Curso Administración GNU/Linux
Nivel I

Trabajo Final:

Emulación de aplicaciones Windows (Wine)

Alumnos:

- **Rodrigo Fernandez**
- **Rafael Vicentín**

Facultad de Ciencias y Tecnología - Oro Verde

2009

Indice

Introducción

Este trabajo se inició con la búsqueda de herramientas para resolver dos situaciones que suelen darse al proponernos migrar desde un Sistema operativo propietario como Microsoft Windows, hacia una nueva plataforma de software libre tal como Debian Linux.

Estas situaciones son:

1) Migración desde Windows a Linux cuando se cuenta con software gratuito diseñado para Windows sin tener a la vista un sustitutivo Linux; o se desea mantener en uso una aplicación para Windows.
2) Casos en los que se cuenta con desarrollos a medida o aplicaciones especiales -propios o de proveedores- que han sido desarrollados con herramientas basadas en Windows.

Ambos casos están emparentados en un requerimiento común: ejecutar sobre Linux aplicaciones desarrolladas para plataforma Windows.

PARTE 1 - CARACTERÍSTICAS, APLICACIÓN E HISTORIA (Información basada en la documentación oficial de Wine)

Qué es Wine

Wine es una capa de traducción -o programa cargador- que aporta capacidades para ejecutar aplicaciones Windows sobre Linux y otros sistemas operativos compatibles con POSIX [1].

Wine *(acrónimo de **WI**ne is **N**ot an **E**mulator* [2]*)* permite la ejecución de programas de Microsoft Windows (incluyendo ejecutables de DOS, Windows 3.x y Win32) sobre Linux. Wine carga y ejecuta un binario de Microsoft Windows, y una librería (llamada Winelib) implementa las llamadas a la API de Windows usando sus equivalentes Linux o X11. La librería puede también puede utilizarse para que ejecutables nativos Linux puedan soportar código Win32.

Los programas originarios para Windows corriendo “sobre” Wine, actúan como lo harían en aquél, corriendo sin caídas de rendimiento o la demanda de uso de memoria de un emulador, con un “look and feel” similar a otras a otras aplicaciones del escritorio.

Wine es software libre, publicado bajo la licencia GNU LGPL. El proyecto Wine comenzó en 1993 como un modo de dar soporte a aplicaciones para Windows 3.1 que se deseaba hacer correr sobre Linux. Bob Amstadt, el coordinador original, fue reemplazado tempranamente por Alexandre Julliard, que lo ha dirigido desde entonces. Durante los años, han sido añadidas compatibilidades (ports) a otros sistemas Unix, junto con el apoyo a Win32 cuando las aplicaciones de Win32 se hicieron populares.

Wine está bajo desarrollo, y todavía no resulta totalmente adecuado para uso general. Sin embargo, muchas personas lo encuentran útil para correr un número creciente de programas de Windows. La página oficial de sus desarrolladores incluye una Base de Datos de Aplicaciones *(Aplication DataBase)* que contiene informes reportando éxitos y fracasos para cientos de programas de Windows corriendo sobre Wine, así como una Base de Rastreo de Errores de Programación *(Bug Tracking DataBase)* para un listado de situaciones conocidas. Contiene también una página de Estado *(Status)* para una visión global del progreso de desarrollo de Wine. La misma web incluye un Foro (*Wine Forum)* al que concurre un amplio espectro de desarrolladores de Wine.

Historia de Wine

Los inicios de Wine se remontan a 1993. Microsoft había conducido con éxito su programa de Windows a la vanguardia de ordenadores personales. IBM tenía esperanzas que OS/2 se pondría de moda, pero confesaba que el soporte para programas de Windows era necesario por lo que incorporó esa capacidad en su producto. El movimiento de software gratuito engendrado en los años ochenta ganaba terreno rápidamente, cuando mucha gente descubrió que era posible correr un sistema operativo multiusuario y multitarea en un ordenador personal.

La adquisición de Tecnologías Praxsys por parte de Sun en septiembre de 1992 llevó al desarrollo de un producto llamado Wabi que permitió a los usuarios del Solaris x86 y Solaris 2.2 para SPARC, correr aplicaciones de Windows fuera del paquete original (Windows). Sun presentó ese software en la Conferencia de Desarrolladores Solaris 1993. Otros productos que entonces permitían correr programas de Windows requerían nivel de emulación e instalación de DOS y Windows. Wabi era único que permitía que el *“Windows windowing”* llamara para ser traducido directamente a llamadas X de Windows. Emulando el resto del código de x86 era posible ejecutar realmente programas de Windows más rápido que en una estación de trabajo RISC. Los rasgos más avanzados de Wabi incluyeron la tecnología “Bitstream's font handling” para manejar fuentes TrueType.

Los usuarios del sistema operativo del advenedizo Linux comenzaron a discutir la posibilidad de un acercamiento similar en junio de 1993. Entonces, las posibilidades de Wabi virando a babor a Linux, no resultaron extrañas para nadie. Una lista de direcciones fue establecida para facilitar la discusión. El nombre "Wine" fue rápidamente adoptado. Entre los desarrolladores iniciales se incluyeron algunos de los primeros hackers del kernel de Linux, como Eric Youngdale y David Metcalfe.

Otros nombres reconocibles incluyeron a Alexandre Julliard que ahora conduce Wine y Miguel de Icaza que alcanzó fama por su participación en GNOMO. Bob Amstadt encabezó el desarrollo.

El trabajo inicial consistió en conseguir un cargador de programa que trabajara pudiendo correr los binarios de 16 bites de Windows. Aquel trabajo fue encabezado principalmente por Bob. La participación de Alexandre se centró alrededor de las funciones de combinación de ventanas *(merging windowing)* escritas por Peter MacDonald en Tcl/Tk.

Se progresó rápidamente, y dentro de los 6 primeros meses era posible correr el Solitario. Noviembre de 1993 también vio el primer port de Wine a otra arquitectura: John Brezak presentó remiendos para permitir que Wine corriera en NetBSD. Bob estimó que, con la tasa corriente de desarrollo, el equipo estaba a seis meses a un año de distancia de la liberación. Irónicamente, Wine siguió estando seis meses a un año de la liberación para la próxima década.

La comunicación inicial entre desarrolladores tomó muchas formas, incluyendo la lista de direcciones del kernel linux. La primera lista de direcciones de Wine fue hecha funcionar por Robert para permitir discusiones entre desarrolladores. Después de un año con muchos éxitos y cultivando el interés en el proyecto, éstos pidieron la creación del grupo de noticias: *comp.emulators.ms-windows.wine*. El voto fue aplastante a favor de su creación y se materializó en Usenet el 19 de julio de 1994. La lista de direcciones actual, donde se discute la mayor parte del desarrollo, fue creada por Doug Ridgway en octubre de 1998.

Los primeros años vieron muchos cambios en el desarrollo de Wine. Robert renunció en 1994 y Alexandre asumió el liderazgo del desarrollo. El *"windowing"* fue vuelto a escribir como llamadas de Xlib directas. Pero quizás lo más relevante fue que Microsoft comenzó a liberar el código de 32 bits y añadir la nueva funcionalidad a sus sistemas operativos. Ya no era bastante cargar sólo el código y ejecutarlo, fue necesario una integración más ajustada con los sistemas operativos subyacentes (principalmente Linux). Tuvieron que ser añadidos mecanismos que soportaran conexiones de red y archivos de registro. La arquitectura de Wine había confiado en un espacio de dirección compartido entre aplicaciones. Gradualmente se hizo claro que la separación de espacio de dirección era necesaria para aumentar la seguridad y soportar bibliotecas que tratan de tener acceso simultáneo al mismo espacio. El trabajo comenzado a principios de 2000 sigue hasta este día.

Alexandre recordó algunos hitos tempranos para Wine en una minuta que entregó en la primer *Wineconf*:

- Mayo de 1995: inicios del soporte para Win32
- Julio de 1995: cambio a autoconf
- Enero de 1996: corren Word y Excel
- Noviembre de 1997: Creación de sitio Web winehq.com

Los voluntarios comenzaron a contribuir a otros aspectos además de la programación. John Sheets y Susan Farley influyeron en la documentación original. Doug Ridgway estableció el sitio Web WineHQ en 1997. El sitio fue sostenido por Corel durante unos años, y luego CodeWeavers lo recibió de aquellos en marzo de 2002. Jeremy Newman es el webmaster.

"Wine Weekly News" apareció en el sitio Web en 1999. Editado al principio por Eric Pouech, Brian Vincent asumió en 2001. Durante los pocos años pasados, varios rasgos han sido añadidos al sitio Web. Una modernización a principios de 2003 añadió varias páginas para ayudar a nuevos usuarios a conocer Wine, proyectar listas para desarrolladores, e incorporó una larga lista de preguntas frecuentes.

En 1998 una decisión estratégica fue tomada por Corel para apoyar incondicionalmente Linux. Los elementos claves se centraron alrededor del suministro de un sistema basado en Linux que fuera tan simple de instalar como fácil de usar. A este fin, procuraron proveer una distribución

basada en Linux y soporte para a sus aplicaciones. La suite de Corel de programas de oficina exigió a Wine un nivel alto de sofisticación. Por primera vez en la historia de Wine el desarrollo comercial financiaba su desarrollo. Internamente, Corel tuvo dos equipos diferentes trabajando en Linux. Un grupo se concentró en el desarrollo de servidor, otro en la plataforma para aplicaciones. Corel mantuvo una gran relación con Wine, en parte debido a la participación de otra compañía, CodeWeavers.

La burbuja pronto se reventó. Los rumores acerca de que Corel discontinuaría su apoyo a Linux comenzaron a circular al final de 2000. Antes de principios de 2001 Corel oficialmente anunció que abandonaría su división Linux, específicamente el grupo que estaba trabajando en la distribución. Su apoyo al trabajo de Wine se terminó. La propiedad intelectual y algunos desarrolladores conformaron una nueva compañía Linux, Xandros.

No tardó mucho antes de que otros participaran para llenar el vacío. Para ese tiempo Alexandre había pasado a CodeWeavers haciendo la mayor parte del trabajo de nivel bajo de Wine. CodeWeavers se habían relacionado con Wine en 1999 y fueron contratados por Corel para mejorar partes de Wine que beneficiaría su compatibilidad con las aplicaciones de Corel. CodeWeavers comenzó a desarrollar sus propios productos y poner mucho empeño en Wine. Su propia versión de Wine incluyó instrumentos de dirección gráficos y un setup fácil. Varias distribuciones lo pusieron a disposición para download. Su primer producto, CrossOver Plug-in, permitió que usuarios de Linux ejecutaran plugins de Netscape diseñados para Windows. Las versiones más nuevas del producto han añadido soporte para más plugins. CrossOver Office fue liberado en marzo de 2002 para proporcionar soporte a aplicaciones de oficina como Excel y Lotus Notes.

TransGaming es una empresa que se formó en agosto de 2001. Gavriel State, quién había estado en Corel, la abandonó y formó su propia compañía. Los juegos de ordenador personal más nuevos habían estado enfocándose en las interfaces DirectX de Microsoft para todo, desde los dispositivos de entrada hasta la aceleración 3D. Atándolo a sus sistemas operativos, ésto hizo que migrar los juegos a plataformas diferentes resultara muy difícil. El soporte para DirectX en Wine fue añadido inicialmente por Marcus Meissner en 1997 y fue muy limitado. Gavriel procuró comercializar el desarrollo y crear una nueva versión de Wine diseñado para gamers. Fueron necesarias más tecnologías que sólo DirectX, y parte del trabajo inicial incluyó el apoyo a medidas de protección de copia. WineX 1.0 fue liberado en octubre de 2001 con el soporte para seis juegos.

También en 2001 otra compañía anunció intenciones de trabajar con Wine. Lindows.com quiso crear una distribución Linux simple de usar, que permitiera a los usuarios correr programas de Windows. No tardaron en abandonar la idea a favor de aplicaciones de Linux nativo. Antes de que esto sucediera patrocinaron *Wineconf* -un acontecimiento de tres días en marzo de 2002- que juntó a desarrolladores de todo el mundo. Para hacer asuntos más interesantes, en

vísperas de la conferencia la comunidad de Wine había emprendido otra discusión sobre licenciamiento.

La historia legendaria del licenciamiento de Wine ha provocado muchos debates. La cuestión de licenciamiento comprende dos áreas primarias: la licencia del código de Wine en sí mismo y la licencia de aplicaciones producidas usando Winelib. El objetivo de los desarrolladores de Wine es dar a los usuarios la capacidad de ponerlo en práctica como así también de aportar al proyecto Wine, de modo tal que esto no dificulte a otros de hacer lo mismo posteriormente. Al mismo tiempo se quiere dar a otros desarrolladores la posibilidad de virar su aplicación a babor sin el miedo de ser atados a una licencia de software que les impida hacer lo que ellos quieran con su creación.

Al principio, Wine adoptó una licencia de estilo BSD. Al final de 1999 la discusión comenzó sobre el cambio de la licencia. Richard Stallman opinó que esto era incompatible con GPL que potencialmente causa problemas con cualquier proyecto de código abierto que quisiera usar el código de Wine. La mayor parte de desarrolladores no vieron una necesidad de trabajar una nueva licencia y la licencia X11, un derivado de la licencia de BSD, tenía la mayor parte de apoyo. Se requirió un voto y en enero de 2000 Alexandre anunció que se haría la nueva licencia de Wine.

En marzo de 2002 una encuesta fue realizada tanto entre los desarrolladores libres como comerciales de Wine para ver si había interés en moverse a una licencia diferente. La mayor parte de desarrolladores no quisieron que su código fuera administrado por una entidad comercial y había preocupaciones sobre qué podría pasar. Después de mucho debate eligieron la licencia LGPL (*Lesser General Public Lissence*) y el 9 de marzo de 2002 el código fuente de Wine quedó atado a aquellos términos. La LGPL, a menudo referido como una licencia de "copyleft", requirió que los desarrolladores de Wine cumplieran con algunas pautas:

- El código fuente (incluso todos los cambios de las fuentes de Wine originales) debe ser puesto a disposición a la gente que recibe un binario de Wine. Esto también se aplica si Wine está usado como una biblioteca, en cuyo caso sólo la fuente de Wine (incluso todos los cambios) debe ser puesta a disposición.
- El simple enlace con Wine no significa que alguien tiene que poner disponible el código fuente de su programa.

El efecto inmediato fue la creación del proyecto ReWind para ampliar el código licenciado con X11. Muchos desarrolladores claves consintieron en permitir que sus adiciones estén usadas tanto por X11 como por código de Wine LGPL. La mayoría ha decidido enfocar sus esfuerzos en la sincronización con Wine licenciado bajo LGPL.

Poco después del cambio del desarrollo de licencia dio un salto mayor. Más patches

comenzaron a aparecer. Alexandre creó más CVS [3], y se reportaron más aplicaciones trabajando. Hacia el final de 2003 se alcanzó otro éxito con la separación de DLLs partiendo la combinación kernel32/ntdll. Estaban realizados la mayor parte de los cambios arquitectónicos para apoyar una liberación del beta. Otra conferencia de desarrolladores se realizó en enero de 2004 en San Pablo, Minnesota, patrocinado por CodeWeavers. Fue presentada una nueva hoja de ruta para tareas que tenían que ser completadas.

Wine ha crecido a más de 1.4 millones de líneas del código en lenguaje C durante la década pasada. Casi 700 personas han contribuido de algún modo. Como siempre, se puede esperar que Wine sea liberado algún día este año; o tal vez a principios de próximo año.

Cinco puntos claves:

Mucha gente en foros de discusión populares sigue pensando que Wine es "la última cosa que Linux necesita" o que no es “importante".
La página del proyecto presenta una lista de argumentos para confrontar con esas opiniones.

1. Variedad de suministro

Se considera universalmente que la diversificación de suministro es un aspecto importante en el manejo del riesgo.

El Ministerio de Justicia estadounidense ha relevado que el Microsoft Widows es ejecutado por más del 95 % de los ordenadores personales. Incluso tomando a Mac OS en cuenta, Microsoft Windows todavía está presente en más del 80 % de ordenadores. Esto también es posiblemente cierto en la mayor parte de otros países, no sólo en los EE.UU. Así los gobiernos, las compañías y los usuarios hogareños de todo el mundo dependen, en última instancia, de un solo proveedor: Microsoft.

La pregunta no es si Microsoft tiene malas intenciones, o si esto puede salirse de las normas del buen negocio: su plan compite con cualquier otro. Una compañía puede querer desprenderse de clientes pequeños para simplificar la administración y ahorrar el dinero en licencias de Windows por cliente. ¿Pero va Microsoft a hacer esto viable y discontinuar su mercado de Windows? ¿Dónde está la alternativa si Microsoft pone en práctica su modelo de suscripción de software? Si Microsoft no está interesado en satisfacer ese mercado, entonces no se tiene a ningún otro proveedor para reemplazarlo.

2. Las poblaciones grandes y homogéneas presentan mayor riesgo

Como fue mencionado anteriormente, Microsoft Windows es ejecutado en una proporción aplastante de ordenadores personales. Incluso teniendo en cuenta las diferentes versiones Windows, sobre todo Windows 9x y la familia de NT, esto representa una población homogénea grande. La mayor parte de los gobiernos, la mayor parte de los negocios, y muchos hogares dependen de él.

Los elementos de esta población, como todos otros sistemas complejos, no están milagrosamente exentos de vulnerabilidades. La infección de *“Code Red”* del verano 2001 debe recordarnos esto. El *“Code Red”* hizo lo que cualquier "virus" presente en una población homogénea grande haría: infectó más de 359.000 ordenadores, solamente el primer día. Por suerte, infectó a un miembro menos común de la familia de Windows y era completamente inocuo: no corrompió archivos al azar ni formateó discos duros.

Es sólo una cuestión de tiempo que un gusano más virulento aparezca. El único modo de disminuir su impacto es diversificar la población OS. Esta cuestión se considera ahora lo bastante seria para que los analistas de seguridad llamen a nuestra confianza en el Microsoft Widows una amenaza para la seguridad nacional.

Como es una realización alternativa de la API de Win32 y corre encima de OS completamente diferentes, Wine no tiene los mismos defectos y así puede proporcionar la diversidad necesaria.

3. Cualquier reemplazo de Windows debe ejecutar aplicaciones de Windows

La dependencia no está tanto en Microsoft Widows como lo está en las aplicaciones de Windows. Aplicaciones a medida, juegos, aplicaciones hogareñas, aplicaciones para mercados verticales, son lo que previene a usuarios, compañías y gobiernos de cambiar a otro sistema operativo. El 90 % de los usuarios toma en cuenta si su proveedor puede ofrecerles una suite de oficina, un cliente de correo electrónico, un navegador y un media-player. Pero todavía habrá un 10 % restante, con necesidades potencialmente críticas, que no son satisfechas. Lamentablemente este 10 % restante está repartido entre un amplio espectro de aplicaciones: miles de aplicaciones que ejecutan variedad desde juegos hasta software de contabilidad especializada para granjas francesas, enciclopedias italianas, software fiscal alemán, software de educación para niños, software bancario, años de desarrollo de software hogareño, etc.. Es la existencia de todo este software lo que hace el Windows tan irresistible y su monopolio tan fuerte. Ninguna plataforma se hará popular a menos que permita correr una parte significativa de aquel software y permita a individuos, compañías y gobiernos conservan sus inversiones en aquel software.

4. Escritorio: Problema del huevo y la gallina

Esto nos lleva al problema del escritorio de Linux. Hasta que Linux pueda proporcionar equivalentes a las susodichas aplicaciones, su porción en el mercado de las aplicaciones de escritorio se estancará. Pero hasta que no crezca el marketshare del escritorio Linux, ningún vendedor desarrollará aplicaciones para Linux. ¿Cómo romper este círculo vicioso?

Otra vez, Wine puede proporcionar una respuesta. Dejando a usuarios reutilizar las aplicaciones de Windows en las que han invertido tiempo y dinero, Wine baja drásticamente el umbral que impide a usuarios cambiar a Linux. Esto hace posible para Linux salir del estancamiento en el escritorio, aumentando su cuota de mercado en aquel segmento. Por otra parte, esto hace viable a las compañías producir versiones Linux de sus aplicaciones, y nuevos productos para salir sólo al mercado Linux.

Este razonamiento podría ser rechazado fácilmente si Wine sólo fuera capaz de correr el

Solitario. Sin embargo puede ejecutar Microsoft Office, aplicaciones multimedia, como Quick Time y Windows Media Player, y juegos como Max Payne ó Espora. Casi cualquier otra aplicación compleja puede ser corrida dedicándole un poco de tiempo. Y cada vez que el trabajo es hecho para añadir una aplicación a esta lista, mucha aplicaciones se benefician con este trabajo y pueden hacerse utilizables.

5. Ventajas de Wine

Último, pero no menos importante: Wine puede proporcionar ventajas sobre el Windows.

- Wine hace posible aprovechar todas las fortalezas Unix (estabilidad, flexibilidad, administración remota) mientras se usan las aplicaciones de Windows de las cuales se depende.
- Unix siempre ha hecho posible escribir potentes scripts. Wine hace posible llamar aplicaciones de Windows desde scrips que aprovechan el ambiente Unix al máximo.
- Wine hace posible tener acceso a aplicaciones de Windows de modo remoto aún estando a miles de millas.
- Wine es económico de usar para pequeños clientes: basta simplemente con instalar Wine en un servidor Linux, y voila, se puede tener acceso a estas aplicaciones de Windows desde cualquier terminal X.
- Wine puede ser usado también para poner aplicaciones de Windows existentes, a disposición en la Web usando VNC y su cliente de Java.
- Wine es un Software de Código Fuente Abierto, entonces se puede extender esto para satisfacer necesidades particulares o recurrir a una de las muchas compañías que lo hacen para sus clientes.

Rebatiendo Mitos sobre Wine

Según los desarrolladores de Wine estos son algunos de los mitos más comunes sobre Wine que son completamente incorrectos o no muy correctos:

Mito 1: "Wine es lento porque es un emulador"

Algunas personas quieren decir con esto que Wine debe emular cada instrucción de procesador de la aplicación de Windows. Esto es un claro error. Cuando el nombre de Wine dice: "Wine no es un emulador" quiere decir: Wine no emula el procesador de Intel x86. No será así tan lento como Wabi que, al no correr en un procesador de Intel x86, tiene que emular también el procesador. Las aplicaciones de Windows que no hacen llamadas de sistema correrán tan rápido como en Windows (ni más ni menos).

Algunas personas sostienen que, ya que Wine introduce una capa suplementaria encima del sistema, una aplicación de Windows correrá despacio. Es verdad que, en la teoría, las aplicaciones de Windows que corren en Wine o son compiladas de nuevo con Winelib no serán capaces de conseguir el mismo rendimiento que aplicaciones de un nativo Unix. Pero esto es la teoría. En la práctica se encontrará que una aplicación de Windows bien escrita puede batir a una aplicación de Unix mal escrita en cualquier momento. La eficacia de los algoritmos usados por la aplicación tendrá un mayor impacto en su rendimiento que Wine.

También, y esto es lo que interesa en general, la combinación Wine+Unix puede ser más eficiente que Windows. Como antes: tanto como sean buenos ó malos sus algoritmos. Ahora bien, para ser francos, el rendimiento no es todavía una prioridad de Wine. Hasta ahora es mucho más importante para el proyecto ampliar la cantidad de aplicaciones trabajando realmente en Wine.

Pero para aquellas aplicaciones que realmente trabajan -y desde un punto de vista puramente subjetivo- el rendimiento está bien. No hay ninguna pérdida obvia, excepto un poco de lentitud gráfica debido a código de Wine no optimizado y a pérdida de rendimiento de traducción del driver X11 (lo que puede ser un problema a veces).

Mito 2: "Wine es malo para Linux”

Hay un hecho indiscutible: existe una biblioteca de software enorme que trabaja con los sistemas operativos de Microsoft. Muchas de estas aplicaciones ya tienen equivalentes Linux. Sin embargo, para la mayor parte de personas allí permanece un puñado de programas que los mantiene atados a Windows. Algunos de estos programas no tienen casi ninguna posibilidad de ser migrados a Linux (p.ej Microsoft Office). Los otros simplemente no pueden ser migrados

porque han hecho abandonware (p.ej. TurboTax 1999). ¿Quién quería tener Windows sólo porque un día puedo tener que ejecutar un viejo programa fiscal?

El hecho que el Wine exista no exigirá a las compañías que migren su software, pero van a tener unos puntos menos de porcentaje de marketshare. Por otra parte Wine pone más software gratuito en las manos de personas que de otro modo no lo usarían. Por su parte, la historia ha mostrado repetidamente que marketshare más grande lleva a más desarrollo comercial. Más desarrollo comercial siempre ha llevado a más esfuerzos de desarrollar mejores equivalentes de software gratuito.

Mito 3: "los emuladores como VMware son mejores”

Seguro ellos son mejores.. si se está dispuesto a comprar una copia completa de un sistema operativo sólo para ejecutarlo bajo una máquina virtual. Para no mencionar que se tiene que comprar una copia de VMware para hacerlo trabajar.

Sin olvidar que el enorme rendimiento que demanda ejecutar una aplicación en un sistema operativo auténtico que corre encima de otro sistema operativo.

Sin embargo, se ha dicho que hay casos donde los emuladores son completamente útiles. Los desarrolladores pueden crear cajones de arena para ejecutar aplicaciones que correrán sobre otros sistemas operativos sin rebootear, etc. Pero tener un emulador auténtico sólo para ejecutar un procesador de textos probablemente no sea la mejor solución.

Mito 4: "se necesita el Windows de todos modos"

No. El objetivo de Wine es una realización nueva y completa de la API Windows que hará el Windows innecesario.

Actualmente se pueden ejecutar muchas aplicaciones sin instalar Windows. Pero esto se tiene que realizar que porque Wine está todavía lejano de su finalización. Muchas aplicaciones, en efecto, requerirán de Windows para un poco más de funcionalidad que Wine todavía no proporciona por sí mismo.

Mito 5: "Wine es malo, Winelib es mejor"

Parece que esto es un mito completamente popular, entre comillas. Básicamente algunas personas piensan que la marcha de una aplicación de Windows regular con el Wine es mucho menos confiable y alcanza un rendimiento muy inferior que una nueva compilación de esta misma aplicación con Winelib. Parece ser una variante del mito 'el Wine es lento porque es un

emulador'.

No hay realmente ninguna razón en esto. Los principiantes seguramente no observarán ninguna diferencia de rendimiento entre el Wine y Winelib para las pocas aplicaciones que se han probado en Winelib. Además se debe asumir que los errores de programación no están en el modo que el Wine maneja PE, es decir programas ejecutables en formato Win32. Los errores de programación y el rendimiento provienen igualmente de la realización de la API de Windows, y esto es compartido de todos modos.

Mito 6: "Wine siempre estará yendo detrás de Windows y no puede tener posibilidad de éxito en la puesta en marcha de nuevas aplicaciones"

La arquitectura de Wine agrega nuevos APIs (o DLLs) bastante fácil. Los desarrolladores de Wine rápidamente añaden funciones como sea necesario, cuando aplicaciones que requieren la última funcionalidad son reportadas, trabajando unos meses después de la liberación. Los ejemplos de esto incluyen el Office XP y Max Payne (un juego DirectX 8.0).

Además, Wine soporte DLLs nativos cuando las versiones incorporadas no funcionan correctamente. En muchos casos, es posible usar versiones DLL nativos para ganar el acceso al 100 % de las funciones que una aplicación requiere.

Mito 7: "Wine sólo pone en práctica un pequeño porcentaje de Windows API, y esto no es suficiente"

Las APIs se asemejan a una biblioteca -siempre es agradable tener tantos libros sobre los anaqueles como sea posible-, pero en realidad unos pocos libros escogidos son referidos repetidamente. La mayor parte de las aplicaciones son escritas al mínimo común denominador a fin de tener el auditorio más amplio posible. El soporte de XP de Windows simplemente no es importante: la mayor parte de aplicaciones requieren solamente Windows 95 o Windows 98. Actualmente Wine soporta más del 90 % de las llamadas encontradas en especificaciones de Windows populares, como el ECMA-234 y Open32. Wine todavía está añadiendo APIs de Win32, pero el mayor trabajo de la actualidad se concentra en fijar las funciones existentes y hacer cambios de arquitectura.

Mito 8: "Wine sólo es para el Windows 3.1 / Wine nunca soportará Win64"

Wine emprendió su viaje cuando Windows 95 aún no existía. Y aunque el Windows NT (y la API Win32) ya existía, sólo soportaba aplicaciones de Windows 3.1. De todos modos, casi nadie usó el Windows NT en aquel tiempo.

Pero desde esos días ha transcurrido mucho tiempo. En agosto de 2005, Wine anuncia su versión junto con el Windows 2000, y varios años después de esto siguió usádose Windows 98. Entonces realmente Win32 es el aspecto prioritario para el soporte de Wine. También el soporte para aplicaciones de Windows 3.1 todavía está girando, por supuesto, como sucede con el soporte a aplicaciones de DOS.

El soporte para Win64 permitiría que Wine corra trozos ejecutables de Windows nativo, pero Wine todavía no tiene este apoyo. Esto está bien, ya que hay muy pocas aplicaciones Win64 comercialmente disponibles. Una excepción, Torneo Irreal 2004, está disponible en una versión de Linux nativo, entonces nadie (excepto tal vez un hacker de Wine) debería querer ejecutar la versión para Win64 de todos modos.

Esto no significa que Wine no influirá en sistemas de 64 bits. Lo hace. Ver WineWiki para más información.

Mito 9: "Wine es sólo para Linux"

Ok, hasta ahora Wine no corre en muchas plataformas: sólo Linux, FreeBSD y Solaris. De todos modos, esto no es 'sólo Linux'.

También es verdad que la mayor parte de desarrolladores están influidos por Linux. Entonces se corre un riesgo más alto de tener una liberación específica de Wine que compila/trabaja en una plataforma no-Linux. Pero esto es por lo general corregido en la siguiente liberación. También es conocido que Wine pierde un poco de funcionalidad importante en no-Linux, buen multihilado por ejemplo. Estos problemas van siendo solucionados y Wine trabaja ajustadamente bien en cualquiera de las tres plataformas mencionadas anteriormente.

También hay un proyecto de compatibilidad Win32 para OS/2 (Odin), que hace el uso del código de Wine.

Mito 10: "Wine sólo es para Intel x86"

Bien, es verdad que Wine sólo corre en los procesadores x86 de Intel. Lamentablemente también se requerirá bastante trabajo antes de que corra en otras arquitecturas de procesador.

Pero qué entendemos por 'correr en un procesador no x86'.

Esto puede significar que 'puedo compilar una aplicación de Windows en Sparc, unirlo con Winelib, y hacerlo correr en Solaris'. Esto puede parecer muy restrictivo y aún sería muy útil: esto significaría migrar aplicaciones de Windows a casi cualquier arquitectura Unix. En cualquier caso esto es el primer paso para el permitir que Wine corra en otras arquitecturas de procesador. Lamentablemente el código de Wine no es muy portátil a otras arquitecturas de

procesador, en parte porque algunas de sus partes tienen que saber mucho sobre el procesador. Esto está siendo trabajado y en progreso, aunque despacio.

Entonces podríamos tomarlo para significar que 'Wine en Alfa debería ser capaz de ejecutar las aplicaciones Windows NT para Alfa. El requisito previo para esto es que Winelib compile sobre Alfa (o MIPS, la otra difunta plataforma de NT). Ahora, ¿esto sería realmente útil? No muchas personas tienen aplicaciones Windows NT para Alfa o procesador MIPS. Entonces probablemente esto no sería demasiado útil; tendría poca probabilidad de ocurrir ya que necesitaríamos a un programador que tiene sólo esta combinación de hardware y software para influir en ello.

Entonces hay lo que cada uno ha estado esperando. 'Quiero que Wine sea capaz de ejecutar mis aplicaciones de Windows x86 en cualquier arquitectura de procesador que me guste' es lo más complejo. Otra vez, el requisito previo es que Winelib trabaje sobre esta arquitectura, lo que definitivamente ocurrirá algún día. Entonces 'todo lo necesario' debe integrar un emulador x86 con el Wine (y también cambiar el nombre del Wine). Ulrich Weigand sólo hizo esto como un experimento hace tiempo cuando tenía 'algún tiempo libre'. Hasta consiguió que algunas aplicaciones Win16 corran. Su código no estaba en un estado que le permitiera ser integrado en Wine aún y no sabemos cuánto trabajo ha sido puesto en la persecución de él, aunque su tentativa realmente provocó muchas discusiones sobre la lista de direcciones de Wine. El resultado consiste en que necesitaríamos un emulador sofisticado que incluya un JIT a fin de conseguir algo realmente viable (es decir no demasiado lento). Y el desarrollo de tal emulador es un proyecto entero en sí mismo.
¿Significa esto que esto nunca pasará? No, seguro. Tal vez conseguiremos a algunos desarrolladores motivados una vez que los problemas Winelib sean solucionados. ¡Por supuesto, sucedería mucho más rápido si, por ejemplo, Compaq hiciera su emulador de Intel x86 con Código Abierto y financiara el desarrollo de Wine para sus máquinas Alfa. Como con todo proyecto de Código Abierto, si bastantes personas están interesadas y reúnen sus recursos juntos, sucederá.

Mito 11: "mi juego tiene una protección de copia que no trabaja con el Wine"

Actualmente los trabajos de SecuRom en Wine, y el soporte de objetos de directorio han sido añadidos para encaminarnos hacia la realización de un ntoskrnl.exe que soportará drivers de protección de copia como Safedisc. Una vez que el ntoskrnl.exe de Wine sea puesto en práctica tendrá soporte completo para las versiones 1 y 2 Safedisc.

Características de Wine

Compatibilidad Binaria

- Carga programas y bibliotecas de Windows 9x/NT/2000/XP, Windows 3.x y programas de DOS
- Disposición de memoria, manejo de excepciones, hilos y procesos compatibles con Win32
- Diseñado para sistemas operativos compatibles POSIX (eg. Linux y FreeBSD)
- Compatibilidad de errores “bug-fot-bug” con Windows

Gráfica

- La gráfica basada en X11 permite la remotización para a cualquier terminal X
- Fuentes X11, TrueType (.ttf/.ttc) y Windows Bitmap (.fon)
- Soporte DirectX para juegos (limitado soporte para Direct3D)
- Soporte para juegos y aplicaciones basados en OpenGL
- Impresión vía driver de PostScript, drivers nativos de impresora
- Driver para EMF y WMF
- "Escritorio en una caja" o ventanas combinables
- Soporte de capa para Windows MultiMedia (WinMM) con códecs incorporados

Permite que programas de Windows conecten con:

- Dispositivos de sonido vía ALSA, OSS, ARTES, GATO, libaudio, etc.
- Teclados multilingües y soporte para método de entrada CJK via XIM
- Módems y dispositivos serie
- Redes (TCP/IP e IPX)
- Scanners ASPI
- Windows Tablets vía XInput (eg. Wacom)

API de Wine API

- Diseñado para compatibilidad fuente y binaria con código de Win32
- Test suite para asegurar la compatibilidad con API de Win32
- Compilable en una amplia variedad de compiladores C
- Permite mezcla de permisos de código Win32 y POSIX
- Permite mezcla de permisos de ELFO (.so) y PE (.dll/.exe) binarios en un espacio de dirección
- Headers de archivo compatibles con Win32

- Documentación API generada automáticamente
- Compilador de recursos
- Compilador de mensajes
- Compilador de IDL
- Ectenso soporte para Unicode
- Internacionalización. Wine soporta 16 lenguas
- Depuración incorporada y mensajes de rastreo configurables
- Soporte de inspección de memoria externa usando
- Programas de muestra

Referencias

(1) POSIX(**P**ortable **O**perating **S**ystem **I**nterface para UNI**X**). Familia de estándares relacionados, especificados por la IEEE para definir APIs con el objetivo de compatibilidad de software entre diferentes sistemas operativos Unix. El término "POSIX" fue sugerido por Richard Stallman en respuesta a un requerimiento de la IEEE. La familia de estándares POSIX es formalmente designada como IEEE 1003, y el nombre del estándar internacional ISO es ISO/IEC 9945. Si los diseñadores de programas se adecúan a POSIX, sus aplicaciones podrán ejecutarse en cualquier sistema operativo compatible con POSIX.

Los sistemas operativos que soportan POSIX son: A/UX, AIX, BSD/OS, HP-UX, INTEGRITY, Irix, LynxOS, Mac OS X, MINIX, OpenVMS, QNX, RTEMS (POSIX 1003.1-2003 Profile 52), Solaris, OpenSolaris, UnixWare, VxWorks, Windows con kernel NT (usados en Windows NT, 2000, 2003; XP, Vista): sólo en algunas ediciones o con determinadas aplicaciones instaladas.

(2) Emulador es un software que permite ejecutar programas de computadora o videojuegos en una plataforma (arquitectura hardware) diferente de aquella para la cual fueron escritos originalmente. A diferencia de un simulador, que sólo trata de reproducir el comportamiento del programa, un emulador trata de modelar de forma completa y precisa el dispositivo que se está emulando.

(3) CVS(Concurrent Versionning System): aplicación informática que implementa un sistema de control de versiones (mantiene un registro de todo el trabajo y de los cambios en archivos) que conforman un proyecto de software. Permite la colaboración de varios desarrolladores independientemente de la distancia a que se encuentren.

PARTE 2: EXPERIENCIAS DE INSTALACIÓN

Instalando Wine sobre Debian Linux Lenny

Es sabido que las distribuciones Debian utilizan APT como herramientas de administración de paquetes. APT está capacitado para la instalación automática de paquetes de software incluyendo sus dependencias necesarias y su actualización, a condición de conocer los repositorios de software (los que se declaran en el archivo /etc/apt/sources.list).

La última versión estable de Wine a la fecha de este trabajo es la 1.0.1 y su repositorio contiene paquetes para arquitecturas i386 y amd64.

Para la instalación del paquete Wine pueden usarse dos métodos: agregar el repositorio (en sources.list) ó descargar un paquete .deb e instalarlo manualmente.

En cualquier caso, primeramente debe removerse las versiones antiguas de Wine. Es prioritario realizar esta operación para cancelar cualquier posiblilidad de conflictos con versiones de wine instaladas anteriormente.

Procedimiento:

Remover versión antigua

* *Abrir un terminal*

$ su #para loguearse como root

$ apt-get remove libwine wine #para remover versiones anteriores

Método 1: Agregar el repositorio

* *Abrir un terminal*

$ su #para loguearse como root

$ gedit /etc/apt/sources.list #para abrir el archivo de repositorios con el editor de texto (puede hacerse con otro editor)

* Para Debian Lenny, agregar esta línea: *deb http://www.lamaresh.net/apt lenny main*

* G*uardar y cerrar sources.list*

$ wget -O - http://www.lamaresh.net/apt/key.gpg | apt-key add - #para agregar gpg key

$ apt-get update #para actualizar la lista de paquetes

$ apt-get install wine #para instalar la última versión de Wine

Método 2: Descargar .deb e instalarlo manualmente

Elegir un paquete .deb desde la lista de binarios y guardarla en el directorio /home.

* *Abrir un terminal*

$ cd ~ #para ir al directorio home

$ su #para loguearnos como root

$ dpkg -i wine_1.1.xxx.deb #cambiar xxx por el paquete elegido. Wine es instalado.

Verificación de la instalación.

Habiendo finalizado la instalación, Wine habrá creado en nuestro directorio home una estructura de directorios semejante a la de Windows.

Esto puede verificarse haciendo:

$ su #para loguearnos como root

$ cd /home/usuarioX/.wine # cambio al directorio de wine (usuarioX es el que estemos usando)

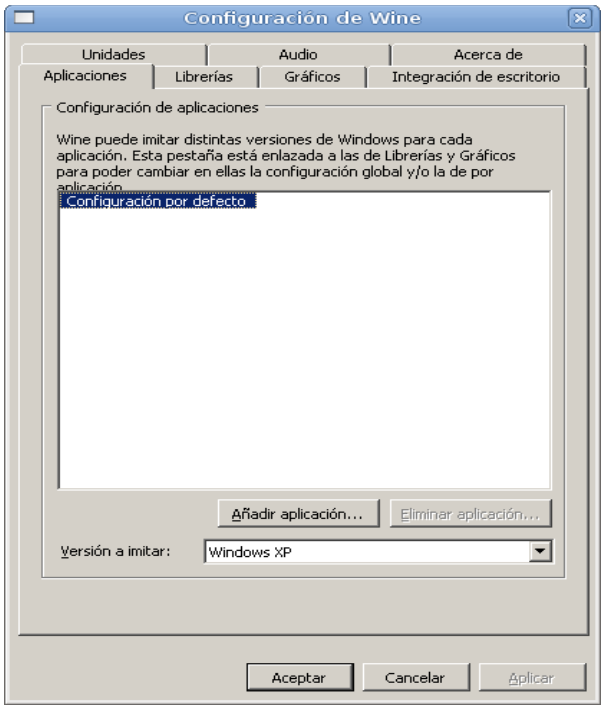
$ ls drive_c -l # para ver la estructura de archivos de"Windows"

Wine en el escritorio

Al terminar la instalación, Wine aparecerá en el menú del escritorio. Dos nuevos lanzadores se agregan al menú de "Herramientas del Sistema":

"wine Configuration" (figura) que abre un completo panel de configuración para agregar aplicaciones y librerías, configurar Audio, Gráficos y Escritorio y

"wine Uninstaller", que es la herramienta para desinstalar aplicaciones previamente instaladas con Wine.



PARTE 3: ENSAYO DE APLICACIONES SOBRE WINE EN LINUX DEBIAN LENNY

A) APLICACIONES GRATUITAS MIGRADAS DESDE WINDOWS

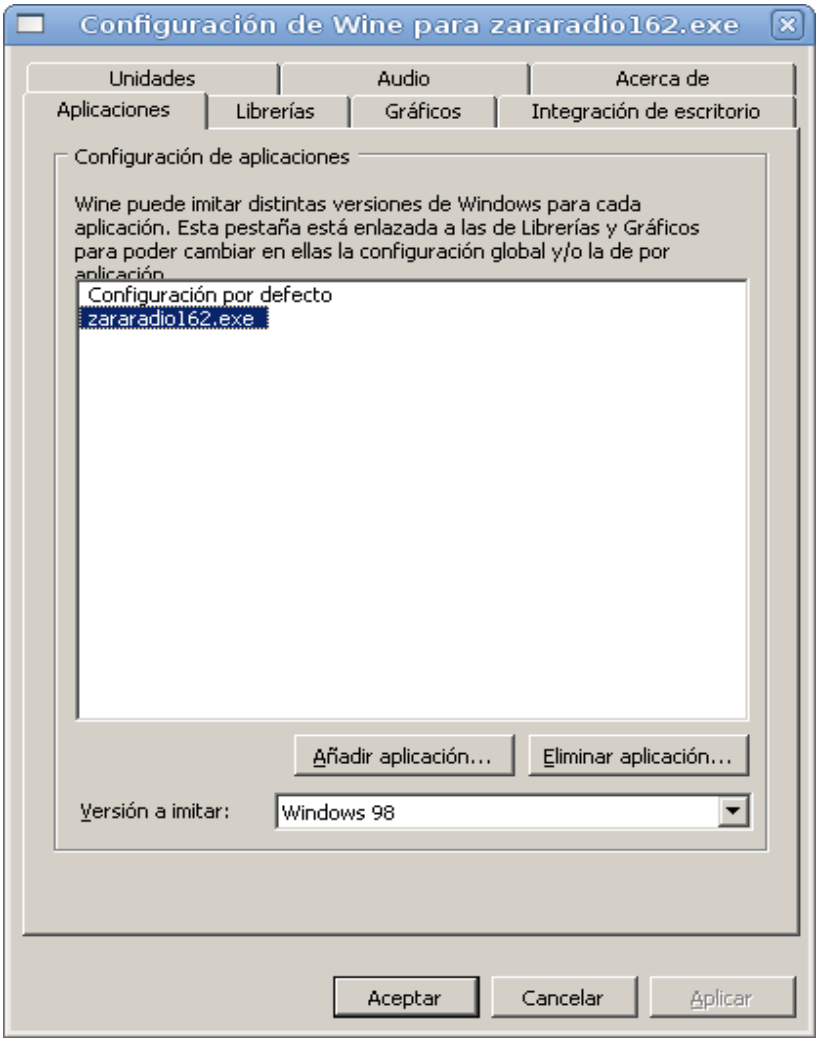
A.1- Instalación del software *ZARARADIO*

Zaradio es una aplicación de uso libre para la automatización de emisoras de radio. Es básica pero adecuada para cubrir los requerimientos esenciales de una aplicación de ese tipo.

La instalación sobre Debian Linux Lenny se realizó desde el escritorio siguiendo el siguiente *Procedimiento:*

* Apertura del panel de configuración *WineConfig* desde el lanzador “wine Configuration”.

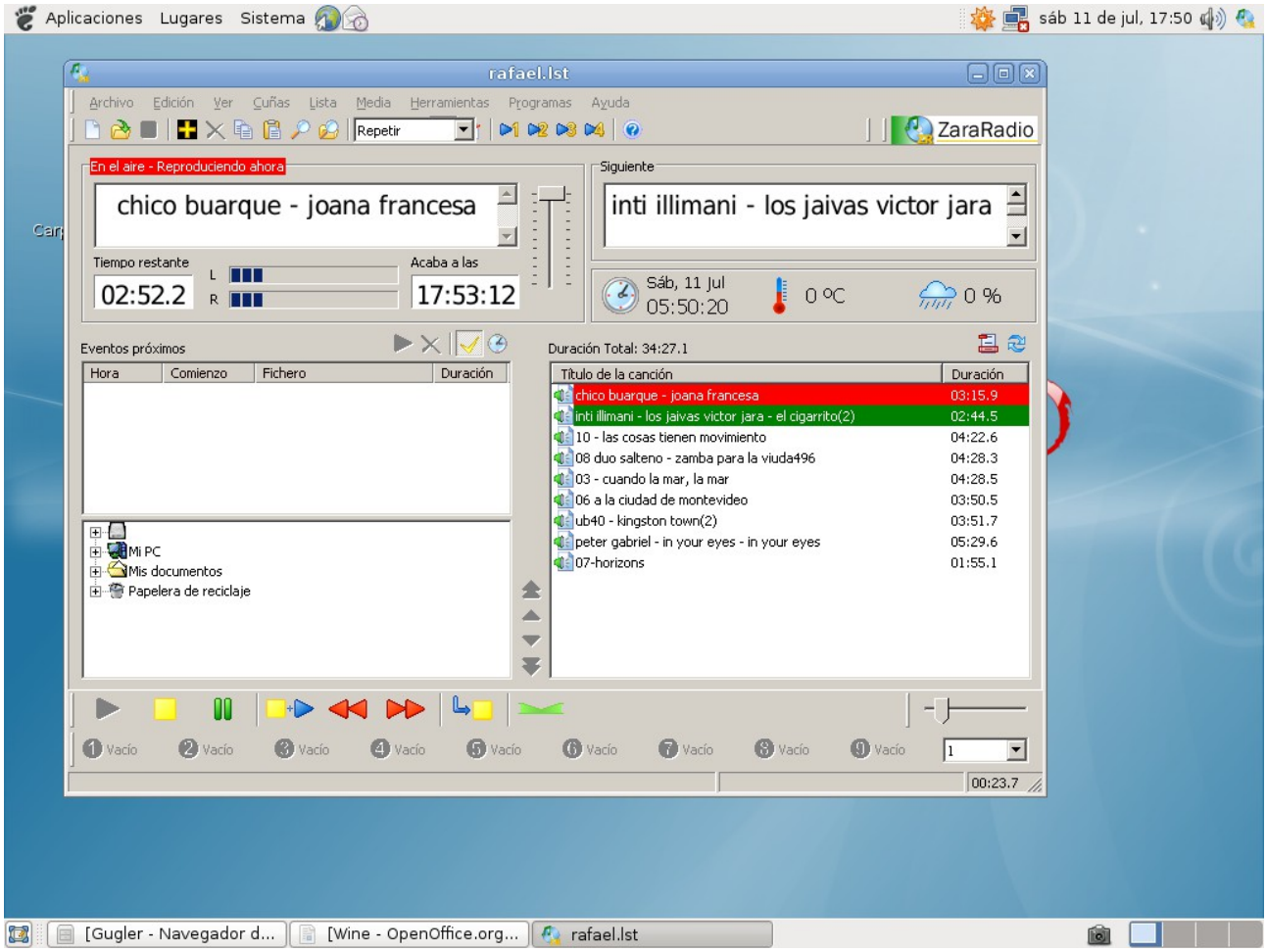
* Inclusión de la aplicación *-zararadio162.exe-* en el panel; luego “Aceptar”:



* Ejecución del ejecutable *zararadio162.exe* desde su ubicación -en nuestro caso */home/usuario*. La instalación ocurrió tal como en Windows.

* Realizada la instalación aparece en el menú de aplicaciones el conjunto '*Wine*' que incluye un lanzador para nuestro programa '*Zararadio*'.

- En una primera prueba básica, Zararadio1.6 se ejecutó con éxito.



A.2- Instalación del software *ECHOLINK*

EchoLink es una aplicación basada en voice-IP utilizada por radioaficionados. Esta aplicación permite -entre otros muchos usos- interconectar distintas radio-estaciones a través de internet y así lograr la comunicación a grandes distancias solo utilizando equipos de radio para corta distancia. Este software es de libre uso sólo para radioaficionados que posean licencia para operar una estación de radio. Dado que no existe una versión nativa de este software para Linux se requiere de wine para poder correrlo en Linux.

Existen dos maneras de instalar el soft: una es si poseemos una instalación de windows en nuestra PC y otra es instalar directamente en nuestro Linux.

Utilizamos la segunda opción. Para ello seguimos los siguientes pasos :

1- Descargamos el binario para instalar *EchoLinkSetup_1_6_848.exe*

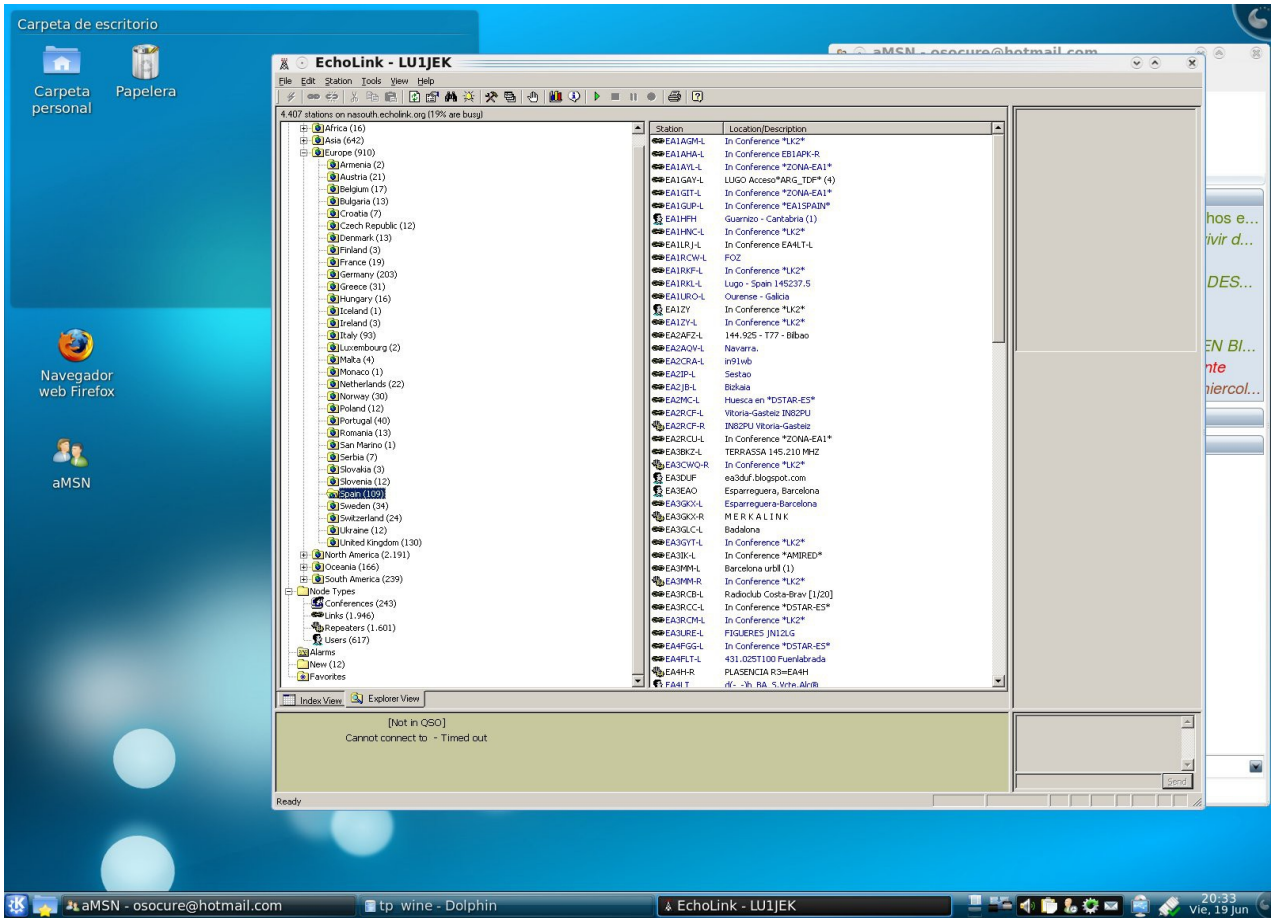
2- Ejecutamos la siguiente orden: *$ wine EchoLinkSetup_1_6_848.exe*

La instalación del soft se inicia tal como en Windows. Seguimos los pasos que nos pide el instalador; le damos como carpeta de instalación '*C:\Archivos de programa\K1FRD\Echolink*' y seguimos con la instalación hasta el final.

Terminada la instalación tendremos agregada la siguiente estructura de archivos en nuestra carpeta home: *.wine/drive_c/Archivos de programa/K1RFD/EchoLink*

En esta ultima carpeta se encuentra el ejecutable *EchoLink.exe*, para hacerlo correr ejecutamos el siguiente comando:

$ wine EchoLink.exe



B) ENSAYO DE DESARROLLO A MEDIDA: APLICACIÓN VISUAL BASIC 4.0

B.1- Instalación del software CONTROL MEGAPREMIOS

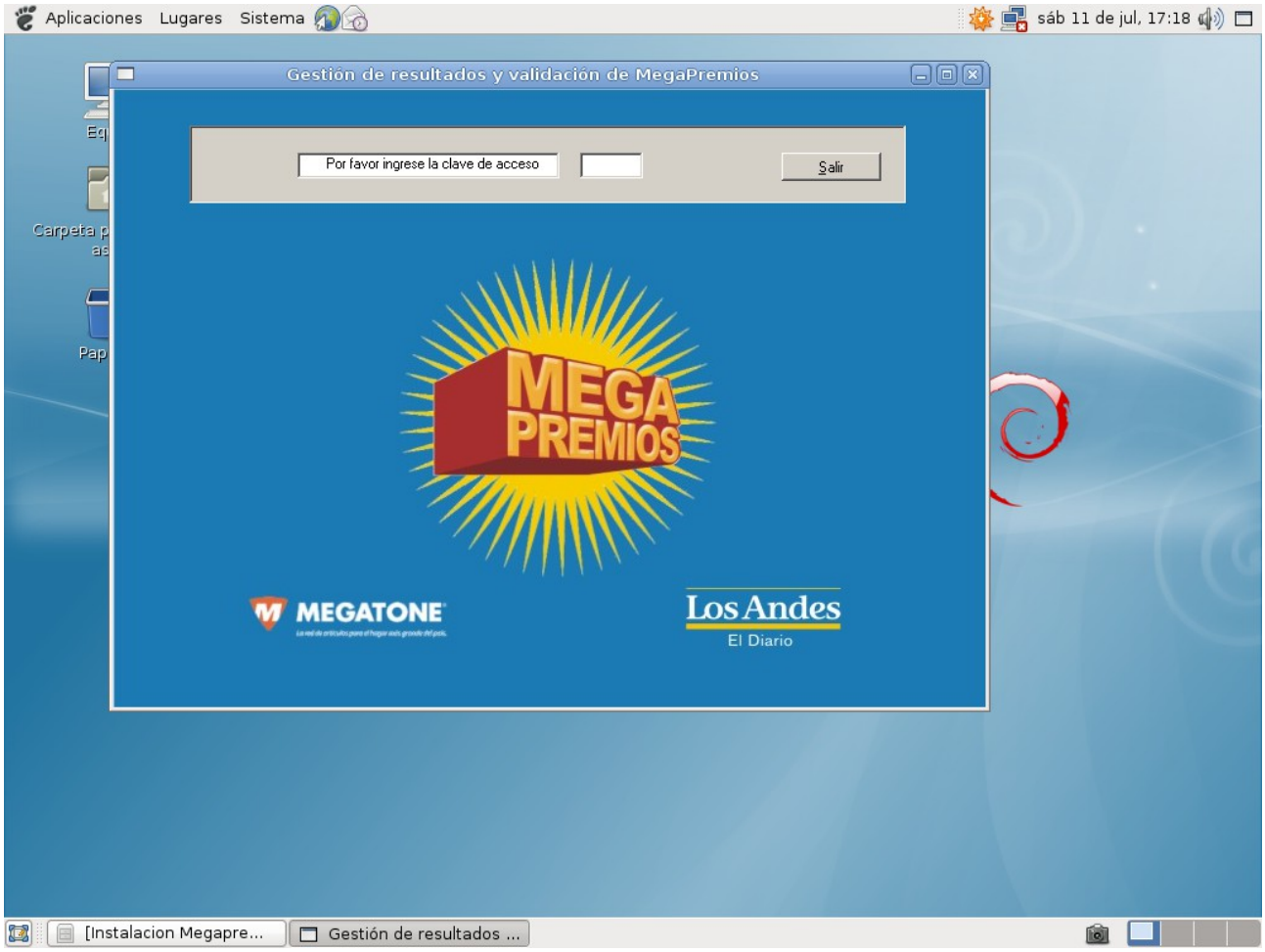
La aplicación ensayada es un desarrollo sencillo para 32 bits realizado en Visual Basic 4.0.

La instalación sobre Debian Linux Lenny se realizó desde el escritorio siguiendo el siguiente *Procedimiento:*

* Ejecución del ejecutable *setup.exe* -correspondiente al pack de instalación del programa '*Control Megapremios*'- desde su ubicación, en nuestro caso */home/usuario*. La instalación ocurrió tal como en Windows.

* Realizada la instalación fue necesario incluir la aplicación en el escritorio mediante un lanzador al ejecutable que se realizó editando el menú 'Aplicaciones'.

* En una primera prueba básica, '*Control Megapremios*' se ejecutó con éxito.



ALGUNAS CONCLUSIONES

- Entre la documentación consultada (ver apartado final) aparecen también variadas críticas a la funcionalidad de Wine. Varias de estas críticas están emparentadas a las que habitualmente se hacen a Linux: “hay que ser programador para usarlo”, “demasiado trabajo para implementarlo”, “programa en estado alfa (ni siquiera beta)”, etc.. etc..
- En nuestro caso, si bien las pruebas realizadas no fueron exhaustivas, Wine se ofrece como una solución interesante al problema planteado inicialmente; esto es, casos con requerimiento de ejecutar sobre Linux aplicaciones desarrolladas para plataforma Windows.

DOCUMENTACION DE INTERES

Sitio oficial:

http://www.winehq.org/

Documentos:

http://www.linux-magazine.es/issue/07

http://es.wikipedia.org/wiki/Wine

http://von-thadden.de/Joachim/WineTools/

http://www.wired.com/news/business/0,1367,49719,00.html

Forums:
http://www.linuxforums.org/forum/wine/