países que han comenzado la migración de todos sus sistemas basados en software privativos a tecnología basada en Software Libre *(Open Source)*. El auge adquirido por estas tecnologías nos obliga a tomar las medidas necesarias e ir previendo el futuro mediato que se nos avecina. A pesar de lo antes mencionado, las herramientas para el desarrollo autorías multimedia educativas, aun no se desarrollan, hay varios factores que influyen en ello, las grandes empresas dígase: Macromedia, SumTotal, RunRev, no desarrollan en su gran mayoría productos

para plataformas libres y aunque algunos desarrollen aplicaciones para múltiples plataformas incluyendo Linux, no distribuyen sus productos bajo licencia GPL *(General Public License)*. En la actualidad constituye un reto desarrollar un software educativo o una multimedia que se ejecute en plataformas libres y aun más desarrollarlo con herramientas y tecnologías libres.

A todo lo dicho con anterioridad y aprovechando el marco de los convenios Cuba-Venezuela se han firmado varios contratos con el gobierno venezolano, en los cuales nuestra universidad tiene que desarrollar una serie de productos educativos y multimedia para plataformas libres y con tecnologías libres.

La situación actual no permite cumplir fielmente los requerimientos de la parte venezolana, en los cuales quedan definidos que estos productos se deben implementar con herramientas o aplicaciones libres. Las razones por lo cual no se ha hecho de esa forma son las siguientes:

- Las herramientas que nos brindan esas posibilidades no existen o no sabemos que existen.
- Las herramientas que conocemos no nos permiten cumplir con los requerimientos de los productos a desarrollar.
- No tenemos conocimiento ninguno de cómo trabajar con algunas de estas herramientas.

1.4 Análisis de otras soluciones existentes

1.4.1 JClic

JClic surge partiendo de la versión de Clic 3.0, una aplicación que desde 1992 ha sido utilizada como herramienta de creación de actividades didácticas para los estudiantes. JClic está desarrollada en la plataforma Java, es un proyecto de código abierto y funciona en diversos entornos y sistemas operativos. El formato para almacenar los datos de las actividades es XML.

JClic está formado por un conjunto de aplicaciones educativas, como por ejemplo: rompecabezas, asociaciones, ejercicios de texto entre otros. Las actividades no se

presentan de manera independiente sino en proyectos el cual está conformado por un conjunto de actividades y una o más secuencias.

JClic está formado por cuatro aplicaciones:

1. *JClic applet*
   Un applet que permite incrustar las actividades JClic en una página Web.

2. *JClic player*
   Es un programa totalmente independiente, una vez que se instale permite realizar las actividades localmente (o desde la red) sin que sea necesario estar conectado a Internet.

3. *JClic author*
   La herramienta de *author* que permite crear, editar y publicar las actividades de una manera más sencilla, visual e intuitiva.

4. *JClic reperts*
   Un módulo de recogida de datos y generación de informes sobre los resultados de las actividades hechas por los alumnos.

Para utilizar JClic hay que tener instalada la versión 1.3.1 o superior de la máquina virtual de Java (JRE).

Novedades del JClic

- Uso de entornos gráficos de usuarios personalizados
- Uso de gráficos tipo BMP, GIF, JPG y PNG
- Cuenta con nuevos recursos multimedia como WAV, MP3, AVI, MPEG, QuickTime, Flash 2.0 entre otros
- Sonidos de evento que se pueden configurar para cada actividad
- Considerables mejoras visuales ya que permite escribir código HTML en la s casillas, incrustación de fuentes *TrueType*, textos con estilos, etc.

Tipos de actividades

JClic permite realizar siete actividades básicas:

1. **Las asociaciones:** el usuario descubre las relaciones existentes entre dos conjuntos de información.
2. **Los juegos de memoria:** consiste en descubrir parejas de elementos iguales o relacionados entre ellos que se encuentran escondidos.
3. **Actividades de exploración, identificación e información:** parte de un único conjunto de información.
4. **Los puzzles:** plantean la reconstrucción de una información que se presenta al principio de manera desordenada. Esta información puede ser de diferentes tipos, como gráfica, de texto, sonora o combinar estos aspectos.
5. **Respuesta escrita:** se resuelven escribiendo un texto.
6. **Actividades de texto:** son ejercicios basados siempre en las palabras, frases, letras y párrafos de un texto que hay que completar, interpretar, corregir u ordenar.
7. **Sopas de letras y crucigramas:** son variantes instructivas de los pasatiempos de palabras escondidas.

Applet JClic

Un *applet* es una aplicación interactiva que se encuentra dentro de una página Web que puede mostrarse en los navegadores que soportan Java. El *applet* hace un trabajo muy similar al de JClic, carga los datos del proyecto, luego muestra las secuencias de actividades y por último comprueba el trabajo realizado.

La principal diferencia entre el JClic y el *applet* radica en que el primero las aplicaciones se descargan y se guardan en el disco duro mientras que el *applet* está diseñado para trabajar desde Internet sin necesidad de guardar ninguna información localmente.

Para insertar un *applet* en una página Web se puede hacer de dos formas: una es que ocupe el espacio completo y la otra que ocupe una dimensión especificada anteriormente.

A la hora de trabajar con el *applet* que nos ocupe todo el espacio nos permite indicar el color de fondo, la alineación y los bloques de texto opcionales que se mostrarán encima y debajo del applet.

Esta página Web solo estará visible desde el ordenador, si el objetivo es que otros usuarios tengan acceso hay que publicarla en Internet.
Para publicar un sitio Web hacen faltas dos cosas esenciales:

- Un espacio en un servidor Web
- Un programa de transferencia de archivos

Es necesario aclarar que para publicar el sitio hay que transferir el espacio Web los archivos .jclip.zip y el documento HTML (index.htm). En el servidor se debe crear una carpeta para cada proyecto de forma independiente con el objetivo de no mezclar los archivos.

La mediateca

La mediateca es donde se almacenará todos los recursos multimedia de un proyecto, donde se gestionan las imágenes y otros recursos que pueden ser utilizados por las actividades. Desde la mediateca se pueden añadir, borrar o visualizar los recursos multimedia del proyecto. Cuando se adiciona un recurso se puede optar entre ver todos o especificar algún tipo de formato.

Importar actividades de Clic 3.0 a JClic

JClic, además de permitir crear proyectos propios con la extensión **.jclic.zip**, ofrece la posibilidad de importar los paquetes de actividades de Clic 3.0, que pueden tener extensiones .PAC o .PCC. Esta posibilidad es útil en caso de que se tengan actividades creadas con Clic 3.0 y se quieran adaptar a JClic.

Una vez importados, los paquetes de actividades de Clic 3.0 se transforman automáticamente en proyectos JClic. Lo único que hay que hacer es guardar el proyecto abierto con JClic autor como archivo **jclic.zip**, que, por otra parte, es la única opción que nos permite el programa. Hay que indicar el lugar donde se quiere guardar

que, por defecto, es la misma carpeta donde se encuentra el archivo PAC o PCC. Un buen sitio para situar estos proyectos es la carpeta proyectos de JClic: **C:\Archivos de programa\JClic\projects**
Después de hacer todos estos pasos, el antiguo paquete quedará convertido en un proyecto y se podrá visualizar desde JClic.

1.4.2 AJAX

Ajax (*Asynchronous JavaScript And XML*) no es una tecnología, sino la unión de varias tecnologías que juntas tiene un resultado impresionante. Se puede decir que Ajax es una técnica de desarrollo Web que es utilizada para crear aplicaciones interactivas. Ajax está conformado por las siguientes tecnologías:

1. XHTML o HTML y CSS para el diseño que acompaña a la información
2. *Document Object Model* (DOM) accedido con un lenguaje de *scripting* por parte del usuario
3. El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor Web

XML es el formato usado comúnmente para la transferencia de de vuelta al servidor, aunque cualquier formato puede funcionar incluyendo HTML. A pesar de que el término "Ajax" fuese creado en 2005, la historia de las tecnologías que permiten Ajax se remonta a una década antes con la iniciativa de Microsoft en el desarrollo de Scripting Remoto. Sin embargo, las técnicas para la carga asíncrona de contenidos en una página existente sin requerir recarga completa remontan al tiempo del elemento *iframe* (introducido en Internet Explorer 3 en 1996) y el tipo de elemento *layer* (introducido en Netscape 4 en 1997, abandonado durante las primeras etapas de desarrollo de Mozilla). Ambos tipos de elemento tenían el atributo SRC que podía tomar cualquier dirección URL externa, y cargando una página que contenga JavaScript que manipule la página paterna, pueden lograrse efectos parecidos al Ajax.(WIKIMEDIA FOUNDATION 2007a)

¿Qué se puede hacer con Ajax?

Ajax nos sirve principalmente para diseñar y programar interfaces de usuario mucho más allá de la Web, rompiendo las limitaciones que la "sincronía" supone y abriendo una nueva puerta que nos permitirá desarrollar aplicaciones Web que en un principio solo podrían concebirse para el escritorio.

La comunicación asíncrona

La comunicación asíncrona para bien o para mal supone un cambio radical a la hora de diseñar aplicaciones Web. Con la comunicación asíncrona nos libramos de unas barreras insalvables dentro del desarrollo clásico. A continuación veremos un ejemplo comparando la comunicación asíncrona con la síncrona para editar un registro.

- **síncrona:** A la hora de actualizar un registro dentro de una aplicación "clásica" encaminamos al usuario a una página (1) donde le indicamos que ha de seleccionar un registro para editar. Una vez transmitida esa información a una página (2) le mostramos la información que disponemos actualmente para ese registro. Tras ser editada esa información le enviamos a otra página (3) donde le indicaremos si la actualización ha sido correcta o si por el contrario se ha dejado algún campo por cumplimentar, pudiendo hacerle repetir los pasos 2 y 3 infinitamente hasta que inserte bien los datos. Hemos utilizado hasta 3 o 4 páginas "vistas" y un mínimo de 3 tiempos de carga.

- **asíncrona:** A la hora de actualizar un registro dentro de una aplicación asíncrona desde la salida de cualquier búsqueda podemos indicarle que pulsando en el botón de herramientas le aparecerán las herramientas de editar y eliminar... pulsará en editar y se le desplegara debajo de su registro un formulario con toda la información de la que disponemos. Mientras escribe los nuevos datos podemos comprobar si son correctos y cuando haya terminado y todo esté bien le dejamos pulsar en el botón guardar cambios. En ese momento el formulario desaparece y los cambios quedan guardados en la página. Hemos usado 1 página "vista". Y un solo tiempo de carga.

Ventajas

Interactividad

Las aplicaciones AJAX se ejecutan del lado del cliente manipulando la página actual dentro de sus navegadores usando métodos de *Document Object Model*. Puede ser usado para multitud de tareas como actualizar o eliminar registro, expandir formularios Web, devolver peticiones simples o búsquedas; todo sin tener la necesidad de tener que recargar toda la página HTML cada vez que se realiza un cambio.

Portabilidad

Las aplicaciones de AJAX utilizan características bien documentadas presente en todos los navegadores importantes en la mayoría de las plataformas existentes.

Dinamismo

Nos perite desarrollar aplicaciones mucho más dinámicas disminuyendo el peso de la página debido a la comunicación asíncrona con el servidor.

Desventajas

Una de las mayores críticas contra AJAX en aplicaciones Web es que pueda fácilmente acabar con el comportamiento normal del botón atrás del navegador, el cliente necesita un navegador que soporte Javascript, aunque hoy en día los navegadores Internet Explore. Mozilla, Firefox, Safari, entre otros soportan Javascript.

1.4.3 Python

Python es un lenguaje de programación creado por Guido van Rossum en el año 1990. Es comparado habitualmente con TCL, Perl, Écheme, Java y Ruby. En la actualidad Python se desarrolla como un proyecto de código abierto, administrado por la *Python Software Foundation*, es considerado como la "oposición leal" a Perl, lenguaje con el cual mantiene una rivalidad amistosa.(WIKIMEDIA FOUNDATION 2007e)

Python es un lenguaje potente y no es muy complicado. Cuenta con eficaces estructuras de datos de alto nivel y una solución de programación orientada a objeto simple pero eficaz. La elegante sintaxis de Python, su gestión de tipos dinámica y su naturaleza interpretada hacen de él el lenguaje ideal para guiones (*scripts*) y desarrollo rápido de aplicaciones en muchas áreas y en la mayoría de las plataformas.

El intérprete de Python y su extensa biblioteca estándar están disponibles libremente, en forma de fuentes o ejecutables, para las plataformas más importantes y se pueden distribuir libremente.

Características

Una característica peculiar de este lenguaje, y que llama la atención poderosamente, para bien o para mal, la primera vez que alguien empieza a utilizarlo, es que la indentación es obligatoria. Esto quiere decir que el sistema para agrupar distintas partes del código es mediante el sangrado de las mismas, sin que se utilicen llaves, paréntesis ni ninguna expresión. En realidad esta práctica la utilizan en general el resto de lenguajes para obtener una mayor legibilidad del código, Python simplemente la hace obligatoria y así elimina los elementos que hacen de delimitadores de bloques en otros lenguajes. Se trata de una característica que a algunos programadores les resulta irritante al empezar, pero termina produciendo un código más compacto y legible.(RODRÍGUEZ)

Otras de las principales características de este lenguaje son:
Extensibilidad: se puede escribir el código en C o en C++ y luego cargarlos a Python. Jython, la implementación de Java en Python integra como una sola pieza a los módulos Java y Python, lo que indica que se puede agrupar tecnologías existentes para el uso en Python.

Potencialidad: Python no es simplemente un lenguaje, viene con una biblioteca estándar extensiva de módulos útiles, las que incluye soporte extensivo para correo electrónico y otras formas de datos de Internet; el HTML, XML y otros lenguajes; http, ftp entre otros protocolos. Usando estos módulos con unas líneas de código Python se llega muy lejos.

Portabilidad: Python corre en la mayoría de los sistemas operativos. Las aplicaciones escritas en puro Python (no en PythonCard) usando las bibliotecas estándar pueden ser detectadas para diferentes plataformas sin recopilación.

Sostenible: la simplicidad elegante de Python produce código no solo elegible, sino que también fácil para rediseñar y modificar, por lo tanto se gasta menos tiempo en entender el código y rescribirlo. Lo cual conduce a los apuros más rápido de problema no resuelto en el software, la integración y el auge más rápido de características nuevas y una base de diseño mejor de código.

Libertad: la calidad y el poder de Python se deben en gran medida a un esfuerzo extraordinario hecho por miles de contribuyentes de todo el mundo. En lugar de controlarse por un equipo propietario Python es accesible para mirar e incorporar nuevas ideas y características. La biblioteca estándar de Python está continuamente siendo mejorada y expandida con soporte para las últimas tecnologías existentes.

1.4.3.1 PythonCard

PythonCard es un entorno de desarrollo rápido de sencillas aplicaciones basado en el lenguaje Python, y que trata de emular al conocido HyperCard de Apple, cuyo objetivo fundamental es organizar la información en forma de cartas y así poner a disposición del usuario cualquier fichero de texto, imagen, sonido o video, de una forma muy gráfica e intuitiva. Como punto de partida se necesita tener una aplicación por lo que se recomienda buscar una aplicación que contenga parte de lo que se necesita.

PythonCard o también conocido como PyCard, es un proyecto de código abierto que se desarrolla bajo los términos de la "*Python 2.2 License*", lo que significa que cualquier persona puede usarlo, acceder al código fuente, y usar cualquier recurso que éste conlleve, siempre manteniendo la autoría y la licencia correspondiente.

Esta herramienta actualmente está en desarrollo. En ella se pueden desarrollar aplicaciones multiplataformas usando el lenguaje Python. Este programa es para desarrollar aplicaciones grafica rápidamente y fácilmente con un mínimo de esfuerzo.

El proceso de creación de la aplicación

La creación de una aplicación con PythonCard comienza con la creación de una estructura básica. Hay dos formar de hacerlo. Una es usar el Editor de Recursos de PythonCard (llamado *resourceEditor.py*), iniciar con la ventana básica vacía, y codificar

desde cero. La otra es copiar la carpeta de una aplicación existente, renombrar unas cuantas cosas básicas, y empezar con un punto de arranque un poco más completo. Tomaré este último curso de acción.

El proceso en resumen

A. Ejecutar el Editor de Recursos para crear o modificar una aplicación existente.
B. Colocar los componentes de la ventana en el Editor de Recursos.
C. Agregar el código ejecutable para los componentes que generarán acciones (botones y/o menús)
D. Limpiar los artefactos del programa copiado.

Crear una aplicación PythonCard stand-alone

Cada archivo .py de PythonCard contiene una definición de clase derivada del *model.Background* de PythonCard así como una variable que permite generar una instancia de esa clase. Una ventana hija es simplemente una instancia de una clase desde dentro de otra aplicación.

Es posible que la ventana hija deba interactuar con los datos de la aplicación principal, así que podríamos estar limitados en cuanto a las restricciones de diseño que nos pueda imponer la aplicación principal. Actuar así es buena idea tratar de separar la funcionalidad de ambas tanto como sea posible para permitir la reutilización de código y simplificar las tareas de depuración. En el ejemplo actual, usaremos la aplicación de ejemplo Minimal como base de nuestra ventana hija en tanto que tomaremos la aplicación *Counter* como aplicación principal.

Minimal.py tiene un control, un campo de texto llamado '*field1*'. Agregaremos un botón a *minimal.py* para que al ser pulsado se restaure el valor del campo de texto de la aplicación principal a 0. También conectaremos los botones de la aplicación principal para actualizar el contenido de *field1* en la ventana hija (*minimal.py*) para que corresponda con el contador de la aplicación.

A pesar de que se trata de interacciones triviales, sirven para demostrar las técnicas involucradas en lograr que dos (o más) ventanas se comuniquen con la otra en una aplicación PythonCard.

Se inicia creando una nueva carpeta donde se guardarán los dos archivos de recursos y los dos archivos de *scripts* de PythonCard. Luego se copia *minimal.py*, *minimal.rsrc.py, counter.py, y counter.rsrc.py* de sus respectivas carpetas en el directorio *samples* hacia esta nueva carpeta. Se pueden dejar los nombres como están, pero en la práctica generalmente deberás cambiar los nombres de los archivos para que coincidan con los de la aplicación en cuestión.

Luego se abre *minimal.rsrc.py* en el *resourceEditor* de PythonCard y agrega un botón Cambia la etiqueta del botón a "*Clear Counter*" y cambia el nombre a *btnReset*. Y por último se guardan los cambios hechos en el archivo de recursos.

1.5 Conclusiones

Hasta ahora hemos visto algunas de las alternativas libres. Su potencialidad y funcionalidad para desarrollar aplicaciones multimedia son visibles claramente. Existe un gran número de herramientas muy potentes que brindan calidad para desarrollar software de este tipo, pero muchos de estos proyectos aún están en un proceso de madurez y renovación, ya sea por el poco apoyo que reciben a nivel mundial (tanto económico como de personal calificado) o su divulgación por los principales medios de comunicación.

Obviamente nos adentramos en un mundo prácticamente desconocido, pero a su vez fascinante por el gran número de alternativas que existen y cada día suman más las personas que dan su paso de apoyo a varios de estos proyectos.

Capítulo 2: Solución propuesta

Capítulo 2
Solución propuesta

2.1 Introducción

En este capítulo se consumará un análisis más profundo de las alternativas libres existentes. Aunque muchas de estas herramientas se encuentran en una fase precoz de su desarrollo, se hace un esfuerzo para que varios de estos proyectos salgan adelante a pesar de la fuerte competencia que existe a nivel mundial del software propietario.

2.2 Posibles herramientas para el IDE

2.2.1 Qt Designer

Es una herramienta muy potente que permite diseñar de una forma muy sencilla y rápida ventanas de diálogo con las librerías Qt, por esta y otras razones que veremos a continuación la escogimos para nuestro trabajo.

Esta herramienta es una aplicación mediante la cual se puede realizar el diseño de aplicaciones GUI de forma gráfica y muy intuitiva. Crear una interfaz con QT Designer es muy sencillo, basta con seleccionar el menú *Archivo -> Nuevo...* y allí se elige un fichero de QT Designer (*.ui) Qt Designer brinda la posibilidad de trabajar con plantillas, de esta forma el trabajo es más fácil, las que se pueden escoger son:

Dialog: Es la plantilla más simple, crea una ventana que incluye únicamente los botones de minimizar, maximizar y cerrar. De esta derivan el resto de las plantillas.

Wizard: Permite realizar aplicaciones paginadas, además de esto incluye los botones de cancelar, anterior (back), siguiente (next) o finalizar (finish).

Dialog with buttons: Igualmente deriva de la primera e incluye los botones de ayuda, apply, ok y cancel.

Características

- Dispone de una paleta de *widgets* muy completa, que incluyen los *widgets* más comunes de las librerías QT. Además cuenta con las librerías para el desarrollo de aplicaciones KDE y *widgets* adicionales.
- El funcionamiento es de estilo selecciona *y* dibuja, es decir, basta con seleccionar un tipo de *widget* en la paleta y luego al ponernos sobre el formulario se dibuja con la geometría que queramos.
- Las propiedades de un *widget* cualquiera se pueden cambiar fácilmente en tiempo de diseño con el panel de propiedades.
- A medida que guardamos los cambios de la interfaz, si editamos el fichero .ui vemos que va cambiando el contenido. La descripción de la interfaz se guarda en ese fichero en formato XML.
- Para hacer que nuestra ventana sea la ventana principal de la aplicación basta con hacer que el *widget* que nos generó como ventana principal herede del nuevo que hemos creado con el QT Designer.

Se pueden añadir los siguientes objetos:

Containers: Se pueden definir como contenedores que se pueden etiquetar con un título y pueden contener texto, imágenes incluso otros objetos tales como botones. Igualmente existen varios tipos Groupbox, Buttongroup, Frame, etc.

Ejemplos de containers:

Inputs: este apartado incluye *LineEdit* (líneas de edición) pueden servir para recoger datos que se introducen por teclado, *MultiLineEdit* (similar), *combobox* (listas desplegables) y otros como *slider*, *spinbox* o *dial.*

Ejemplos de *Inputs*

Displays: en este apartado se incluyen *textlabel* (etiquetas de texto), *pixmapLabel* (imágenes), *progressbar* (barras de progreso) además de *LCDnumber, Line* etc.

Ejemplos de *Displays*

Views: Son objetos que pueden albergar iconos, también en este apartado se incluyen las tablas.

Se pueden realizar señales, conocidas como Slots y Signal:

Los *Slots* y *Signals* (señales) son un mecanismo de comunicación entre objetos, esta es la principal característica de Qt y es el rasgo que hace distintas las librerías Qt del resto de herramientas para la elaboración de GUI, es un mecanismo de comunicación seguro, flexible y totalmente orientado a objetos y por supuesto implementado en C++.

En la programación con GUI se busca que los cambios producidos en un objeto sean comunicados a otros objetos, por ejemplo cuando hacemos clic en un botón para que se cierre una ventana, lo que se hace es posibilitar la comunicación entre los dos objetos.

Otras herramientas de diseño de GUI llevan a cabo la comunicación entre objetos usando los llamados *callbacks*. Un *callback* es un puntero a una función, con este mecanismo si se quiere procesar una determinada función cada vez que ocurre un evento en un objeto, lo que se hace es pasar un puntero a otra función (el *callback*) a la función deseada y será esta la que se encargue de llamar al *callback* en el momento apropiado. Este tipo de comunicación tiene el inconveniente de no ser totalmente seguro puesto que no se sabe si se llamará al *callback* con los argumentos apropiados y además la función que llama al *callback* debe saber exactamente a que *callback* llamar, además es un sistema inflexible y no esta orientado a objetos.

Se puede crear los archivos *.h y los *.cpp:

La aplicación uic permite implementar la clase del modelo realizado con Qt Designer a partir del archivo .ui, es decir obtener los archivos .cpp y .h para ello utiliza la siguiente instrucción en la consola.

Para obtener el archivo de cabecera (.h):

uic -o <ejemploqt.h> <ejemploqt.ui>

Para obtener el archivo de implementación (.cpp):

uic -o <ejemploqt.cpp> -impl <ejemploqt.h> <ejemploqt.ui>

Con esto ya están los archivos implementados, es decir la clase de la ventana creada ya esta definida, pero aun falta algo. La herramienta **moc** (*meta object compiler*) sirve

para obtener archivos de extensión .cpp que se encargan de implementar el mecanismo de slots/signals.
moc -o <moc_ejemploqt.cpp> <ejemploqt.h>

El sistema de pintado Arthur

El Sistema de Pintado de QT se basa en tres clases principales: QPainter, QPaintDevice, y QPaintEngine.

- **QPainter** es la clase encargada de representar operaciones como: *drawLine () y drawRect ().*
- **QPaintDevice** representa un dispositivo que puede pintarse al usar un QPainter.
- **QPaintEngine** suministra la interfaz que el pintor suele dibujar encima de los dispositivos.

El Sistema de pintado de Arthur hace más uso de operaciones nativas de gráficos, ganando beneficios en sus operaciones ya que potencialmente pueden ser realizadas en hardware. Esto nos motivó al trabajo con esta herramienta ya que sus mejoras son significativas sobre muchas implementaciones del software. Entre estos son transformaciones del nativo (Mac OS X y OpenGL).

Otro aspecto significativo de esta alternativa y que dio lugar a su adopción es que con QT + Arthur se puede lograr especificar cómo debería estar un contorno lleno debido a que se puede especificar la textura, el gradiente y otros aspectos de interés.

Unas de las mejoras de Qt fue el QImage como un *PaintDevice* ya que provee un motor basado en píxeles de pintura de trama. Este motor soporta QPainter y unas de sus ventajas principales es que es multiplataforma

QImage provee un pixmap independiente que consiente en tener un acceso directo para los datos de píxel, y puede ser utilizado como un dispositivo de pintura. Qt provee dos clases para datos manipuladores de imagen: QImage y QPixmap.

Qt requiere que un objeto de QApplicación antes de que cualquier dispositivo de pintura pueda ser creado. Los dispositivos de pintura ganan acceso a los recursos de sistema de la ventana, y estos recursos no son inicializados antes de que un objeto aplicativo sea creado.

2.2.2 Primeros pasos con MTASC

Hoy día MTASC puede dar problemas con ciertas cosas como los componentes de Macromedia, pero a medida que ha transcurrido el tiempo su madurez ha ido en aumento y esta fue unas de las razones por las cuales decidimos trabajar con MTASC. La adopción de esta herramienta no solo fue porque es gratuita y de código abierto sino porque tiene calidad de sobra.

MTASC funciona desde la línea de comandos y sólo permite trabajar orientado a objetos, lo que nos obliga a que todos y cada uno de nuestros proyectos sean conjuntos de clases.

Básicamente lo que haría MTASC sería inyectar el código. Aunque por ser un compilador ActionScript dibujar con él puede resultar un proceso difícil debido que todo esto se desarrollaría mediante código, o sea, las animaciones, los dibujos, generar gráficos entre otras cosas. La propuesta de nosotros es contar con clases en C++ con las que posteriormente el IDE y luego este código se traduciría en código ActionScript que MTASC lo compilaría, y de este modo obtener un SWF.

En muchos proyectos que se han desarrollado bajo esta herramienta no se usa como mencionamos en el párrafo anterior debido a que los programadores realizan toda esta parte del dibujo en Flash para posteriormente crear las clases ActionScript que manejarán todos estos elementos, las compila y luego inyectarían el código al SWF con el MTASC, dicho de otra forma más sencilla… MTASC no lo utilizan para el diseño, sino solamente para el código.

Para empezar ejecutaremos la acción ‘mtasc’ desde la línea de comandos de Windows y situándonos sobre el directorio en el que tengamos instalado MTASC, donde nos aparecerá una pequeña ayuda que nos mostrará información sobre las distintas opciones, entre ellas:

- pack (path): compila todos los archivos de la ruta donde esté el paquete.
- cp: añade las rutas indicadas al classpath
- swf nombreArchivo.swf: donde nombreArchivo.swf es el nombre del archivo resultante.
- main: Habilita un punto de entrada a la aplicación.
- header ancho:alto:fps. Dimensiones y velocidad de la película en fotogramas por segundo.
- Archivo_origen.as. La clase principal de nuestra aplicación.

Compilando la primera clase con MTASC

La primera clase que servirá como prueba será la siguiente, la cual guardaremos en un archivo llamado 'Tuto.as' y se encontrará en el mismo directorio en el que se encuentre MTASC.

```
class Tuto {
static var app : Tuto;
function Tuto() {
// creates a 'tf' TextField size 800×600 at pos 0,0
_root.createTextField("tf",0,0,0,800,600);
// write some text into it
_root.tf.text = "Bienvenidos a GALM!";
}
// entry point
static function main(mc) {
app = new Tuto();
}
}
```

Para la compilación ejecutaremos la siguiente instrucción desde la línea de comandos y situados sobre el directorio donde se encuentre MTASC:

```
Mtasc -swf tuto.swf -main -header 800:600:20 Tuto.as
```

¿Qué es AsWing?

AsWing es un *framework* en desarrollo y bajo la etiqueta *open source* que permite crear aplicaciones flash de una manera cómoda e independiente del IDE, o entorno de desarrollo de macromedia. Este *framework* o conjunto de librerías es similar a Java Swing y proporciona una serie de componentes que nos pueden hacer la vida más fácil a la hora de desarrollar con ActionScript 2.0. En la actualidad se encuentra en fase de desarrollo.

Requisitos para utilizar las clases de AsWing con MTASC.

El único y principal requisito es (además de contar con AsWing) tener las clases en el mismo directorio que las clases que incorpora MTASC, o en su defecto indicarle en el *classpath* donde están estas clases.

Un ejemplo de Aswing con MTASC

El ejemplo consiste en una pequeña ventana que contendrá un botón (tipo *JButton*) y un campo de texto (tipo *JtextArea*). El funcionamiento será simple, se creará la ventana, el botón y el campo de texto y se controlará el clic del botón dejando un mensaje en el campo de texto.

[Ver Anexo # 2]

Una vez creada nuestra clase la guardaremos como 'Prueba.as' y situaremos este archivo en el directorio donde tengamos MTASC. Ejecutaremos MTASC y le indicaremos la siguiente instrucción desde la línea de comandos:

mtasc -swf pruebando.swf -main -header 300:300:20 Prueba.as

Donde le indicaremos que nos cree 'pruebando.swf' con las dimensiones indicadas y siendo la clase base Prueba.as.

2.2.3 Ming. Comandos básicos de dibujos

El objeto principal en una película flash es una forma. En Ming, se refiere aun objeto *SWFShape*. En C++, se instancia un objeto *SWFShape* con la siguiente expresión:

*SWFShape*shape=new SWFShape*();*

El dibujo en Ming está basado en pluma, el objeto forma guarda una referencia de la ubicación de la pluma imaginaria en la superficie de dibujo. Se pueden utilizar dos funciones para dibujar líneas con Ming. La función *drawLine* usa un posicionamiento relativo pero también so podría utilizar la función *drawLineTo* que usa coordenadas absoluta (x,y) para dibujar.

Otros de los comandos más útiles son:

shape ->movePenTo (x,y);

el cual mueve la pluma a las coordenadas (x,y) sin dibujar línea (*movePen* se mueve usando posicionamiento relativo), mientras que

shape ->drawCurveTo (cx, cy, ax, ay);

dibuja una curva cuadrática de Bezier simple desde la ubicación actual de la pluma hasta el punto (ax, ay) usando el punto (cx, cy) como punto de control. Con *draCurve* se puede realizar lo mismo, excepto que se usa coordenadas relativas a la posición actual de la pluma.

Estilos de rellenos

A la hora de definir una forma se debe indicar qué porción del espacio entre las líneas debe quedar relleno. El reproductor flash dibuja la forma mientras verifica las líneas, revisa los registros de relleno del lado izquierdo y del lado derecho para ver que estilo de relleno se está aplicando sobre la línea.

Para fijar el relleno del lado izquierdo y del lado derecho se utilizan los métodos *setRightFill* y *setLeftFill*. Estos trabajan como *setLine*, se aplican a todos los bordes que se utilicen. *setRightFill* y *setLeftFill* reciben un argumento, un objeto *SWFFill.*

SWFFill permite crear rellenos sólidos, gradientes y mapas de bits. En un lugar de instancias un *SWFFill* directamente, se solicita un *SWFShape* para que se cree uno para el programador utilizando el método *addFil*l de *SWFShape*. Se debe realizar así ya que el relleno sólo tiene significado dentro del contexto de la forma. A continuación se muestra una tabla con los principales métodos para un objeto de tipo forma.

Métodos	Descripción
*SWFShape*shape = new SWFShape();*	Crea un objeto de tipo forma.
shape->setLine	Determina el estilo de la línea a pintar.
shape->addFill	Añade relleno sólido a la forma
shape->setLeftFill	Añade el relleno por la izquierda de la línea pintada.
shape->setRightFill	Añade el relleno por la izquierda de la línea pintada.
shape->movePenTo	Mueve la pluma de la forma (posición absoluta).
shape->movePen	Mueve la pluma de la forma (posición relativa).
shape->drawLineTo	Dibuja una línea mediante posiciones absolutas.
shape->drawLine	Dibuja una línea mediante posiciones relativas.
shape->drawCurveTo	Dibuja una curva mediante posiciones absolutas.
shape->drawCurve	Dibuja una curva mediante posiciones relativas.

Tabla 1: Métodos para un objeto de tipo forma

Transformación de un relleno

Se pueden realizar modificaciones en los rellenos usando los siguientes métodos: (tenemos un objeto f de tipo *SWFFill*)

f->moveTo (x,y);

mueve el relleno a las coordenadas (x,y)

f->rotateTo (deg);

gira el relleno deg grados, y

f->sacaleTo (xscale (, yscale));

fija la escala de relleno en *xscale* en dirección x, *yscale* en dirección y.

2.2.3.1 Algunas funciones

Ming presenta varias funciones de gran utilidad para nuestro trabajo, en el anexo # 3 se muestran algunas de estas funciones así como su nombre y descripción de cada una de ellas.

2.2.4 UIRA

Importando recursos

Los recursos son muy importantes en las composiciones, los usuarios de Uira deberán poder importar cualquier recurso multimedia que previamente tengan y necesiten para su composición. Eventualmente los usuarios deberán poder importar una composición a la composición y considerando esto como un simple objeto de la composición actual.

Recursos incluidos:

- Objetos Gráficos:
  - imágenes estáticas de mapas de BIT (jpeg, png, ...)
  - imágenes estáticas vectoriales (SVG, ai, ...)
  - imágenes animadas de mapas de BIT (gif, ani ...)
  - animaciones vectoriales (swf, uira, ktoon ...)

- Sonidos
    - *waves*
    - mp3
- Contenido Interactivo
    - programas,
    - composiciones interactiva
    - *java applets*
- Cualquier combinación de estos
    - videos conteniendo sonidos (avi, mpeg ...)

El sistema de importar debe ser entonces lo más extensible posible.

Importando Scripts

Uira nos ofrece características como la edición de *script*, que los usuarios deben poder editar sus *script* donde ellos deseen. Por lo que Uira debe poder ofrecer la posibilidad de importar los script desde otro lugar o desde otra composición (reusabilidad)

Importar *script* no solo significa importar el texto de los *script* sino que también significa la refabricación del *script* de forma tal que use el nombre de objetos correctos y las llamadas correctas. Así la importación de un *script* el ligeramente diferente a la importación de un recurso, pues ellos implican entenderlo y hacer las correspondientes modificaciones para usarlo en la composición actual.

Manejando contenido Multimedia

El contenido multimedia debe poder ser insertado en la composición cuando el usuario lo desee. Una vez insertado o creado in la interfaz, los objetos deben poder ser modificables. El contenido multimedia debe obedecer un esquema de la biblioteca, permitiendo al usuario que defina sus plantillas que serán instanciadas en su momento.

Las plantillas llevan a cabo las características comunes de los objetos, las instancias modifican solamente un subconjunto definido de cada cualidad. Esto permite la reutilización de plantillas como una biblioteca.

2.2.5 SWFC

SWFC forma parte del conjunto de herramientas del SWFTools que su función no es más que generar una película SWF. Su trabajo está muy vinculado con la librería Ming debido a que mediante códigos se pueden desarrollar películas con líneas de tiempo y ActionScript.

A continuación veremos un simple ejemplo creando un fichero SWF. El siguiente script crea una caja roja con borde amarillo.

```
.flash filename="box.swf"
   .box b1 100 100 color=yellow fill=red
   .put b1 pin=center scale=0%
   .frame 100
   .change b1 pin=center scale=100%
   .frame 200
   .change b1 pin=center scale=0%
.end
```

El comando *.box* crea la caja, todos los objetos creados deben ser explícitamente expuestos utilizando *.put* para ser visible. Por otra parte, modifica un objeto ya existente y trabaja gradualmente en el ejemplo anterior, el cambio ocurre sobre los 100 *frames*. Si se quiere cambiar el objeto de forma rápida de un frame a otro se utiliza el comando *.jump*.

Transformación de colores

En SWFC se pueden definir el número de parámetros *.put*, *.change* y *.jump*, dentro de estos están los parámetros de transformación del color rojo, azul y alpha, por tal razón también existe la iluminación que establece el rojo, el verde y el azul al mismo tiempo.
Cada uno consiste en dos partes: el multiplicador y el shift, esta es la sintaxis <multiplicador> y <shift>. Por ejemplo para hacer un objeto 50% más brillante se debe utilizar este valor: *luminace=+128*. Se pueden especificar los valores negativos tanto para <multiplicador> como para <shift>. Esto hace posible convertir el objeto en *luminace=-1 + 125*.

El siguiente ejemplo muestra algunas de las posibles transformaciones.

```
.flash filename="cxform.swf" version=5 fps=25

    .jpeg s1 "photo.jpeg" quality=80%

    .put s1 x=50 y=50 scalex=110 scaley=110
    .frame 50
    .change s1 x=0 y=0 scalex=210 scaley=210 red=-1+255 green=-1+255 blue=-1+255
#invert
    .frame 100
    .change s1 x=100 y=50 scalex=110 scaley=110 red=0 green=+0 blue=+0 #remove
red
    .frame 150
    .change s1 x=0 y=0 scalex=210 scaley=210 red=+0 green=2 blue=-1+255 #amplify
green, invert blue
    .frame 200
    .change s1 x=50 y=100 scalex=110 scaley=110 red=2-128 green=-2+255
blue=+0.7+40 #alien glow
    .frame 250
    .change s1 x=0 y=0 scalex=210 scaley=210 red=8-1024 green=8-1024 blue=8-1024
#palette reduce
    .frame 300
    .change s1 x=0 y=0 scalex=210 scaley=210 red=+0 green=+0 blue=+0 #back to
normal
    .frame 350
    .change s1 x=105 y=105 scalex=0 scaley=0 luminance=0 #fadeout
.end
```

Fuentes

SWFC tiene un soporte de fuente que significa que se puede insertar el texto en una animación. La forma más fácil de cargar una fuente es *.font Arial filename="Arial.ttf"* y tendremos una fuente llamada Arial

Por ejemplo, para un programa *"Hello world"*

```
.flash filename="helloworld.swf"

    .font Arial filename="Arial.ttf"
    .text helloworld font=Arial text="Hello World!"
    .put helloworld
.end
```

Además de la fuente *TrueFonts*, Swfc soporta las fuentes nativas de SWF. Si se tiene un SWF con una fuente que desearías usar realiza lo siguiente:

swfextract file.swf

Después escriba la fuente id y realice:

swfextract -f <fontid> file.swf -o myfont.swf

Formas

Existen muchas formas de crear un contorno, se puede realizar mediante el comando *.textshape* o por el comando *.outline*, nos concentramos en este último.

La sintaxis de la entrada del comando *.outline* es la misma como en SVG. Esto significa que se puede utilizar en editor SVG de su selección (por ejemplo *.inkscape*) para crear estos contornos. Después necesitas extraerlo fuera del fichero .xml/.svg. Están dentro del atributo “d” de la etiqueta “path”

...

```
  <path
     style="fill:#0000ff;fill-opacity:0.75000000;fill-rule:evenodd;stroke:#000000;stroke-
width:1.0000000pt;stroke-linecap:butt;stroke-linejoin:miter;stroke-opacity:1.0000000;"
     d="M 369.90625 299.31250 L 155.21875 513.96875 L 204.40625 513.96875 L
204.40625 819.15625 L 397.31250 819.15625 L 397.31250 623.37500 L 483.68750
623.37500 L 483.68750 819.15625 L 538.40625 819.15625 L 538.40625 513.96875 L
584.56250 513.96875 L 369.90625 299.31250 z M 287.90625 537.00000 L 345.50000
537.00000 L 345.50000 606.09375 L 287.90625 606.09375 L 287.90625 537.00000 z "
     id="rect908" />
```

El contorno se puede llenar con gradiente y mapa de bit, al igual que se puede hacer con el *.textshape*.

[Ver Anexo # 4]

El ejemplo anterior demostró como se llena un contorno con un gradiente. Existen dos tipos de gradientes, el radial y el lineal. El primero tiene un punto centro a un radio y son inmunes a las rotaciones y el segundo tiene un punto de partida, una amplitud y además puede rotarse.

Los gradientes pueden posicionarse libremente dentro del objeto que se desee llenar pasándole los parámetros X; Y y la amplitud y altura al gradiente.

```
.flash filename="gradients2.swf"

   .outline o:
         moveTo -50,-50

         lineTo 0,-45
         lineTo 50,-50

         lineTo 45,0
         lineTo 50,50

         lineTo 0,45
         lineTo -50,50

         lineTo -45,0
         lineTo -50,-50
   .end

   .gradient horizon1 radial x=-50 y=-50 r=100:
      0% cyan
      49% blue
      50% green
      100% cyan
   .end
    .gradient horizon2 radial x=0 y=0 r=50:
      0% cyan
      49% blue
      50% green
      100% cyan
   .end

   .filled o1 outline=o fill=horizon1 line=0
   .filled o2 outline=o fill=horizon2 line=0

   .put o1 x=50 y=50
   .put o2 x=150 y=50
.end
```

Botones

ActionScript resulta manejable cuando se trabaja con los botones SWF. El botón define, en el contexto SWF un objeto sensible al movimiento del ratón o al presionar una tecla. A continuación mostraremos un ejemplo cuando un objeto cambia su forma una vez que el cursor está encima de ellos.

```
.flash filename="button1.swf" fps=50

.box box1 color=white fill=#336633 width=50 height=50
.box box2 color=white fill=#99cc99 width=100 height=100
.button mybutton1
   .show box1 as=shape x=25 y=25
   .show box2 as=hover x=12.5 y=12.5
.end

.frame 1
   .put b1=mybutton1
   .put b2=mybutton1 x=100 red=+255
   .put b3=mybutton1 y=100 green=+255
   .put b4=mybutton1 x=100 y=100 blue=+255
.end
```

El comando .*show* (se puede utilizar dentro de .*button*) tiene una sintaxis muy similar al .*put*. Se puede especificar la posición, transformación de colores, la rotación, la escala, etc. igual que lo hace .*put*.

La única diferencia real de estos comandos es el parámetro *as*, con eso se indica al botón cuando debe desplegar una forma específica. Existen cuatro parámetros permitidos al *as*:

- *Idle:* la forma se despliega cuando el botón es *idle*
- *Hover*: la forma se despliega si el cursor del ratón está dentro de ese botón
- *Área:* esta forma no se despliega. Sirve como el borde a la caja para el botón. El botón se considera activo si el ratón está dentro del área
- *Pressed*: es la forma para desplegar si el usuario da un clic en el botón

Modo Blend

El Modo Blend se introdujo por primera vez en el Macromedia Flash 8. Permite utilizar diferente aritmética cuando viene para poner formas o imágenes en cima de cada una. Los diferentes *Blend Mode* son:

- Normal
- Layer
- Multiply
- Screen
- Lighten
- Darken
- Add
- Subtract
- Difference
- Invert
- Alpha
- Erase
- Overlay
- Hardlight

Por ejemplo con el objetivo de establecer un modo invertido tenemos:

```
.flash filename="invert.swf" fps=50 box=511x127
.jpeg pic stripe.jpg
.put pic
.font arial Arial.ttf
.text txt font=arial text="Test inverted blend mode...
ABCDEFGHIJKLMNOPQRSTUVWXYZ" size=200%

.put txt x=512 y=120 blend=invert
.frame 700
.change txt x=-4096
.end
```

Combinando dos o más archivos SWF usando un archivo master

De los ficheros flash a combinarse, todos excepto uno se empaquetarán dentro de la estructura *sprite* (*Movieclip*) que posteriormente se insertará en el fichero *master* .swf.

esto significa que en términos de árbol cuando se combinan varios ficheros de árboles, uno formará la raíz del árbol, mientras que los otros se adicionarán a la raíz como un subnodo. El usuario tiene que especificar cual de los ficheros se convertirá en la raíz de árbol *master* y cual será adicionado al nodo "*slave*".

El fichero *slave* debe tener un nombre que posteriormente se utiliza para determinar la posición exacta dentro del fichero *master*. Los ficheros *slave* serán convertidos en *sprite* insertado en el fichero *master*, y todas las etiquetas *PlaceObject* en el fichero *master* que corresponda al nombre de un fichero *slave* será actualizado para desplegar correctamente el *sprite slave*.

2.3 Bases de Datos en entornos libres

Las bases de datos no es más que un método para el almacenamiento estructurado de datos. Varias aplicaciones en el mundo utilizan aplicaciones de este tipo, hasta los teléfonos móviles y las agendas electrónicas utilizan tecnología de bases de datos para asegurar la integridad de los datos y facilitar la labor, tanto de usuarios como de los programadores que las desarrollaron.

En otras palabras, una base de datos es un conjunto estructurado de datos que representa entidades y sus interrelaciones. La representación será única e integrada, a pesar de que debe permitir utilizaciones varias y simultáneas.

Sería ilógico desarrollar una aplicación sin contar con una base de datos debido a que trabajar de forma estática sería un proceso obsoleto y engorroso. Por tal motivo se propuso que el trabajo con una base de datos resultaría de gran importancia.

2.3.1 Tendencias actuales

En los años ochenta y noventa los tipos de datos que se podían definir en los Sistemas Gestores de bases de Datos (SGBD) eran muy limitados. La incorporación de tecnologías multimedia nos obliga a que los SGBD acepten atributos de este tipo.

Sin embargo, algunas aplicaciones no tienen suficiente con la asociación de tipos especializados en multimedia, o sea, que se necesitan tipos abstractos de datos TAD.

Los SGBD más actualizados añadieron esta posibilidad y así abrir un amplío mercado al TAD.

Esto nos lleva a la programación orientación a objetos (OOP). El éxito de la OOP en la década de los ochenta, en el desarrollo de software básico, en las aplicaciones de ingeniería industrial y en la construcción de interfaces gráficas con los usuarios, proporcionó que durante los años noventa se extendieran en varios campos de la informática.

A medida que han transcurrido los años las empresas debido al propio desarrollo industrial han ido acumulando gran cantidad de datos de todo tipo. Si estos datos se analizan y se procesan pueden aportar información muy valiosa.

2.3.2 PostgreSQL

Unas de las razones por las cuales se propone trabajar con PostgreSQL no solo es porque su distribución es bajo licencia BSD, sino que este SGBD incluye extensiones de orientación de objetos, aunque como se comentó en el capítulo anterior un SGBDOO total. PostgreSQL está disponible para la mayoría de distribuciones de GNU/Linux.

PostgreSQL tiene fama de ser más complejo de administrar que sus competidores de código abierto, lo que se debe, sobre todo, a que ofrece más prestaciones. En las tareas administrativas como la instalación y la gestión de usuarios, es donde realmente se aprecian las diferencias entre gestores de bases de datos.

Arquitectura de PostgreSQL

En el siguiente gráfico se muestran las diferentes entidades involucradas en el funcionamiento de PostgreSQL.

[Ver Anexo # 5]

PostgreSQL está basado en una arquitectura cliente – servidor. El programa servidor se denomina *postgres* y del lado del cliente existen muchos programas como *pgaccess* y *psql*.

Un proceso servidor *postgres* puede atender exclusivamente a un solo cliente; es decir, de esta forma no hacen falta tantos procesos servidor *postgres* como clientes halla. El proceso *postmaster* es el encargado de ejecutar un nuevo servidor para cada cliente que solicite una conexión.

Se llama sitio al equipo anfitrión (*host*) que almacena un conjunto de bases de datos PostgreSQL. En un sitio se ejecuta solamente un proceso *postmaster* y múltiples procesos *postgres*. Los clientes pueden ejecutarse en el mismo sitio o en equipos remotos conectados por TCP/IP.

De esta forma hemos visto como funciona la arquitectura de PostgreSQL. Es una arquitectura sencilla y a la vez potente y nos puede servir de gran utilidad para el trabajo con bases de datos en entornos libres.

2.3.2.1 ¿Por qué PostgreSQL?

Como mencionamos anteriormente el Sistema Gestor de Bases de Datos PostgreSQL presenta una serie de facilidades útiles para nuestro trabajo, no solo por su distribución bajo licencia BSD sino otros aspectos de gran importancia que veremos a continuación.

Herencia

La herencia se define como el mecanismo mediante el cual se utiliza la definición de una clase llamada “padre”, para definir una nueva clase llamada “hija” que puede heredar sus atributos y operaciones.

PostgreSQL ofrece como característica particular la herencia entre tablas, que permite definir una tabla que herede de otra previamente definida. Esto hace más fácil el trabajo entre tablas.

El cliente psql

Para conectarse con un servidor es necesario un programa cliente. Con la distribución de PostgreSQL se incluye un cliente, *psql*, fácil de utilizar, que permite la introducción interactiva de comandos en modo texto.

El próximo paso sería conocer el nombre de una base de datos residente en el servidor. A continuación mostramos el siguiente comando que permite conocer las bases de datos residentes en el servidor:

[Ver Anexo # 6]

Para realizar una conexión son necesarios los siguientes datos:

- Servidor. De no especificarse su emplea *localhost*.
- Usuario. Si no se especifica se utiliza el nombre del usuario Unix que ejecuta *psql*.
- Base de datos.

Plantillas de creación de Bases de Datos

PostgreSQL tiene definidas dos bases de datos de sistema, *template0* y *template1,* con un conjunto de objetos tales como los tipos de datos que soporta o los lenguajes de procedimiento instalados.

Automáticamente la base de datos *template0* se crea al instalar el servidor, con los objetos por defecto, y la base de datos *template1* se crea a continuación de la anterior con otros objetos por lo que se recomienda heredar siempre de *template1*.

Ventajas de esta situación

- Si se quiere adicionar algún objeto (una tabla o un tipo de datos) a todas las bases de datos, sólo hay que adicionarlo a *template1*, y el resto de bases de datos que se hayan creado a partir de ésta lo tendrán disponible.

- Si se instala un lenguaje sobre la base de datos *template1,* automáticamente todas las bases de datos también usarán el lenguaje. En las distribuciones de Linux es frecuente que se haya realizado de este modo, por lo que no hay necesidad de instalarlo.

2.4 Conclusiones

La potencialidad de estas herramientas es palpable, existe un gran número de variantes que de tener conocimiento de ellas los resultados serían alentadores. Hasta ahora hemos visto cómo estas alternativas han cambiado la forma de concebir la industria de la programación. Depende de nosotros tomar la decisión correcta a la hora de elegir una de estas opciones.

A nivel mundial varias compañías del sector empiezan a invertir y realizan cambios en soluciones basadas en algunas de estas herramientas libres. El software libre es un movimiento imparable que se basa en la cooperación y deja a un lado la competencia. En los últimos años estas herramientas han tenido un aumento sustancial en la industria del software, proporcionando programas de excelente calidad y fiabilidad.

Capítulo 3: Resultados obtenidos

Capítulo 3 Resultados obtenidos

3.1 Introducción

En el mundo del software propietario, el fabricante puede bloquear el acceso al código fuente de sus programas y de esta manera impedir que los usuarios modifique sus programas según sus necesidades. En este capítulo veremos de forma general varios de los beneficios y aportes que nos puede traer la utilización de herramientas libres para el desarrollo de colecciones de productos educativos multimedia.

Cuando trabajamos con herramientas libres los primeros beneficios o resultados que nos vienen a la mente son los económicos, pero sin embargo estos no son los únicos bienes que nos aportan. Muchos pueden ser los aportes de estas alternativas en el aspecto científico y social.

3.2 Beneficios tangibles e intangibles

A la hora de desarrollar un software con herramientas libres los beneficios pueden ser múltiples, ya sean tangibles o intangibles. Como hemos mencionado en capítulos anteriores, en la Universidad de las Ciencias Informáticas la mayoría de las colecciones de software educativo multimedia se desarrollaron bajo software propietario, sin embargo ahora tenemos otra opción, con el conocimiento de estas herramientas tenemos la posibilidad de adentrarnos en el mercado internacional sin temor ninguno debido a que estas alternativas son totalmente libres y su distribución es gratuita por lo cual se pueden desarrollar colecciones de productos de software educativos multimedia.

A medida que transcurran los años estas herramientas ganarán en experiencia y personal calificado para su uso adecuado**.** Cuba no está ausente en este ámbito y los beneficios serán notables ya sea a corto o a largo plazo. Otros de los beneficios es la reutilización del conocimiento, dicho de otra manera esto permite que se reutilice el conocimiento que se ha empleado en el desarrollo de un software en vez de empezar de cero siempre se puede comenzar de un soporte previamente establecido.

3.2.1 Beneficios económicos

Una de las ventajas o beneficios, como se le quiera llamar, de estas alternativas son los beneficios económicos que aportan estas series de herramientas. En años anteriores se desarrollaban aplicaciones que apenas podían sobrepasar la frontera nacional por razones que hemos mencionado anteriormente, pero sin embargo, ahora no ocurre lo mismo.

El factor económico es importante, pero existen razones más profundas por las que el software de Código Abierto resulta preferible a los sistemas propietarios. Una de ellas es evitar la existencia de monopolios del *Software* que restrinjan las opciones que tenemos a la hora de utilizar un ordenador. Un ejemplo conocido es el caso de Microsoft. Cuantos más ordenadores utilicen sus sistemas operativos más dependientes serán los usuarios de una única compañía.(RODRÍGUEZ)

Al cumplimentarse todas estas ideas del proyecto de desarrollar un IDE la ayuda a la economía será de gran importancia porque de esta forma ya no tenemos que prescindir más de software con los cuales nuestro país no tenía licencia. Estas herramientas al ser libres tienen un bajo costo de adquisición y se pueden modificar y adaptar a necesidades específicas.

3.2.2 Defensa de la Patria

Cuba lleva más de cuarenta años con un bloqueo económico impuesto por la mayor potencia del mundo, EE.UU. En ese propio país se encuentran las mayores compañías de software propietario, dígase la famosa y cotizada Microsoft u otras como Adobe y Macromedia, estas dos últimas enfocadas a multimedia.

El bloqueo no ha impedido que nuestro país y específicamente nuestra Universidad se detenga en su andar investigativo y creativo de diferentes alternativas para darle solución a dicha situación. Con el conocimiento de estas herramientas se podrán desarrollar multimedia y colecciones de productos educativos que llegarán a todos los rincones del mundo llevando la verdad de nuestro país y difundiendo el arduo trabajo que en esta pequeña isla se desempeña día a día.

3.2.3 Experiencias y aportes a la Universidad de las Ciencias Informáticas

En los primeros años en la Universidad de las Ciencias Informáticas (UCI) no se tenía conocimiento, o mejor dicho muy poco conocimiento sobre el software libre. A partir del tercer año la estructura docente de la Universidad cambió, y de seis facultades iniciales se extendieron a diez, de ellas, la última centraría su perfil en el movimiento de software libre.

Antes de la realización de este trabajo se conocía muy poco sobre la existencia de estas herramientas para el desarrollo de aplicaciones multimedia con tecnologías libres. En facultades como la 9 se desarrollaron pequeños proyectos multimedia pero ninguno con alternativas libres como hemos mencionado anteriormente.

Sin embargo, la situación ha cambiado. Ahora tenemos conocimientos de estas herramientas y su dominio sobre ellas depende mucho de lo que nosotros seamos capaces de desarrollar.

En estos momentos en la UCI se desarrollan varios proyectos de esta índole, entre los que se destacan el Grupo de Alternativas Libres para Multimedia (GALM) en la que trabajan abnegados estudiantes de las facultades 8 y 9 para brindar su pequeño aporte a la producción y al movimiento del software libre.

3.3 Ventajas de las herramientas libres

El software libre le debe en gran mediada su éxito a nivel mundial a Internet. Mediante la autopista de la información las personas se comunican e intercambian todo tipo de criterios e ideas para la creación de nuevos proyectos.

Las herramientas libres presentan un gran número de ventajas en relación con otras aplicaciones propietarias, estas ventajas en ocasiones son aprovechadas más por los usuarios que por las propias compañías.

Una de las principales ventajas de utilizar estas herramientas para nosotros fue la gran flexibilidad que presentan, o sea, dicho de otra manera, que todo el mundo tiene derecho a su código y adaptarlo a su forma. Esto fue de gran utilidad, un ejemplo de lo

anteriormente planteado fue con UIRA, que sirvió como punto de partida para la futura implementación del IDE. Generalmente el software propietario se distribuye por paquetes y no se adapta a los objetivos y necesidades de quienes los compra.

Otras de las ventajas es que los requisitos de hardware de estas herramientas son muchísimo menores que los que requiere por ejemplo Toolbook o Director, por lo tanto son más barata de implementar.

Uno de los principales problemas del software propietario es la dependencia que se crea entre el usuario y el fabricante, pero esto no ocurre a la hora de utilizar algunas de estas herramientas debido a que garantizan una independencia del proveedor por la disponibilidad de su código fuente. Cualquier interesado no solo en nuestra Universidad sino en cualquier parte del mundo nos puede aportar sus conocimientos o ideas para nuestro trabajo.

Como mencionamos anteriormente gran parte del éxito del software libre se lo debe a Internet donde existe una gran comunidad de programadores y desarrolladores que mediante debates y foros se intercambian código y experiencia. Como es obvio el código es visible para todo el mundo y cualquier error encontrado se reporta rápidamente y de esta manera contribuyen a su mejoramiento.

3.4 Desventajas de las herramientas libres

Aunque quizás no se objetivo de este capítulo analizar algunas de las desventajas de estas herramientas consideramos necesario abordar algunas de las mayores dificultades que existen al utilizar estas alternativas.

Cuando comenzamos a trabajar con esta serie de herramientas algo notamos rápidamente: que este tipo de software no presenta ningún tipo de garantía proveniente del autor, o sea, que el fabricante no ofrece una fiabilidad del producto y se corren varios riesgos. Pero esta no es la única desventaja ya que también a la hora de trabajar con algunas de estas alternativas se requiere un conocimiento previo para lograr un funcionamiento adecuado.

De forma general podemos afirmar que aunque existen estas desventajas la potencialidad de estas herramientas cada día es mayor y su uso se expande por el mundo.

3.5 Selección de las herramientas

Varias fueron las herramientas analizadas a lo largo de esta investigación, escoger con cual trabajaríamos para la futura implementación del IDE no fue una tarea fácil. Por ejemplo, teníamos la opción de PythonCard, es un lenguaje potente y se pueden desarrollar varias aplicaciones pero no tenía nada que ver con nuestros objetivos, aunque es válido aclarar que quizás muchas de estas herramientas no nos aportaron mucho para nuestros intereses pero siempre se aprendió de cada uno de estos proyectos en cuanto a su organización y estructura.

A continuación veremos algunas de las herramientas más importantes y los beneficios y ventajas de cada una. A nuestra consideración hay tres herramientas que se consideran la columna principal del IDE: SWFTool, SWFMill y MTASC.

3.5.1 Resultados y beneficios de las herramientas

Como hemos mencionado SWFTool es un conjunto de herramientas de gran utilidad, entre las que se encuentra el SWFCombine (SWFC) y SWFExtract, ambas las incluimos en nuestro trabajo por su gran aporte.

De ambas herramientas aprendimos mucho, por ejemplo del SWFExtract no tiene una aplicación de sistema, sólo descomprime los contenidos a una carpeta usando la utilidad como WinZip. El Extractor de SWF no hace nada al registro o crea cualquier archivo no deseado. Después de dirigir swfex.exe, se puede abrir un SWF usando la opción "Abierto y Extraer todo". Los archivos serían inmediatamente extraídos a la misma carpeta que el archivo swf.

Como hemos mencionado SWFC es una herramienta que pertenece al paquete de SWFTools y genera SWF nos permitió junto a la librería Ming crear complejas animaciones.

El SWFC se usaría para toda la parte que tiene que ver con el dibujo y animaciones, por tal motivo se convierte en unas de las herramientas principales de nuestro IDE. SWFC recibe un archivo de texto como entrada con un código y genera un SWF, de esta forma permite varios fotogramas y el trabajo con línea de tiempo y así el desarrollo de animaciones y dibujo de formas a través de sencillos scripts, además de otras utilidades.

En el caso de SWFMill genera SWF partiendo de xml y elementos externos que luego con MTASC se le adiciona el código. Estas dos herramientas están muy vinculadas y al igual que el SWFTool constituyen de vital importancia. El SWFMill se utilizaría para el trabajo con la librería de objetos. A la hora de importar un recurso multimedia ya sea un video o un sonido lo colocamos en una librería, que dicha librería no sería más que un xml que el SWFMill recibe y luego genera un SWF al que posteriormente se le insertará el código.

El MTASC inyecta código a un SWF existente, y está orientado a objetos. Cuando el cliente comience con la programación de sus clases en ActionScript para determinado proyecto en nuestro IDE, se usaría el MTASC para la compilación de todo este código y se inyectaría en el SWF que genera el SWFC junto al SWFMILL. MTASC presenta un problema, y es que solo se puede insertar el código en el primer fotograma de la película.

Ni el MTASC ni el SWFMill se utilizarían para dibujar o crear animaciones, el primero se utilizaría simplemente para agregar interactividad a la película y para la programación de su comportamiento, al ser orientado a objetos las ventajas serían enormes. Y como ya conocemos el SWFMill genera un SWF.

Aunque no lo mencionamos al principio es necesario destacar el trabajo de la librería Ming. Esta librería constituye la base del SWFC, por lo que la usaríamos indirectamente, esta librería se ha utilizado mucho con Qflash y ha demostrado su potencialidad.

3.5.2 Herramientas colaborativas

Las herramientas anteriores no fueron las únicas que nos aportaron su ayuda y beneficio, hay otro grupo de herramientas que quizás su aporte no fue como las ya mencionadas pero por eso no dejan de ser importantes para nuestro trabajo.

Cuando se decidió trabajar con Qt observamos que esta librería podía ser de gran utilidad debido a que sus características se adaptaban a nuestras necesidades, como por ejemplo que dispone de una paleta de *widgets* muy completa, además de contar con librerías para el desarrollo de aplicaciones KDE y *widgets* adicionales.

Varias herramientas, aunque no están incluidas directamente en el IDE, dieron también su aporte a nuestro proyecto. Uno de estos casos es el de Ktoon, un proyecto que desde su fundación trabaja arduamente en su superación y consagración.

No queremos dejar pasar en este punto el trabajo de la web como medio de comunicación y verlo como una herramienta que nos aportó más de lo que concebimos al principio. En los proyectos libres se acostumbra a crear documentación con manuales o tutoriales y de esta forma recopilar la mayor información posible.

Las famosas Wiki han alcanzado un punto extremadamente importante en este ámbito y se han convertido en una herramienta de colaboración que permiten al usuario crear y modificar su contenido en la web.

3.5.3 Importancia de estas herramientas

Es cierto que muchas de las herramientas y proyectos descritas en este trabajo no han alcanzado un grado de madurez que les permite colocarse en un lugar privilegiado a nivel mundial pero por tal motivo ninguna de estas herramientas deja de ser importante para su investigación.

Ya hemos mencionado que para nuestro país constituye una necesidad la migración para el software libre debido a razones conocidas. El trabajo con estas alternativas nos brinda la posibilidad de ampliar nuestro mercado y crear una base en reputación en la industria del software a nivel mundial.

Estas herramientas cobran un papel fundamental en la industria actual del software debido a que cada día muchos países se ven obligados a tomar esta vía del software libre. Nuestro país no ha estado ausente a esta alternativa y específicamente en la Universidad de las Ciencias Informáticas desde hace algún tiempo se viene realizando un trabajo exhaustivo con el software libre, pero no así con herramientas para el desarrollo de creaciones multimedia.

3.5.4 Aportes del software libre

El software libre tiene disponible su código fuente, por lo tanto las mejoras son notables debido a que se adapta a nuestras necesidades y precisamente por su carácter de tener el código abierto dificulta la entrada de código malicioso.

El costo es algo primordial y que no podemos olvidar a la hora de realizar un trabajo y ver lo que nos aporta trabajar y desarrollar programas bajo software libre. Cuando se desarrolla una aplicación con estas características su costo será mucho menor que si fuese con software propietario debido entre otras cosas a que el software libre no tiene costo de licencia y es un dinero que se ahorra considerablemente.

Otro aspecto que se considera importante es que los requisitos de hardware son menores y menos exigentes que los de programas propietarios. Por ejemplo, Ktoon, programa de animación en 2D tiene como requisitos 256 de RAM, un micro superior a una velocidad de 800 Mhz y un espacio libre en disco de 5Mb.

La personalización del software libre es un área que se desenvuelve mejor que el software propietario, precisamente por tener dominio total del código. El software propietario como tal no se vende, el usuario lo que adquiere es una licencia que le permite actuar con ciertas libertades, la gran diferencia con el software libre es que con el software propietario el cliente lo ejecuta tal y como viene, o sea que no se puede modificar incluso, si tuviese errores se ejecuta de igual forma pero sin embargo esto no ocurre con el software libre debido a que se puede ejecutar y modificar todas las veces que el usuario lo desee.

3.6 Resultados con las Bases de Datos

Era necesario crear un medio de almacenamiento dinámico ya que proporcionaría un mejor trabajo con el IDE. La selección de trabajar, o mejor dicho, de proponer el trabajo con PostgreSQL no fue obra de la casualidad.

Este SGBD es extensible. Es posible agregar nuevos tipos de datos y funciones al servidor que se comporten como los ya incorporados. También es posible insertar nuevos lenguajes de programación del lado del servidor para la creación de procedimientos almacenados. Todas estas ventajas hacen que muchos programadores lo elijan para el desarrollo de aplicaciones en todos los niveles.

PostgreSQL está aún en evolución, se espera que en futuras versiones se incluyan nuevas características y mejoras al diseño interno del SGBD.

3.6.1 Migrando a entornos libres

La migración de bases de datos alojadas en productos comerciales a PostgreSQL se facilita gracias a que soporta ampliamente el estándar SQL. PostgreSQL cuenta con una serie de características atractivas como son la herencia de tablas (mostrada en el capítulo 2), un rico conjunto de tipos de datos que incluyen arreglos, BLOB, tipos geométricos y de direcciones de red. PostgreSQL incluye también el procesamiento de transacciones, integridad referencial y procedimientos almacenados. En concreto, hay procedimientos documentados para migrar los procedimientos almacenados desarrollados en lenguajes propietarios de bases de datos comerciales (PL/SQL) a PL/PGSQL.

Su flexibilidad, relativa facilidad de uso, excelente documentación y robustez la hacen una buena elección en múltiples escenarios, y eso sin siquiera entrar en temas de vital importancia para nosotros y enunciar los beneficios que tiene que sea *open source*.

Aunque PostgreSQL no es totalmente orientado a objeto presenta varias prestaciones de ese paradigma de la programación. Esto trae como resultado que el modelado sea más natural de la realidad, facilita la reutilización de componentes de software y sobre todo que ofrece mecanismos de abstracción para mantener controlable la construcción

de sistemas de objetos. Aun sin haber alcanzado una madurez total, la programación orientada a objeto es el paradigma que mejor permite solucionar la gran variedad de problemas que existen en el desarrollo de la industria del software.

3.7 Conclusiones

Como se puede observar los beneficios de estas herramientas son evidentes. Quizás los resultados no sean alentadores los primeros años por la falta de experiencia, pero sí podemos asegurar que el trabajo con estas alternativas nos traerán una serie de ventajas con las cuales antes no contábamos.

Nuestra Universidad como centro investigativo debe desarrollar un trabajo más profundo en el software libre y sus alternativas, no solo en herramientas para el desarrollo de aplicaciones de corte educativo sino en otros campos y los resultados serán alentadores debido al gran auge que tiene a nivel mundial el software libre.

Conclusiones

Conclusiones

En este trabajo se le dio cumplimiento a los objetivos propuestos al principio de la investigación, por lo que:

- Se realizó un estudio detallado de las herramientas propuestas para el trabajo con cada una de ellas.
- Su fiabilidad para desarrollar aplicaciones multimedia son indiscutibles. Actualmente hay una gran variedad de herramientas muy poderosas que su aporte puede ser excelente en las aplicaciones multimedia.
- No se había realizado ningún trabajo investigativo con ninguna de estas herramientas en la Universidad de las Ciencias Informáticas.
- Representaría un ahorro significativo al país si nos adentráramos más en el estudio de alternativas libres, no solo de herramientas para el desarrollo de aplicaciones multimedia, sino en otros campos de interés.
- Varios países empiezan importantes inversiones y a buscar variantes en estas alternativas libres.

Recomendaciones

Recomendaciones

Luego de concluir con la investigación de este trabajo se recomienda:

- Seguir indagando las herramientas libres que sirvan para el desarrollo de productos educativos multimedia.
- Publicar los resultados de esta investigación para tener un mayor conocimiento de estas alternativas.
- Presentar el trabajo en eventos científicos.
- Asociar varias de estas herramientas con cursos optativos de la facultad 9.

Bibliografía

Bibliografía

Camps Paré, Rafael... [et al.]. Bases de datos en entornos libres. Madrid, 2003.

Categorías de software libre y no libre. [en línea] Copyright (C) 1996, 1997, 1998, 2001, 2006 Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110, USA. Última Actualización: 2006/12/16.

http://www.gnu.org/philosophy/categories.es.html#Non-CopyleftedFreeSoftware

Consulta: feb. 2007.

Copiar o no copiar, he ahí el dilema?. Revista TodoLinux 23:12-13 noviembre 2002.

Culebro Juárez, Monserrat, W. Gómez Herrera, S. Torres Sánchez. Software libre Vs software propietario, ventajas y desventajas. México D.F.: [s.e.], 2006.

Eric S., Raymond. La catedral y el bazar. 21 p. (Artículo).

González Barahona, Jesús M. Licencia Pública GNU. [en línea] Copyright (C) 1989, 1991 Free Software Foundation, Inc., EEUU. Copyright (C) 1989, 1991 Free Software Foundation, Inc., EEUU. http://gugs.sindominio.net/licencias/gples.html Consulta: abr. 2007.

Introducción a Jclic. [en línea] http://www.juntadeandalucia.es/averroes/jclic/ Consulta: dic. 2006.

Introducción a PythonCard [en línea] Última modificación 2005-08-17 http://pythonmexico.org/index_html/doc/PythonCard/walkthrough1.html Consulta: abr. 2007.

JoanGarnet. [en línea] http://www.joangarnet.com/blog/ Consulta: feb. 2007.

KToon: Herramienta de Animación en 2D. [en línea] Última Actualización (Friday, 08 June 2007)

http://ktoon-es.toonka.com/index.php?option=com_frontpage&Itemid=1 Consulta: junio 2007.

Libming Documentation. [en línea] http://www.libming.org/docs/libming/index.html Consulta: may. 2007.

Masi Hernández, Josi. Software libre: técnicamente visible, económicamente sostenible y socialmente justo. Madrid: Editorial: Infonomia, 2005. 191 p.

Matías Sánchez, E. Breve introducción al software libre. [s.l.]: Octubre, 2004 (artículo).

¿Qué es el JClic? [en línea] Xarxa Telemática Educativa de Catalunya http://clic.xtec.es/es/jclic/howto.htm Consulta: ene. 2007.

Sierra Rodríguez, Pío, R. Enríquez Miranda. Guía práctica para usuarios open source. [s.l.]: Editorial: Anaya Multimedia, 383 p.

Swftools: SWF manipulation and generation utilities. [en línea] http://www.swftools.org/documentation.html Consulta: may. 2007.

Uira: flash en linux. [en línea] http://www.etynos.org/web/2006/07/09/uira-flash-en-linux/ Consulta: abr. 2007.

Wheeler, David A. ¿Por qué usar programas abiertos? ¡Atención a las cifras!. 17 p. (Artículo).

Referencias bibliográficas

Referencias bibliográficas

WIKIMEDIA FOUNDATION, I. *Software educativo*, [Web]. 2007. [Disponible en: http://es.wikipedia.org/wiki/Software_educativo

CULEBRO JUÁREZ, M. *Software libre vs software propietario*. 2006. 158 p.

HERNÀNDEZ, J. M. I. *Software libre: técnicamente viable, económicamente sostenible y socialmente justo.* Primera Edición. Infonomia, 2005. 191 p.

RODRÍGUEZ, P. S. *Guía práctica para usuarios Open Source*. ANAYA MULTIMEDIA, 383 p.

SÁNCHEZ, E. M. Breve introducción al software libre, 2004.

WIKIMEDIA FOUNDATION, I. *AJAX*, [Web]. 2007a. [Disponible en: http://es.wikipedia.org/wiki/AJAX

Biblioteca Ming, [Web]. 2007b. [Disponible en: http://es.wikipedia.org/wiki/Biblioteca_Ming

Código abierto, [Web]. 2007c. [Disponible en: http://es.wikipedia.org/wiki/C%C3%B3digo_abierto

Licencia Open Source, [Web]. 2007d. [Disponible en: http://es.wikipedia.org/wiki/Licencia_Open_Source

Python, [Web]. 2007e. [Disponible en: http://es.wikipedia.org/wiki/Python

Software educativo, [Web]. 2007f. [Disponible en: http://es.wikipedia.org/wiki/Software_educativo

SWFTools, [Web]. 2007g. [Disponible en: http://es.wikipedia.org/wiki/SWFTools

Anexos

Anexos

Anexo #1 Condiciones para que un programa se considere Open Source

- Libre distribución. No se puede impedir la venta o distribución del programa o parte de él. Así mismo, tampoco se puede exigir el pago de un canon o tasa a cambio de su distribución por parte de terceros.

- Código fuente. El programa debe incluir su código fuente y no se puede restringir su redistribución.

- Trabajos derivados. No debe impedirse realizar modificaciones o trabajos derivados del programa y debe permitirse que éstos sean distribuidos bajo mismos términos del software original.

- Integridad del código de fuente original. Puede exigirse que una versión modificada del programa tenga un nombre y número de versión diferente que el programa original para poder proteger al autor original de la responsabilidad de estas versiones.

- No discriminación contra personas o grupos. Las condiciones de uso del programa no pueden discriminar contra una persona o un grupo de personas.

- No discriminación contra usos. No se puede negar a ninguna persona hacer uso del programa para ningún fin como, por ejemplo, comercial o militar.

- Distribución de la licencia. Los derechos del programa deben aplicarse a todos quienes se redistribuyen el programa sin ninguna condición adicional.

- La licencia no debe ser específica de un producto. Los derechos garantizados al usuario del programa no deben depender de que el programa forme parte de una distribución o paquete particular de software.

- La licencia no debe restringir otro software. La licencia no debe poner restricciones en otros programas que se distribuyen junto con el software licenciado.

- La licencia debe ser tecnológicamente neutra. No puede existir ninguna disposición de la licencia que obligue al uso de una tecnología concreta.

Anexo #2: Ejemplo de Aswing con MTASC

```
// importamos las clases necesarias
import org.aswing.BorderLayout;
import org.aswing.Event;
import org.aswing.JButton;
import org.aswing.JTextArea;
import org.aswing.JFrame;
import org.aswing.utils.*;
// creamos la clase a compilar que heredará de JFrame
class Prueba extends JFrame{
// Creamos dos variables de tipo JButton y JTextArea
private var miBoton:JButton;
private var miTexto:JTextArea;
// El constructor de la clase
public function Prueba() {
super(_root, true);
// Creamos los botones
miBoton = new JButton(”JButton1″);
miTexto = new JTextArea(”Pulsa el botón”);
// Los colocamos
getContentPane().append(miBoton, BorderLayout.SOUTH)
getContentPane().append(miTexto,BorderLayout.CENTER)
// Controlamos el clic del botón
miBoton.addEventListener(JButton.ON_PRESS, Delegate.create(this,
marcarPulsacion));
}
// Función que se ejecutará al hacer clic en el botón
private function marcarPulsacion(eventObj:Event):Void {
miTexto.setText(”Botón Pulsado”);
}
// Main de la clase
public static function main(Void):Void {
var myWindow:Prueba = new Prueba();
```

```
myWindow.setLocation(50, 50);
myWindow.setTitle('Aswing Prueba..');
myWindow.setSize(100, 100);
myWindow.show();
}
}
```

Anexo #3: Algunas funciones de Ming

Nombre

Ming_setSWFCompression

Sinopsis

#include <ming.h>

int Ming_setSWFCompression(int level);

Parámetros

int level

New compression level.

Descripción

Esta función establece el valor del nivel de compresión a utilizarse cuando se genera salida. El nivel debe ser el valor entre 1 y 9 que corresponda a los niveles de compresión utilizado por *libz.*

Returns

Valor previo

Nombre

Ming_getScale

Sinopsis

#include <ming.h>

float Ming_getScale(void);

Descripción

Toma el factor de la escala global

Returns

El factor de la escala global corriente

Nombre

Ming_setScale

Sinopsis

#include <ming.h>

void Ming_setScale(float scale);

Parámetros

float scale

New scaling factor.

Descripción

Establece el factor de la escala global

Nombre

Ming_setCubicThreshold

Sinopsis

#include <ming.h>

void Ming_setCubicThreshold(int num);

Parámetros

int num

Descripción

Establece el umbral utilizado cuando se aproxima a la forma

Nombre

Ming_setWarnFunction

Sinopsis

#include <ming.h>

SWFMsgFunc Ming_setWarnFunction(SWFMsgFunc warn);

Parámetros

SWFMsgFunc warn

Descripción

Esta función se activa cuando ocurre un warning dentro de la librería

Returns

La función del warning previamente establecida

Nombre

Ming_setErrorFunction

Sinopsis

#include <ming.h>
SWFMsgFunc Ming_setErrorFunction(SWFMsgFunc error);

Parámetros

SWFMsgFunc error

Descripción

Esta función se active cuando ocurre un error dentro de la librería. Por defecto la función imprime el mensaje de error.

Returns

La función del error previamente establecida

Nombre

Ming_useSWFVersion

Sinopsis

#include <ming.h>

void Ming_useSWFVersion(int version);

Parámetros

int version

Flash version.

Descripción

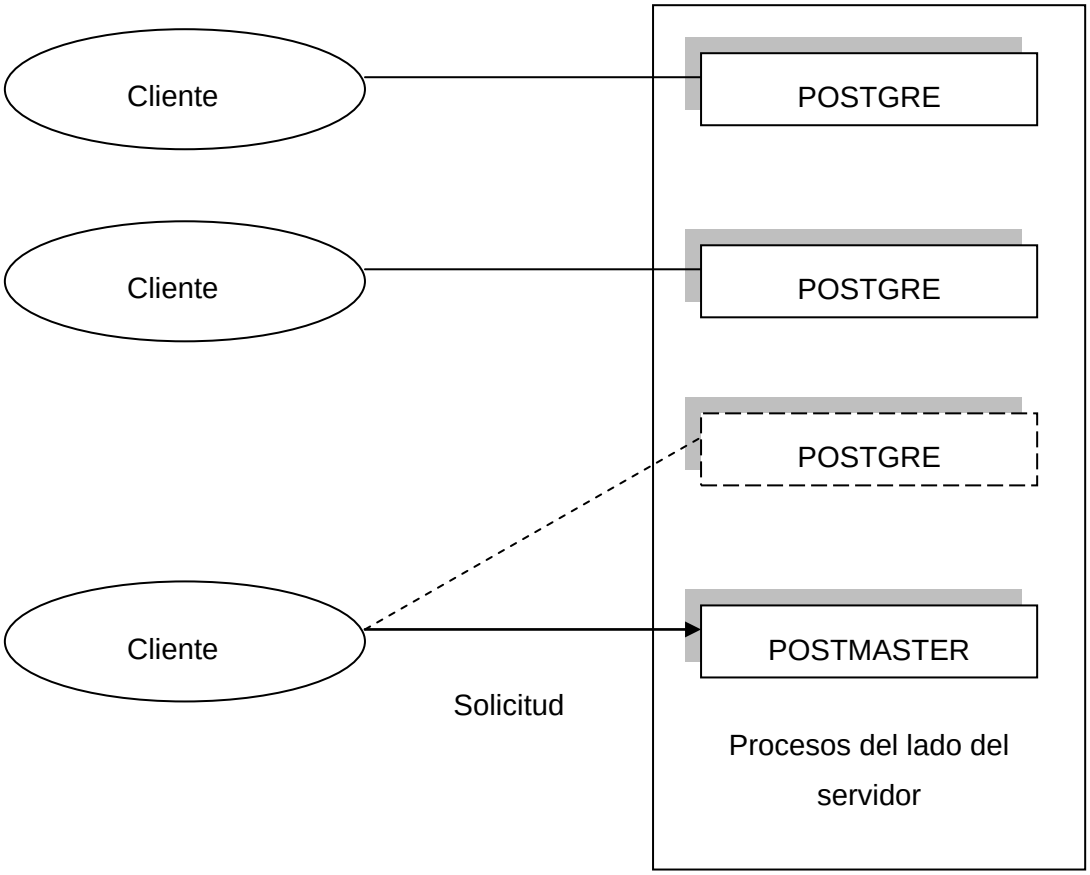
Esta función establece la versión del SWF a producirse en la librería.

Anexo #4

```
.flash filename="gradients.swf"


   .outline star:
      M 521,640 C 502,678 370,546 328,554 C 270,566 152,731 93,722
            C 51,716 147,549 127,512 C 98,460 -107,400 -117,341
            C -124,299 63,339 93,308 C 133,265 127,50 180,23
            C 218,3 238,195 276,213 C 330,238 532,166 575,208
            C 605,238 429,316 424,358 C 416,417 547,587 521,640
   .end

   .gradient rainbow:
      0% blue
      25% green
      50% yellow
      75% orange
      100% red
   .end

   .gradient fire radial:
      0% white
      50% yellow
      100% red
   .end

   .gradient horizon:
      0% cyan
      49% blue
      50% green
      100% peru
   .end

   .gradient transparent:
      0% #ff000000
      100% #ff0000ff
   .end

   .box scenery fill=horizon width=200 height=200
   .box bar fill=transparent width=240 height=20
   .filled star1 outline=star fill=rainbow line=1
   .filled star2 outline=star fill=fire    line=1

   .put scenery rotate=90%
   .put star1 scale=10% x=-70
   .put star2 scale=10% x=-180 y=110
```

```
.put bar x=-180 y=10 rotate=45
```

Anexo #5: Entidades involucradas en PostgreSQL



Anexo #6: Comando que permite conocer las BD del servidor

~$ psql -l

List of databases

Name	Owner	Encoding
demo	postgres	SQL_ASCII
template0	postgres	SQL_ASCII
template1	postgres	SQL_ASCII

(3 rows)

~$

Glosario de términos

Glosario de términos

ActionScript: es un lenguaje de programación orientado a objetos (OOP), utilizado en especial en aplicaciones Web animadas realizadas en el entorno Macromedia Flash, la tecnología de Macromedia para añadir dinamismo al panorama Web.

API: (del inglés **Application Programming Interface - Interfaz de Programación de Aplicaciones**) es un conjunto de especificaciones de comunicación entre componentes software. Representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general, por ejemplo, para dibujar ventanas o iconos en la pantalla.

Base de Datos: Conjunto estructurado de datos que representa entidades y sus interrelaciones. La representación será única e integrada, a pesar de que debe permitir diversas utilizaciones, sigla: *BD*

Document Object Mode: (en español 'Modelo de Objetos de Documento'), frecuentemente abreviado **DOM**, es una forma de representar los elementos de un documento estructurado (tal como una página Web HTML o un documento XML) como objetos que tienen sus propios métodos y propiedades. El responsable del DOM es el *World Wide Web Consortium* (W3C). En efecto, el DOM es una API para acceder, añadir y cambiar dinámicamente contenido estructurado en documentos con lenguajes como ECMAScript (Javascript).

Ecma Internacional: es una organización internacional basada en membresías de estándares para la comunicación y la información. Adquirió el nombre *Ecma International* en 1994, cuando la *European Computer Manufacturers Association* (ECMA), cambió su nombre para expresar su alcance internacional.

Entorno de Desarrollo Integrado: o en inglés *Integrated Development Environment* (**IDE**) es un programa compuesto por un conjunto de herramientas para un programador. Un IDE es un entorno de programación que ha sido empaquetado como

un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica GUI.

F4L: fue un programa que se desarrolló como una propuesta para Linux del Macromedia Flash, solo que su divulgación no ha sido muy difundida. Existen otras propuestas de este tipo bastante elaboradas como Ktoon y otras que van comenzando como es Qflash.

GNU Assembler: es el ensamblador del proyecto GNU. Es el back end por defecto del GNU Compiler Collection y es usado para compilar Linux y otros sistemas operativos como el sistema operativo GNU. Es una parte del paquete GNU Binutils, y se puede acceder con el comando **as** cuando se escribe desde el shell. Al igual que el resto de aplicaciones GNU, es un software libre, y se lanza bajo la licencia GNU General Public License.

GNU Compiler Collection: es un conjunto de compiladores creados por el proyecto GNU. GCC es software libre y lo distribuye la FSF bajo la licencia GPL.

GNU GPL: (*General Public License* o licencia pública general) es una licencia creada por Free Software Foundation a mediados de los 80, y está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

IDLE: Entorno Integrado de Desarrollo para Python.

KDevelop: es un entorno integrado de desarrollo para sistemas Linux y otros sistemas Unix. Tiene licencia GPL. KDevelop 3.0 ha sido reconstruido completamente desde los cimientos, se dio a conocer junto con KDE 3.2 en diciembre de 2003. A diferencia de muchas otras interfaces de desarrollo, KDevelop no cuenta con un compilador propio, por lo que depende de Gcc para producir código binario.

Objective CAML: también llamado **Ocaml** o **O'Caml**, es un lenguaje de programación avanzado de la familia de los lenguajes ML, desarrollado y distribuido por el INRIA en Francia. Ocaml admite los paradigmas de programación imperativa, programación funcional y programación orientada a objetos.

OpenGL: es una especificación estándar que define una API multi-lenguaje multi-plataforma para escribir aplicaciones que producen gráficos 3D, desarrollada originalmente por Silicon Graphics Incorporated (SGI). OpenGL significa **Open Graphics Library**, cuya traducción es **biblioteca de gráficos abierta**.

OSI: **Open Source Initiative** es una organización dedicada a la promoción del software abierto. Fue fundada en febrero de 1998 por Bruce Perens y Eric S. Raymond.

Qflash: fue una propuesta para realizar un clon de Macromedia Flash para Linux, esto se logró, principalmente, en la parte de la interfaz, que es en gran parte muy parecida a la de Macromedia Flash, pero aún carente de muchas de sus funcionalidades.

Scripting remoto: es una tecnología que permite a los scripts que se ejecutan dentro de un navegador Web intercambiar información con el servidor. Los scripts locales pueden invocar scripts en el servidor remoto y procesar la información devuelta. El **scripting remoto** ha sido desarrollado por Microsoft.

SQL92: SQL92 define Palabras Clave para el lenguaje que tienen un significado específico. Algunas palabras están reservadas, lo cual indica que su aparición esta restringida sólo en ciertos contextos. Otras Palabras clave no están restringidas, lo cual indica que en ciertos contextos tienen un significado específico pero no es obligatorio.

XHTML: acrónimo inglés de e**X**tensible **H**yper**t**ext **M**arkup **L**anguage (lenguaje extensible de marcado de hipertexto), es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas Web. XHTML es la versión XML de HTML, por lo que tiene, básicamente, las mismas funcionalidades, pero cumple las especificaciones, más estrictas, de XML.

***XML**:* acrónimo en inglés de e**X**tensible **M**arkup **L**anguage («lenguaje de marcas extensible»), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML.

XMLHttpRequest: es un API que puede invocarse desde JavaScript, JScript, VBScript y otros lenguajes de script incluidos en un navegador Web, que se usa para transferir y manipular datos XML hacia y desde el navegador Web, estableciéndose un canal de conexión independiente entre el lado del cliente de la página Web y el servidor.