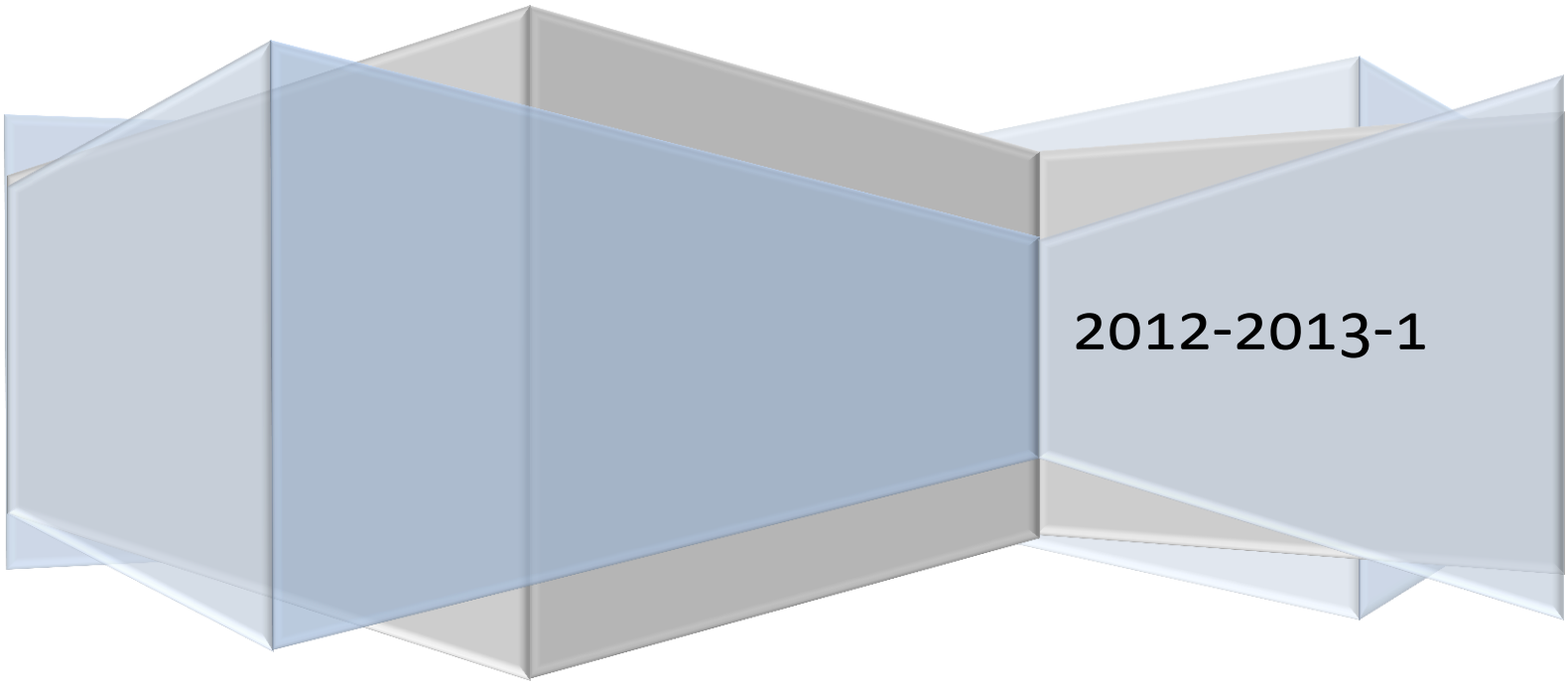
UOC Universitat Oberta de Catalunya

Estado del arte de los gestores de ventanas en GNU/Linux

TFC – GNU/Linux

Raúl Gómez Sánchez



Índice

Tabla de ilustraciones

1 Introducción a los gestores de ventanas

Un **gestor de ventanas** es un programa informático que controla la ubicación y apariencia de las ventanas bajo un sistema de ventanas en una interfaz gráfica de usuario.

Originalmente, cada aplicación diseñaba y codificaba su propio interfaz de usuario y sus componentes. Después, fueron desarrolladas librerías con subrutinas para simplificar la escritura de la parte de los programas destinada al usuario de interfaz. Finalmente, se produjeron kits de herramientas que fueron ampliamente distribuidas y que permitieron a algunas de ellas convertirse en estándar de facto.

Las acciones asociadas al gestor de ventanas suelen ser, abrir, cerrar, minimizar, maximizar, mover, escalar y mantener un listado de las ventanas abiertas. Es también muy común que el gestor de ventanas integre elementos como: el decorador de ventanas, un panel, un visor de escritorios virtuales, iconos y un tapiz, por lo que hablaremos a continuación de las metáforas del escritorio y el paradigma WIMP, para explicar el sistema de relación entre el usuario y el computador.

Las plataformas Windows y Mac OS X ofrecen un gestor de ventanas estandarizado por sus vendedores e integrado en el propio sistema operativo. En cambio el sistema gráfico X Window, popular en el ámbito GNU/Linux, permite al usuario escoger entre varios gestores. Los gestores de ventanas difieren entre sí de muchas maneras, incluyendo apariencia, consumo de memoria, opciones de personalización, escritorios múltiples o virtuales y similitud con ciertos entornos de escritorio ya existentes, entre otras.

En los entornos de escritorio, éstos traen integrados un gestor de ventanas. Para KDE está KWin, para GNOME, Metacity, para Xfce está Xfwm, y en LXDE es usado Openbox. Otros gestores de ventanas son Fluxbox, IdeWM y Compiz, este último es muy popular por los efectos gráficos que posee. Estos son totalmente intercambiables, puede usarse por ejemplo GNOME con Compiz, o KDE con Openbox.

Un gestor de ventanas no es otra cosa que el conjunto de programas, ventanas y funcionalidades que hacen posible que el usuario pueda interactuar con el sistema de forma gráfica y no en modo texto.

Para usar un gestor de ventanas, hay que tener configurado un servidor X. También hay que decir que el gestor de ventanas utilizado es totalmente independiente del servidor X utilizado.

A diferencia de otros sistemas operativos, en GNU/Linux no es necesario utilizar un conjunto servidor X - gestor de ventanas para usar el sistema. El sistema operativo y el conjunto servidor X - gestor de ventanas usado, son cosas totalmente distintas e independientes entre sí. De hecho, existen usuarios que trabajan en modo texto sin ningún problema y sin usar un interfaz gráfico.

Existen numerosos y variados gestores de ventanas para Linux, unos mejores y otros más desarrollados y estables. Es el usuario el que tiene que decidir que gestor satisface mejor sus necesidades, pudiendo incluso tener más de uno instalado. Para explicarlo de un modo más sencillo, podríamos decir que, si un ordenador es usado por varios usuarios, todos utilizarán el mismo servidor X pero no necesariamente el mismo gestor de ventanas.

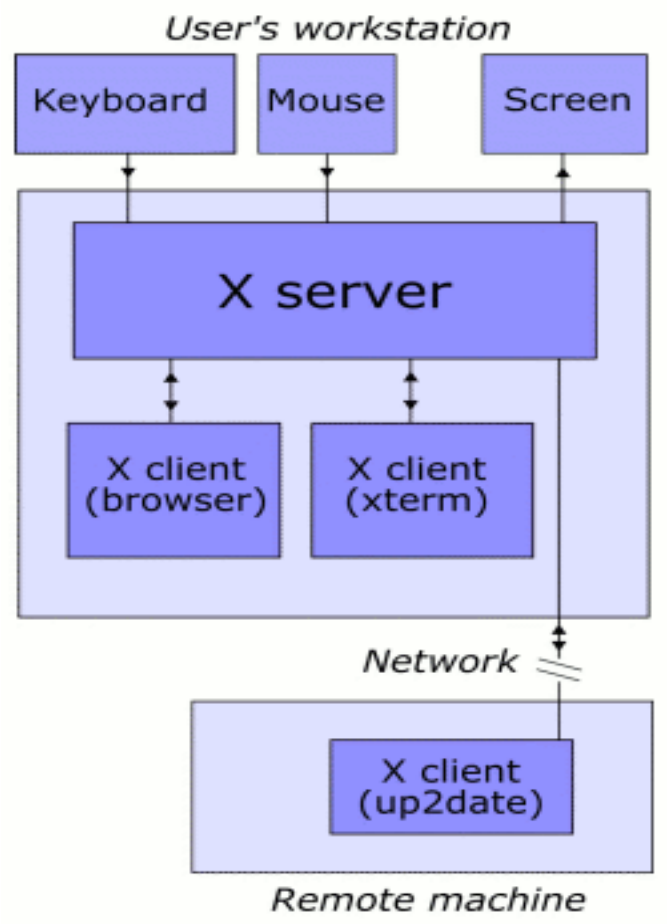


Figura 1. Esquema cliente-servidor del gestor X

Es habitual que los conceptos de entorno de escritorio y gestor de ventanas sean confundidos o incluso usados indistintamente, por lo que es necesario remarcar en este estudio que un entorno de escritorio funciona uniendo diferentes clientes de X, de manera que se cree un ambiente de usuario común al usarlos conjuntamente. De esta manera, los entornos de escritorio permiten a los clientes de X y a las aplicaciones que se ejecutan, comunicarse entre ellas permitiendo la realización de tareas avanzadas.

Sin embargo, los gestores de ventanas son programas clientes de X que forman parte del entorno de escritorio (como el gestor de ventanas Kwin forma parte del entorno de escritorio KDE, o el gestor de ventanas Xfwm forma parte del entorno de escritorio Xfce), o que funcionan de modo independiente (como

el gestor de ventanas Fluxbox) pero cuyo propósito es controlar la forma en que se muestran las ventanas y sus funcionalidades.

1.1 Metáfora del escritorio

La metáfora del escritorio es una metáfora de interfaz que se compone de una serie de conceptos usados por los usuarios de interfaces gráficos para ayudarlos a interactuar más fácilmente con los equipos. Esta metáfora trata al monitor como si fuese el escritorio del usuario, en el que pueden ser ubicados objetos como documentos y carpetas de documentos. Un documento puede ser abierto dentro de una ventana, que representa una copia de papel del documento que se encuentra en el escritorio. También existen pequeñas aplicaciones llamadas accesorios de escritorio, como por ejemplo calculadoras o bloc de notas, etc.

La primera aplicación conocida de la metáfora del escritorio fue llevada a cabo por Apple en los años 70, cuando miembros de un grupo de investigación del proyecto Apple Lisa usaron el término escritorio para definir una nueva interfaz gráfica de usuario entre usuarios y computadores. En paralelo, Xerox presentó el Xerox Star, que sería el primer ordenador comercial basado en esta metáfora.

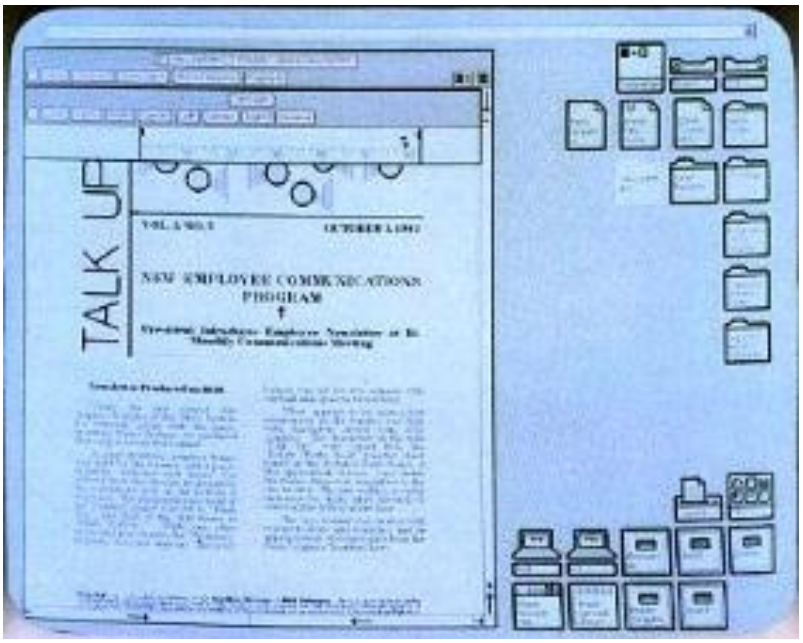


Figura 2. Escritorio del Xerox Star

Tras la popularización del Apple Macintosh en 1984, su uso se estandarizó en los sistemas operativos hasta nuestros días, con pequeñas diferencias en el enfoque según el fabricante (por ejemplo, en GNU/Linux los discos duros aparecen ubicados en el escritorio por defecto, mientras que en Windows su acceso se obtiene mediante un icono llamado “Equipo”).

La metáfora del escritorio ha sido implementada por numerosos entornos de escritorio, en los que el acceso a nuevas características y la usabilidad del equipo son generalmente más importantes que el mantenimiento de la esencia de la metáfora, dando lugar a barras de tareas o volúmenes de disco, que no tienen una representación en un escritorio real.

1.2 WIMP

WIMP es el acrónimo en inglés de “Windows, icons, menus, pointer” (“Ventanas, iconos, menús, punteros”), y es un paradigma de interacción entre personas y computadores que usa a estos elementos como base de su interfaz de usuario, también es habitual sustituir ratón (“mouse”), o menú desplegable (“pull down menu”) para menús y puntero respectivamente.

Fue Xerox, quién en 1972 desarrolló el primer sistema operativo con un paradigma WIMP para el Xerox Alto, como una evolución a los CLI (Command line interface) que existían hasta aquel momento y que centraban la interacción entre humano y computador a la línea de comandos.

Su popularización se produjo cuando Apple presentó su Macintosh en 1984, el cual añadió los conceptos de barra de menú y gestor de ventana extendido. Desde entonces ha ido difudiéndose hasta situarse actualmente como el paradigma principal de la interacción entre humanos y computadores.

En un sistema WIMP una ventana ejecuta un programa autocontenido, separado de otros programas que se puedan estar ejecutando al mismo tiempo en otras ventanas, un icono actúa como un atajo a una acción que realiza el equipo, un menú es un texto o una selección basada en un icono que selecciona y ejecuta programas o tareas y un puntero es un símbolo que se encuentra en la pantalla y que representa el movimiento de un dispositivo físico que el usuario controla para seleccionar iconos u otros elementos.

Es habitual confundir o usar sinónimos WIMP y GUI (Graphical User Interface), pero en realidad no lo son. Los sistemas WIMP derivan de los GUIs ya que cualquier interfaz que utiliza gráficos se puede llamar GUI, pero no al revés, ya que no todos los GUIs deben utilizar los gráficos como elemento principal (por ejemplo el uso de iconos y punteros).

El éxito de WIMP es entendible debido a que es la mejor opción para representar entornos rectangulares en una pantalla plana de dos dimensiones, llevando al máximo las posibilidades de abstraer documentos o espacios de trabajo completos y facilitando a su vez la introducción de este sistema a usuario noveles.

Aunque actualmente se intenta superar este paradigma, debido principalmente a la introducción de nuevos dispositivos de interfaz que incorporan pantallas táctiles y que pueden ser combinadas con diferentes metáforas visuales, el paradigma WIMP sigue vigente para la gran mayoría de ordenadores personales y dispositivos que podemos encontrar en el mercado.

No es posible obviar sin embargo, que este paradigma que lleva ya vigente 40 años en un mundo tan dinámico como el de la informática, da signos de agotamiento, y que el modo diferente en el que estamos comenzando a interactuar con los ordenadores puede llevarnos a que sea necesario formular un nuevo paradigma de interacción, que nos ayude a relacionarnos de un modo más certero con las nuevas posibilidades que nos empiezan a brindar los dispositivos actuales.

Sirva como ejemplo los llamados “Natural User Interface” (NUI), que ya están siendo presentados como la siguiente evolución del paradigma de interacción y que cuenta como característica principal la no inclusión de una interfaz gráfica visible (o que se vuelva invisible basándose en aprender las interacciones que se llevan a cabo).

Figura 3. Evolución de los paradigmas

1.3 Tipos de gestores

Los gestores de ventanas pueden ser divididos en tres clases según la forma en la que manejan las ventanas. Esta categorización no es estrictamente excluyente, ya que hay gestores de ventanas que implementan características de varias categorías.

1.3.1 Gestores de ventanas de composición

Los gestores de composición de ventanas permiten que todas las ventanas se creen y dibujen de forma separada y después sean combinadas y dibujadas en varios entornos 2D y 3D. Esto permite una gran variedad de estilos de interfaz y la presencia de efectos visuales 2D y 3D. Mac OS X fue el primer sistema operativo empaquetado con un gestor de composición de ventanas, al cual siguieron distribuciones de GNU/Linux gracias a Compiz y más tarde Windows Vista.

En los gestores de composición se efectúa un proceso de dos pasos: primero las ventanas se dibujan en buffers individuales y posteriormente sus imágenes se componen dentro del buffer de pantalla. Debido a este diseño, este tipo de gestores son los que más recursos consumen, pero pueden ser aplicados diferentes efectos visuales como sombras o transparencias.

Estos gestores de ventana evitan los problemas que pueden darse en gestores de ventana de pila, los cuales pueden sufrir incidencias durante el proceso de renderizado que lleven a una ventana a dejar de responder o a quedarse en blanco. En el caso de la composición, aunque exista un problema con el último renderizado solicitado a una ventana, ésta mantendrá su última versión dibujada en pantalla e incluso podrá mostrar mensajes que indiquen que la ventana se ha vuelto inestable.

Los efectos característicos de los gestores de composición en GNU/Linux se logran mediante el uso de cambios en la librería X11 para permitir el uso de aceleración 3D. La aparición de AIGLX (Accelerated Indirect GLX) ha permitido el uso de estas capacidades de renderizado a los clientes X, permitiendo aceleración sobre el protocolo GLX y mediante el uso de OpenGL.

Dentro de GNU/Linux, algunos gestores de ventanas de composición son: Beryl, Compiz, KWin, Metacity, Mutter o Xfwm.

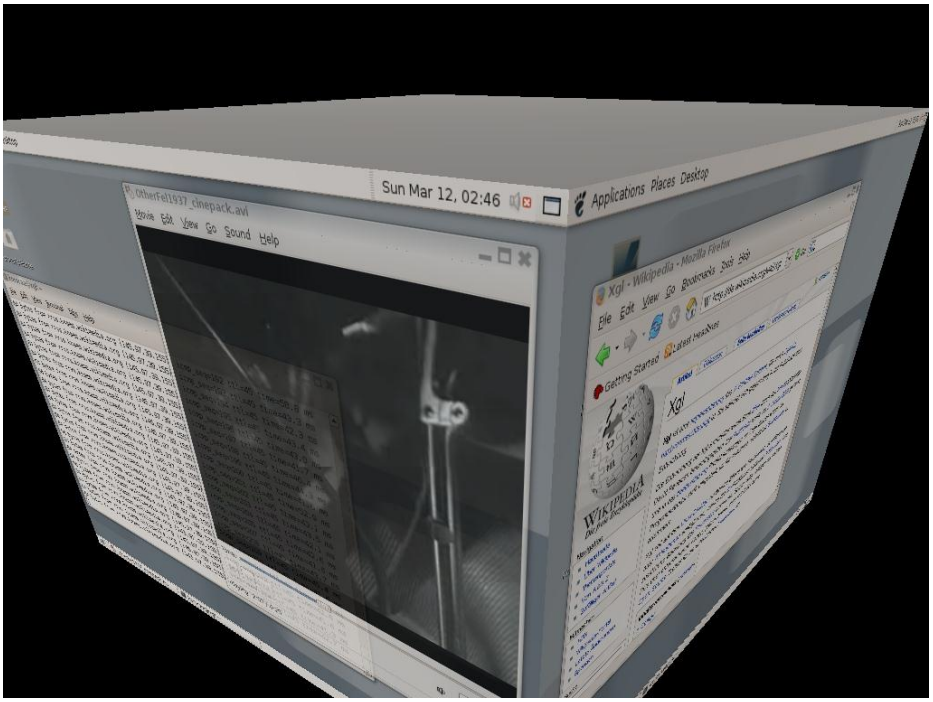


Figura 4. Gestor de ventana de composición con efecto cubo

1.3.2 Gestores de ventanas de pila

Todos los gestores de ventanas que tienen ventanas superpuestas y no gestionan composición son gestores de ventana de pila o stacked window managers, aunque no todos utilizan las mismas metodologías. Estos gestores permiten que varias ventanas se solapen dibujando primero las ventanas que están más al fondo.

Estas ventanas se dibujan de manera individual en las coordenadas especificadas, de manera que si el área de una ventana queda por encima de otra, la ventana que se encuentre por encima sobreescribe la parte visible de la otra, por lo que esta apariencia resulta familiar a muchos usuarios, ya que las ventanas actúan de manera parecida a trozos de papel en un escritorio, donde pueden ser movidos y pueden solaparse.

Al contrario que los gestores de ventanas de composición, la ausencia de buffers separados de pantalla puede hacer que su eficiencia sea mayor, pero por otro lado no pueden realizar efectos como las transparencias.

Dentro de GNU/Linux, algunos gestores de ventanas de pila son: Blackbox, Enlightenement, Fluxbox, FVWM, IceWM o MWM.

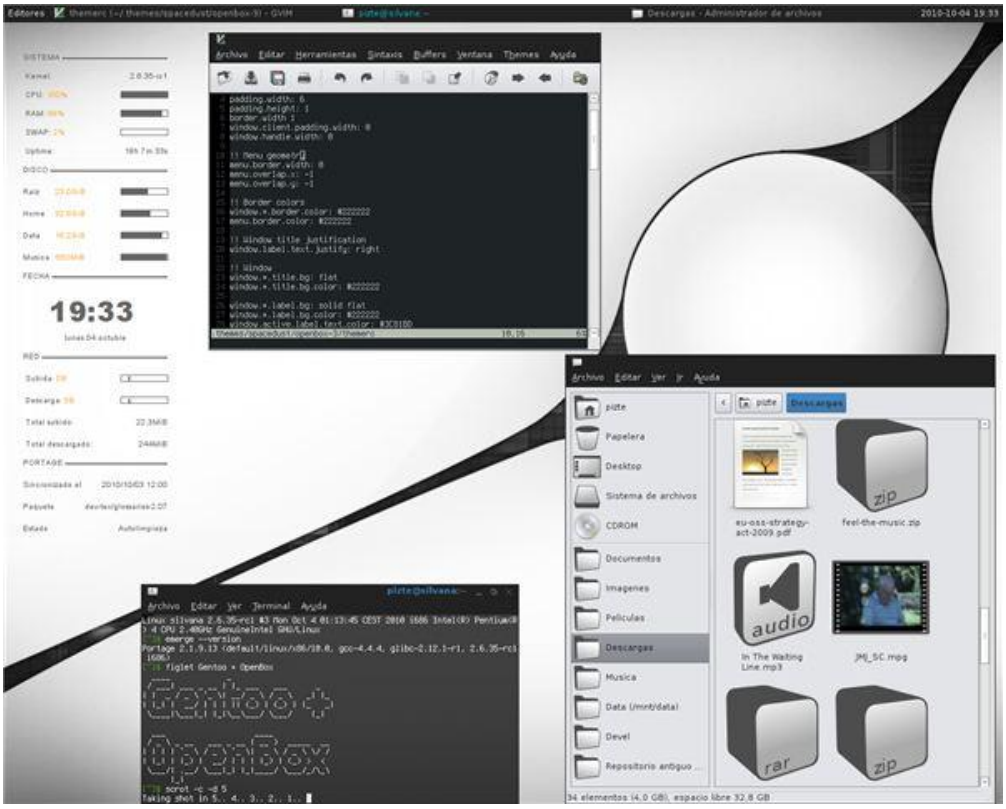


Figura 5. Gestor de ventana de pila

1.3.3 Gestores de ventanas de mosaico

Los gestores de ventana de mosaico o tiling window managers posicionan todas las ventanas en la pantalla sin solaparse, Microsoft Windows 1.0 usa este tipo de posicionamiento y en la actualidad existen muchos gestores de este tipo para X Window.

Dentro de GNU/Linux, algunos gestores de ventana de mosaico son: Awesome, Dwm, Larswm, Stumpwm o Xmonad.

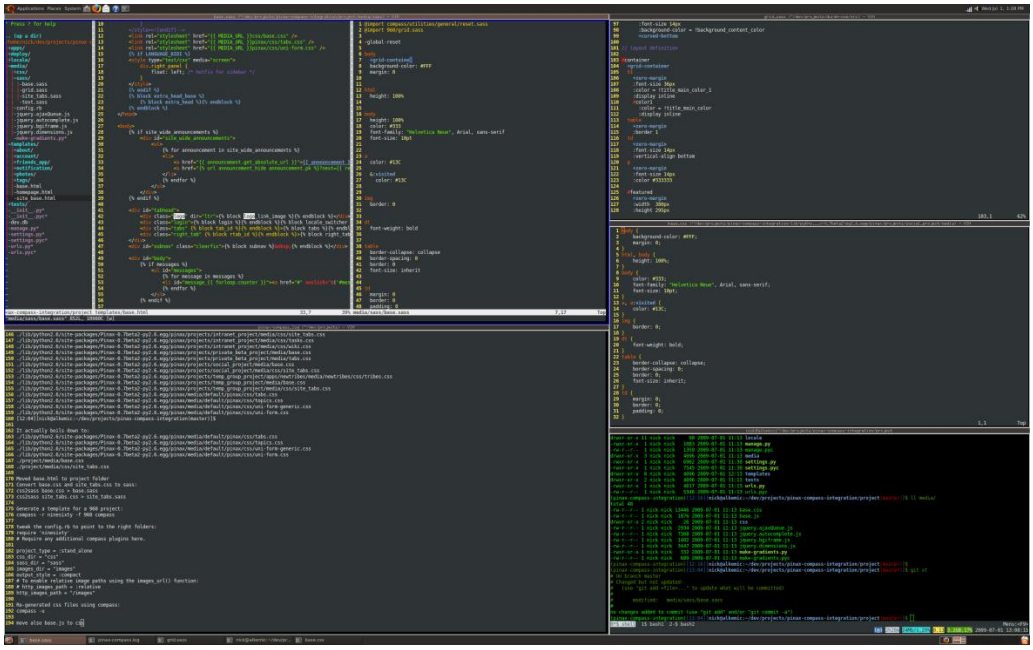


Figura 6. Gestor de ventana de mosaico

2 El sistema X Window

El sistema X Window es un software que fue desarrollado a mediados de los años 1980 en el MIT para dotar de una interfaz gráfica a los sistemas Unix. Este protocolo permite la interacción gráfica en red entre un usuario y una o más computadoras haciendo transparente la red para éste. Generalmente se refiere a la versión 11 de este protocolo, X11, el que está en uso actualmente. X es el encargado de mostrar la información gráfica de forma totalmente independiente del sistema operativo.

X fue diseñado primariamente para implementar clientes ligeros, donde mucha gente usaba simultáneamente la capacidad de procesamiento de un mismo computador trabajando en tiempo compartido. Cada persona usaba un terminal en red que tenía capacidades limitadas para dibujar la pantalla y aceptar la entrada del usuario. Debido a la ubicuidad del soporte para el software X en Unix, es usado en los computadores personales incluso cuando no hay necesidad del tiempo compartido.

El sistema de ventanas X distribuye el procesamiento de aplicaciones especificando enlaces cliente-servidor. El servidor provee servicios para acceder a la pantalla, teclado y ratón, mientras que los clientes son las aplicaciones que utilizan estos recursos para interacción con el usuario. De este modo mientras el servidor se ejecuta de manera local, las aplicaciones pueden ejecutarse remotamente desde otras máquinas, proporcionando así el concepto de transparencia de red.

El **consorcio X** rige los estándares bajo los que se desarolla el sistema X, incluyendo los siguientes:

- El mismo protocolo X.
- Los interfaces de programación, la librería Xlib y el ICCCM.
- El protocolo del gestor de ventana XDMCP.
- Los estándares de fuentes
- Las extensiones

El sistema X consta de tres partes principales, y esta separación es lo que principalmente ha brindado a X sus beneficios:

- El **servidor**, que es el software que controla el interfaz de usuario, incluyendo la pantalla, ratón y teclado.
- Los **clientes**, que son las aplicaciones, las cuales están completamente separadas del servidor.
- Un **canal de comunicación** que conecta a los clientes y al servidor.

Otra característica importante de X es que el canal de comunicación entre clientes y servidor permite que los clientes se encuentren separados en redes distintas entre sí o con el propio servidor

X ha ido evolucionando desde sus inicios, ya que en un primer momento controlaba desde el hardware hasta los protocolos de comunicaciones, lo que pronto hizo que se convirtiera en un sistema demasiado complejo para ser funcional. Es por esto que se han ido recortando las características de X para hacerlo más práctico y ligero, delegando estas tareas en el kernel u otras aplicaciones.

2.1 Características principales

X usa el modelo cliente-servidor, en el que un servidor X se comunica con varios programas cliente. El servidor acepta los pedidos para la salida gráfica (ventanas) y devuelve la entrada del usuario (desde el teclado, del ratón, o de la pantalla táctil).

El servidor puede funcionar así:

- una aplicación exhibiendo hacia una ventana de otro sistema de visualización
- un programa del sistema controlando la salida vídeo de un equipo.
- una pieza de hardware dedicada

Esta terminología de cliente servidor en la que el terminal de usuario siendo el servidor y las aplicaciones siendo los clientes, a menudo confunde a nuevos usuarios de X, ya que los términos parecen invertidos. Pero X toma la perspectiva de la aplicación, en vez de la del usuario final: X proporciona la exhibición por pantalla y los servicios de entrada/salida a las aplicaciones, así que es un servidor; las aplicaciones usan estos servicios, por lo tanto son los clientes.

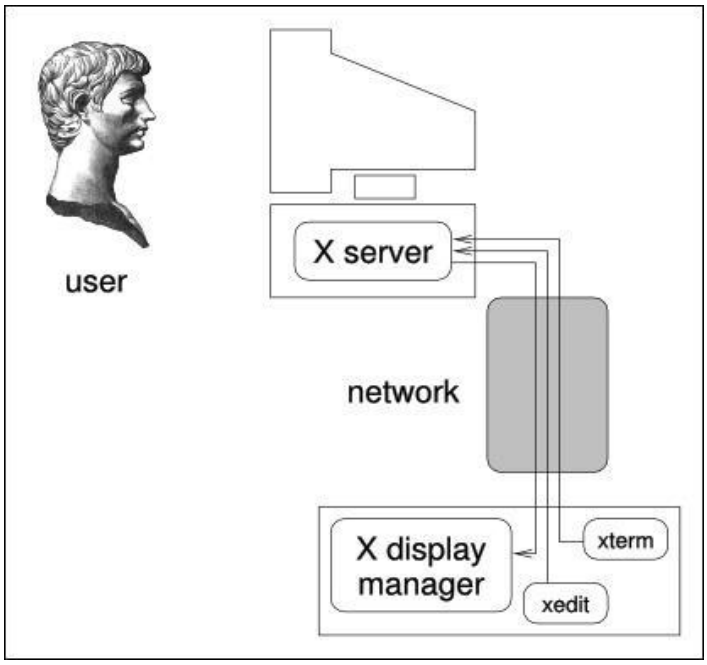


Figura 7. Esquema de comunicación del sistema X Window

X es primariamente una definición de primitivas de protocolo y gráficas, y deliberadamente no contiene especificaciones de diseño de interfaz de usuario, como estilos de botón, menú, barra de título para las ventanas. En vez de eso, un software de aplicación (tal como los manejadores de ventana, Widget toolkits de GUI y ambientes de escritorio, o las interfaces gráficas de usuario específicas de una aplicación) definen y proporcionan tales detalles. Como resultado, no hay interfaz X típica y varios ambientes de escritorio han sido populares entre los usuarios.

Un gestor de ventana controla la colocación y la apariencia de las ventanas de aplicación. Esto puede resultar en interfaces semejantes a las de Microsoft Windows o Macintosh o tener controles radicalmente diferentes.

Muchos usuarios usan X con un ambiente de escritorio, que, independientemente del manejador de ventana, incluyen varias aplicaciones usando una interfaz de usuario consistente. GNOME, KDE y Xfce son los ambientes de escritorio más populares. El ambiente estándar de Unix es Common Desktop Environment (CDE).

Tras todos estos datos, podemos definir las siguientes como las principales características del sistema X Window:

- Transparencia respecto a la red: Las aplicaciones que son ejecutadas en una máquina pueden mostrarse en otra distinta. En la práctica esto permite usar programas que corren en una máquina distinta a la que se está utilizando, pero que muestran su imagen en tu misma pantalla y toman como entrada tu teclado y ratón del mismo a como lo harían si estuvieran conectados en su misma máquina. Esta capacidad también permite el uso de aplicaciones distribuidas a través de la red.

- Independencia respecto a la red: X no depende de las características específicas de un protocolo de red, por lo que puede trabajar sobre múltiples tipologías de red.
- Independencia de los dispositivos respecto a las aplicaciones: No es necesario recompilar o incluso volver a enlazar una aplicación para trabajar con una nueva pantalla u otro dispositivo conectado.
- Servidor con una mínima dependencia de los dispositivos: el servidor ha sido desarrollado para ser lo más portable posible, por lo tanto X puede ser implementado en una amplia variedad de hardware. Otro detalle importante es que X no está embebido en el sistema operativo y se reconoce como otro programa a nivel de usuario, lo que lo hace más sencillo de implementar y de depurar en caso de error.
- Están disponibles extensiones para el sistema: Un mecanismo de extensiones está introducido en X para permitir que las nuevas funcionalidades puedan ser añadidas limpiamente y sin afectar al funcionamiento de los sistemas existentes en ese momento en lo equipos.
- Su interfaz no está predefinida: De este modo el sistema puede manejar diferentes interfaces, lo que es muy útil si se necesita que X emule a un interfaz disponible en otro sistema.
- X es no propietario: Las implementaciones tanto del servidor como de las aplicaciones están disponibles en código abierto y no hay restricciones comerciales a su uso.

2.2 Arquitectura

El servidor acepta peticiones que son enviadas a través del canal de comunicación por los clientes (aplicaciones) para crear ventanas en la pantalla, cambiar su tamaño o localización y dibujar texto o gráficos en estas ventanas. También manda eventos de vuelta hacia los clientes, informándoles sobre la modificación en la entrada del ratón o teclado o indicando ha ocurrido algún cambio en el estado de las ventanas.

El cliente contiene el código para realizar las funciones que requiera el usuario, así como el código necesario para proveer a X de la interfaz gráfica de usuario.

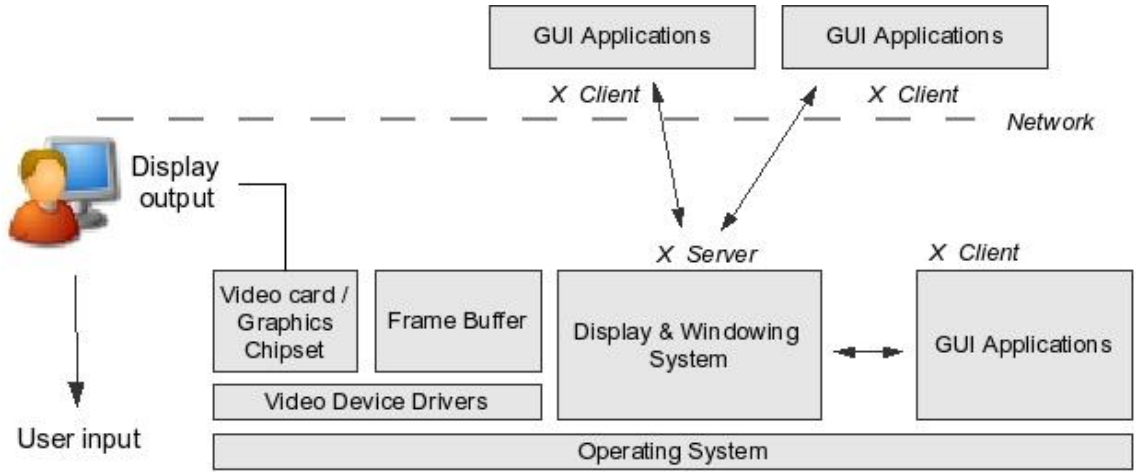


Figura 8. Arquitectura X

Dadas estas premisas nos podemos encontrar con las siguientes situaciones:

- Cliente y servidor se ejecutan en la misma CPU: En este caso el canal de comunicaciones puede estar disponible en cualquier forma posible que permita una comunicación entre procesos (sockets, pipes, memoria compartida etc)
- Cliente y servidor se ejecutan en distintas máquinas: En este caso existen algunas diferencias ya que el canal de comunicación siempre se encontrará a través de una red, debiéndose definir el protocolo de comunicación (X permite el uso de TCP/IP y OSI), y en el caso de que los dos equipos tengan diferentes arquitecturas, la comunicación será realizada mediante objetos de alto nivel que permitirán la comunicación.

Los elementos de la interfaz gráfica de usuario, como menús, botones o barras de scroll de una aplicación en X son parte de esa aplicación y no dl servidor. Debido a esto, todo el interfaz de usuario de X es modificable, de manera que X permite usar múltiples interfaces distintos.

Esta arquitectura permite correr una aplicación localmente o remotamente, con la singularidad de que el interfaz de usuario no es parte del servidor pero consta de dos partes diferenciadas:

1- La interfaz de aplicación que es construida en cada aplicación cuando ésta es compilada, normalmente mediante algún toolkit, por lo que distintas aplicaciones pueden tener distintas interfaces de aplicación. Este interfaz puede ser modificado únicamente cambiando el código fuente del programa para usar un toolkit diferente y recompilándolo posteriormente.

2- El gestor de ventanas es el que provee la interfaz de gestión y puede ser modificado si se cambia el gestor de ventanas

Para entender cómo funciona la arquitectura de X, vamos a realizar el seguimiento de un evento desde el dispositivo de entrada hasta el punto donde este cambio aparece en pantalla:

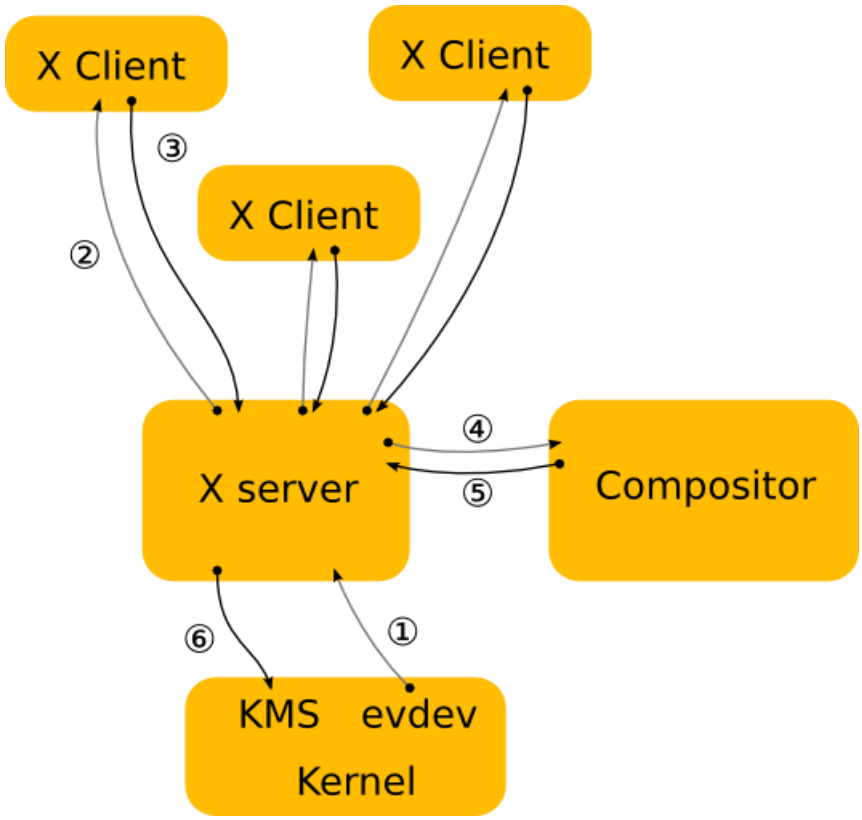


Figura 9. Pasos del funcionamiento de la arquitectura del sistema X Window

1. El kernel recoge un evento de un dispositivo de entrada y lo manda hacia X través del driver de entrada evdev, traduciendo su protocolo.

2. El servidor X decide a qué ventana afecta el evento y lo envía a los clientes que han sido seleccionados

3. El cliente recoge el evento y decide qué hacer con él. Eventualmente la interfaz de usuario puede haber cambiado en este punto debido a una interacción, por lo que el cliente manda una petición de renderizado de vuelta hacia el servidor X.

4. Cuando el servidor X recibe la petición, la envía al driver para permitir que al hardware realizar el renderizado. El servidor X también calcula la región colindante a la renderización y envía un evento de “daño” (damage).

5. El evento de “daño” avisa al compositor que algo ha cambiado en la ventana y que hay que recomponer la parte de la pantalla donde la ventana es visible. La pantalla renderiza los contenidos través del servidor X.

6. El servidor X recibe la petición de renderizado del compositor.

Los beneficios derivados de este diseño permiten que mediante X sea posible integrar a las aplicaciones en diferentes plataformas en un mismo entorno computacional de usuario, gracias al uso de un mismo interfaz de usuario, por lo que es posible gestionar programas y datos de una manera centralizada o distribuirlos entre unidades funcionales.

2.3 Protocolo ICCCM

ICCCM es el acrónimo de Inter-Client Communication Conventions Manual (“Manual de convenciones de comunicación entre clientes) y es el estándar para interoperabilidad entre los clientes de X Window System pertenecientes al mismo servidor X. Se usa principalmente para la comunicación entre clientes “normales” y el gestor de ventanas.

El ICCCM especifica el cortado y pegado de buffers, la interacción del gestor de ventanas, gestión de sesiones, cómo se manipulan los recursos compartidos y cómo gestionar los dispositivos destinados al color. Estas funciones de bajo nivel, son generalmente implementadas por manejadores adicionales que evitan que los programadores tengan que tratar directamente con ICCCM.

La mayoría de las comunicaciones se llevan a cabo mediante el uso de propiedades sobre las ventanas de aplicación de los niveles superiores. El gestor de ventana puede también generar eventos que estarán disponibles para la aplicación.

Para la comunicación entre los clientes, ICCCM usa el concepto denominado **Atom** (expresando de modo explícito el concepto de brevedad que los caracteriza), que es un nombre único formado por octetos hasta 32 bits y representado por strings. El motivo de uso de estos Atoms es debido a que su representación se ciñe a un string de tamaño corto y fijo, lo que favorece el envío de las clases de paquetes que más probabilidades tienen de ser enviados con los mismos strings, haciendo que el uso de los canales de comunicación sea más eficiente.

Los Atoms son elegidos por el servidor, por lo que si el cliente quiere que un nuevo Atom sea creado, sólo deberá enviar el string que será almacenado, mientras que el identificador será elegido por el servidor y devuelto al cliente. Son usados para muchos propósitos distintos relacionados con la comunicación entre varios clientes conectados con el mismo servidor, en asociación con las propiedades de las ventanas.

NetWM o Extended Window Manager Hints, es un estandar de X Window System para gestores de ventanas. Define varias interacciones entre los gestores de ventanas, utilidades y aplicaciones, todo como parte de un entorno de escritorio completo, y a su vez, está construido sobre las funcionalidades del ICCCM y que ayuda a completar ciertos comportamientos no especificados en

primera instancia en este protocolo pero que son comúnmente conocidos en los gestores de ventanas y de escritorio modernos.

2.4 Librerías de interfaces de usuario para el entorno X Window

Qt y GTK + son los frameworks de desarrollo de código abierto para el interfaz de usuario más importantes dentro del mundo GNU/Linux, ya que ambas opciones son de código abierto y permiten a los desarrolladores el uso de unas herramientas muy potentes para diseñar interfaces gráficas. GTK+ es usado como el kit de trabajo estándar para el proyecto GNOME, LXDE y Xfce, mientras que Qt se usa en KDE y Razor-Qt.

Framework Qt

Este framework multiplataforma es usado de una manera muy amplia para el desarrollo de aplicaciones con interfaz gráfico mediante el uso de C++ y Java. También posee soporte de internalización, acceso SQL, parseo de XML, uso de hilos y gestión de ficheros.

Está distribuido bajo licencia GNU y desarrollado como como proyecto open source (Qt Project), por lo que todas las ediciones poseen muchos compiladores, incluyendo GCC, el compilador de C++ y la suite Visual Studio.

Qt presenta una serie de módulos que facilitan su uso a la hora de desarrollar aplicaciones:

- Módulos para desarrollo general de software: QtCore (contiene el núcleo no gráfico), QtGui (colección de los componentes básicos), QtNetwork (útiles para escribir clientes y servidores TCP/IP), QtXML etc.
- Módulos para trabajar con las herramientas Qt: (QtHelp, QtDesigner, QtUiTools, QtTest)
- Módulos para desarrollo específicos del SO: (En GNU/Linux encontramos el módulo QtDBus, para la comunicación entre procesos).

Otro punto importante de Qt es el moc (Meta Object Compiler), ya que este framwork hace un uso intensivo de este generador especial de código junto con otras macros para hacer el lenguaje más completo. El moc interpreta estas macros como anotaciones y se usan para generar código C++ añadido con meta información sobre las clases que se usan en el programa. Esta información la usa Qt para presentar información no disponible nativamente en C++ a los desarrolladores (los sistemas signal / slot y las llamadas asíncronas).

En el gráfico que encontramos a continuación, podemos observar un esquema de la integración de Qt con C++ y Java, y cómo está definida la estructura modular que indicamos.

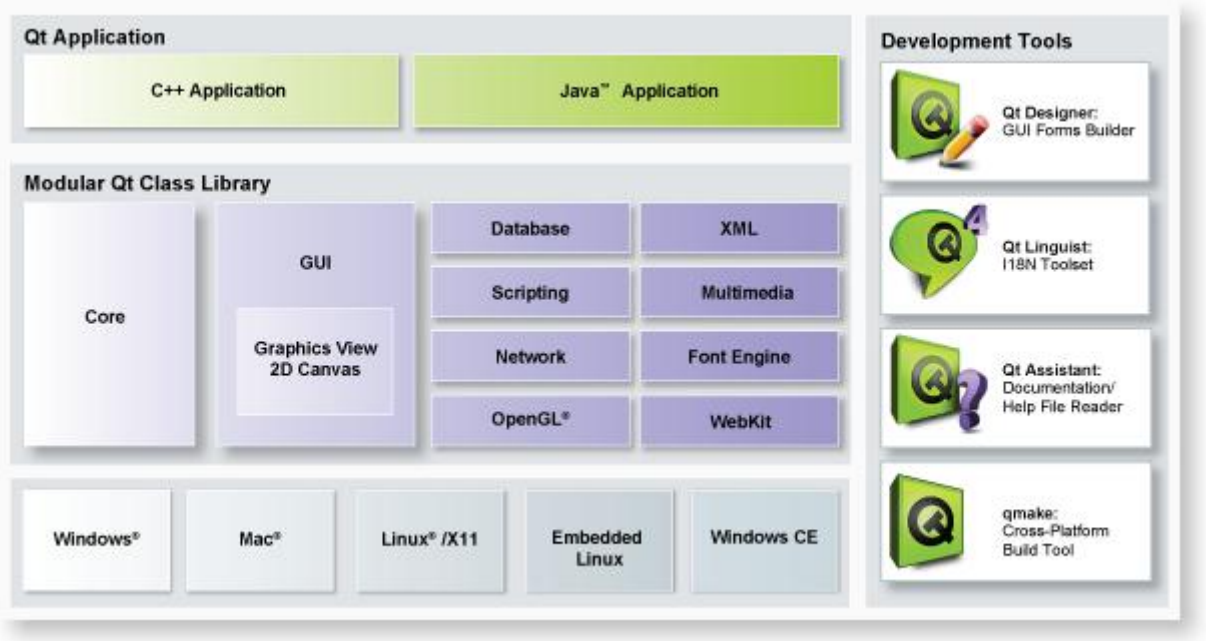


Figura 10. Esquema de módulos de Qt y relación con C++ y Java

GTK+

Este kit de desarrollo de widgets multiplataforma que permite el uso de software libre y propietario se ha convertido en una de las librerías más importante para el sistema X Window.

GTK fue desarrollado inicialmente para el programa GIMP en 1997 y actualmente está mantenido por miembros de la fundación GNOME.

Está orientado a objetos mediante el uso de las librerías GLib y escrito en lenguaje C. El uso de Xlib en el servidor de pantalla de X11 le brinda la flexibilidad necesaria para ser usado en plataformas donde el sistema X Window no está disponible, por lo que aunque fue diseñado para el uso en sistemas GNU/Linux, puede funcionar también en Windows y Mac OSX.

Puede ser también configurado para cambiar el diseño de widgets ya dibujados, lo que se realiza mediante el uso de diferentes motores de visualización.

3 Evolución de los Gestores de ventanas GNU/Linux

A continuación se van a presentar los principales gestores de ventanas que han evolucionado a partir de la funcionalidad de X Window, analizando sus características y qué aportan como evolución respecto a X.

3.1 WayLand

Wayland provee un método para que los gestores de pantalla de composición se comuniquen directamente con aplicaciones y hardware de vídeo. Las aplicaciones dibujan las imágenes usando sus propios buffers y los gestores de pantalla se convierten en los servidores de pantalla, componiendo todos estos buffers para de este modo formar la pantalla que dibujará las ventanas de la aplicación.

Este método se ha demostrado más eficiente y simple que mediante el uso de gestores de ventanas de composición con el sistema X Window y gracias a esa mejora de eficiencia, los gestores Compiz, KWin y Mutter están empezando a implementar soporte directo para WayLand.

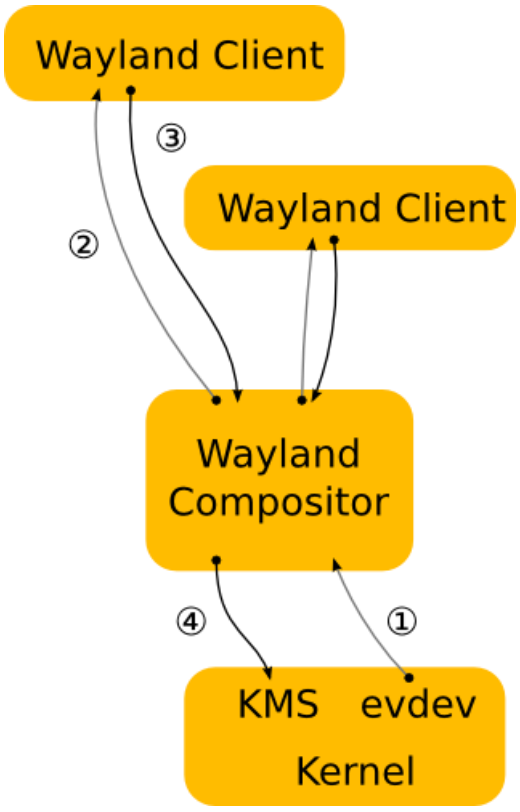


Figura 11. Arquitectura Weyland

Wayland, a diferencia de X está escrito desde 0 y no carga con un legado de código heredado de antiguas implementaciones y componentes ya obsoletos. También es un programa más ligero ya que únicamente se encarga de coordinar el flujo de datos gráficos entre el kernel y las bibliotecas de vídeo, aplicaciones y pantalla que son mostradas al usuario.

Usa OpenGL, lo que lo convierte en más potente que X a pesar de ser más sencillo, permitiendo la aceleración de hardware, y facilitando de este modo la experiencia del usuario.

Por otro lado, la falta de integración de los proyectos actuales realizados con Qt y GTK con la arquitectura de WayLand, está complicando su despegue como alternativa real y generalizada al uso de X. A pesar de ello, la decisión de Ubuntu de sustituir X por este sistema, avanza que el proyecto tiene un futuro asegurado.

Es también destacable que es compatible con X y sus programas, ya que lo usa como si fuera un cliente más, lo que permite una migración sencilla a los programas actuales.

Figura 12. Escritorio ejecutando Weyland

Aplicación

- Ubuntu: Ha sido anunciado que WayLand reemplazará a X como el servidor primario de pantalla para el entorno de escritorio Unity de Ubuntu.
- KDE: El gestor de pantalla KWin ha añadido soporte para las librerías de salida OpenGL, y del mismo modo permite la modificación inicial para el soporte de WayLand a través del anuncio de que el principal desarrollador de este gestor empezará a realizar el port que se espera estará finalizado en 2013.

- Fedora: Se especula que WayLand se convierta en un futuro cercano en el gestor de ventanas por defecto, ya que uno de sus principales desarrolladores ha alabado las características de este gestor incidiendo en que sus novedades lo hacen ser la opción adecuada para el desarrollo de Fedora.
- Compiz: El movimiento de dependencias de X dentro de un plugin, facilita que Compiz pueda ser usado como gestor de pantalla de WayLand. Del mismo modo el anuncio de Ubuntu, y su movimiento respecto a Unity implica un acercamiento hacia este gestor de ventanas.

3.2 MicroXwin

MicroXwin es un gestor de ventanas similar a X, pero que debido a su diferente arquitectura, provee una interfaz gráfica de usuario más rápida que el sistema X Window, pero que a su vez permite mantener la compatibilidad con el estándar de las aplicaciones X11, así como reducir la cantidad de memoria necesaria por el sistema y los recursos destinados al disco.

La lentitud relativa de X es debida al uso del modelo cliente-servidor y a su gestión separada en dos partes, creando de este modo una latencia entre el cliente y el servidor y haciéndose necesario un cambio de contexto entre la ejecución de código del cliente y el servidor.

La arquitectura de MicroXwin se ha desarrollado para intentar paliar estos defectos, el servidor X ha sido remplazado por módulos del kernel (“X11.ko, LibX11.so y LibXext.so”) de manera que son usados por los clientes de X para interactuar con estos módulos.

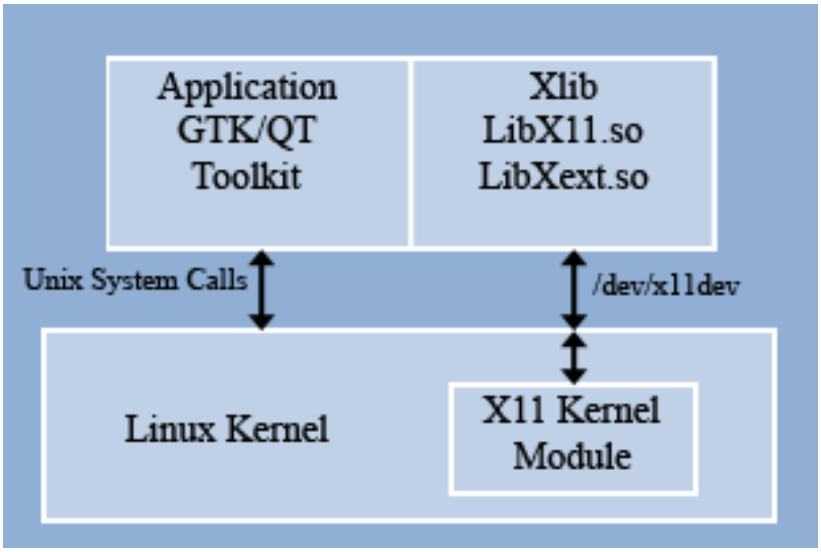


Figura 13. Relación entre módulos de MicroXWin

Mediante esta opción de diseño se obtienen diferentes ventajas, como una latencia menor, menor intercambio de peticiones, menor número de cambios de contexto y sin la necesidad de realizar cambios adicionales para aplicaciones ya existentes.

La última versión de este gestor distribuye un kernel ya unificado que soporta la implementación tanto del sistema operativo Android como de X en el mismo paquete, lo que implica que el equipo de desarrollo está intentando aplicar sus características de ligereza a dispositivos móviles.

3.3 Metisse

El gestor de ventanas Metisse también está basado en el sistema X Window, basado en intentar facilitar el diseño y la implementación de los gestores considerados “clásicos” en GNU/Linux y más cercano al gestor FVWM.

Su arquitectura está basada en el tipo de composición, creando una distinción entre el dibujo de las ventanas y la interacción entre los procesos de composición. El servidor Metisse es un servidor X modificado que dibuja las ventanas de aplicación fuera de la pantalla, mientras que la parte destinada a la composición es una modificación de la versión del gestor FVWM combinada con un visor llamado FvwmCompositor.

FvwmCompositor usa librerías OpenGL para mostrar las ventanas. Esta librería ofrece modelos de gráficos bien adaptados para esta gestión de ventanas, de manera que es posible la realización de transformaciones de tamaños de ventana en tiempo real, escalados o rotaciones en 2D y 3D.

Metisse también implementa las denominadas User Interface Façades, un sistema que proporciona al usuario herramientas para adaptar, reconfigurar y recombinar interfaces gráficas ya existentes, mediante el uso de técnicas de manipulación directas.

Figura 14. Ejemplo de efectos en Metisse

3.4 Xynth

Xynth es un gestor de ventanas embebido, lanzado bajo licencia LGPL y desarrollado para sistemas con bajos recursos como alternativa al uso del sistema X Window. El objetivo de este proyecto es conseguir un sistema ligero pero portable.

En cuanto a su arquitecura, Xynth actúa como un interfaz entre el hardware del sistema y el entorno de escritorio, trabajando en gran cantidad de hardware, incluyendo dispositivos embebidos.

3.5 DirectFB

DirectFB proviene de Direct Frame Buffer y es una librería para GNU/Linux que provee aceleración gráfica por hardware, manejo de dispositivos de entrada y un sistema de ventanas integrado con soporte para transparencias.

Esta librería permite a los desarrolladores una alternativa a sistema X Window completo, ya que permite a las aplicaciones comunicarse directamente con el hardware de vídeo, acelerando y simplificando de este modo las operaciones gráficas.

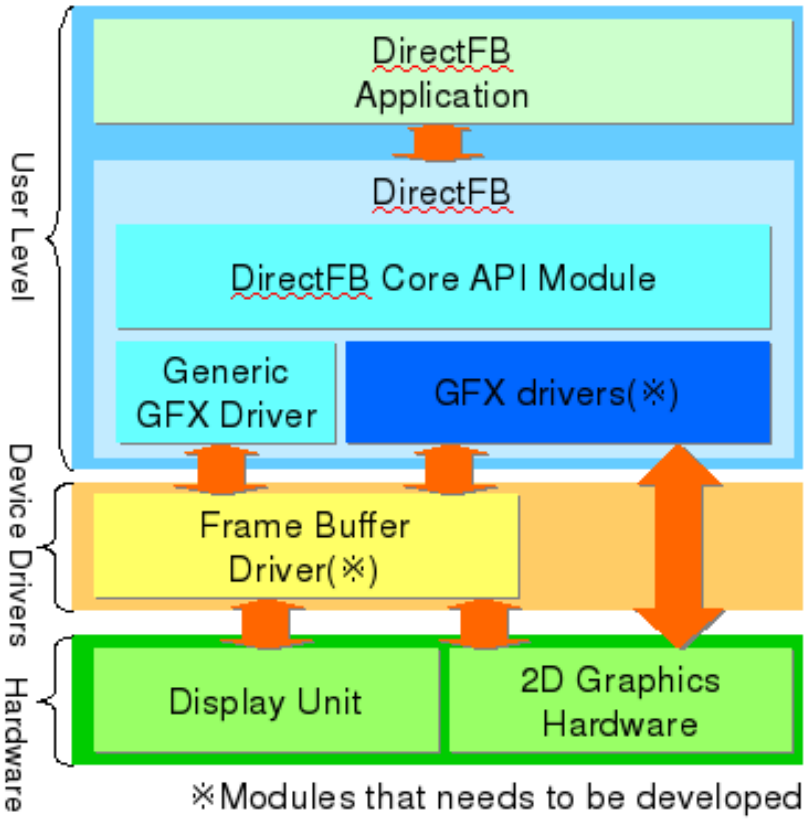


Figura 15. Esquema del funcionamiento de DirectFB

4 Análisis práctico de Gestores de ventanas en GNU/Linux

Se presenta a continuación un estudio de los gestores de ventanas usados en los gestores de escritorio más utilizados en GNU/Linux, como puede ser por su importancia desde el punto de vista de número total de usuarios, KWin o Compiz, pero incorporando también a Enlightenment, Mutter, Fluxbox, Openbox y Xfwm.

Para ello, se realizará un análisis de sus características, destacando sus principales diferencias y en qué situaciones se hace más recomendable optar por su uso, añadiéndose también una prueba práctica que analizará el rendimiento de estas soluciones en una máquina real.

La máquina utilizada para el análisis ha sido un equipo portátil HP 6375s con las siguientes características:

- Procesador AMD Athlon 64,
- 4Gb de memoria RAM DDR2 800
- 250Gb de disco duro de 7200rpm
- ATI Radeon HD3200
- Sistema operativo Kubuntu 12.04 LTS. Con arquitectura de 64bits

En las pruebas de rendimiento se ha intentado recrear la misma carga del sistema para lograr que los resultados sean lo más ajustado a la realidad posible, aunque debido a que los gestores de ventanas a menudo se encuentran integrados en entornos de aplicaciones más complejos, estos resultados se deberán tratar más como una aproximación a la realidad que nos podemos encontrar al enfrentarnos a cada una de estas aplicaciones.

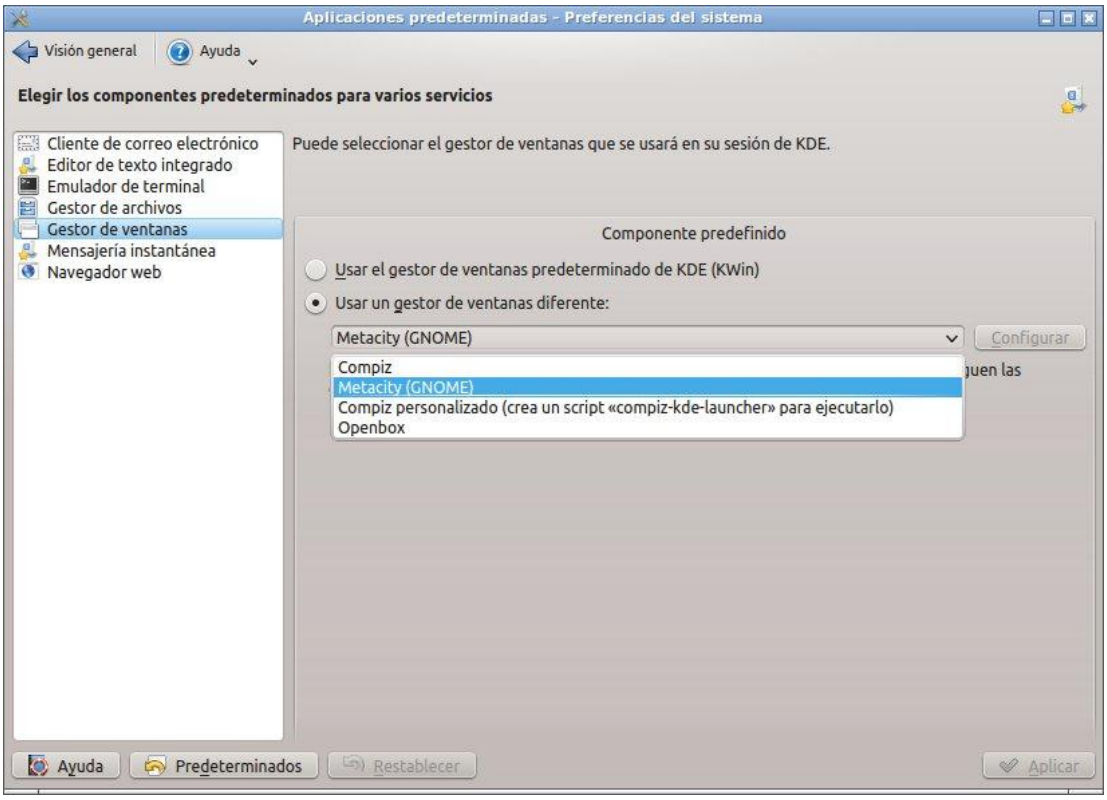


Figura 16. Selección de gestores de ventanas en Kubuntu

Para obtener resultados medibles de rendimiento se han valorado varias aplicaciones de benchmarking disponibles en GNU/Linux, como Hardinfo, o Phoronix Test Suite, pero finalmente se ha realizado un test individual a cada gestor con la aplicación Gtkperf, ya que es la única que se ha mostrado como una herramienta lo suficientemente ligera y compatible con la ejecución de tests incluso en los gestores de ventanas con menor capacidad gráfica.

Gtkperf realiza diferentes rutinas y basa sus resultados en el tiempo de ejecución total de éstas. Las rutinas que efectúa son las siguientes:

- GtkEntry (test de textos cambiando entre tipos),
- GtkComboBox y GtkComboBoxEntry (test de combo box, abriendo y cerrando estos campos de formulario),
- GtkSpinButton (test de campos con datos para aumentar / disminuir valores),
- GtkProgressBar (test de ejecución de barras de progreso),
- GtkToogleButton y GtkCheckButton (test de valores on/off),
- GtkRadioButton (test de botones de selección tipo *radio button*),
- GtkTextView (test de scroll de texto),
- GtkDrawingArea (test de dibujo de líneas, círculos, textos y una imagen predefinida de modo aleatorio).

4.1 KWIN

KWin es un gestor de ventanas para el sistema X Window que forma parte integral del entorno KDE pero que puede ser también utilizado en otros entornos de escritorio.

Actualmente KWin se encuentra en su versión estable 4.8, y en su último cambio principal de versión fueron añadidas funcionalidades experimentales con soporte para las extensiones GL de modo que es compatible con la utilización de efectos de tipo Compiz.

La apariencia de KWin trae incluída varias decoraciones y viene definida por defecto por la decoración denominada Oxygen, También puede usar temas del gestor IceWM y permite descargar decoraciones de Internet.

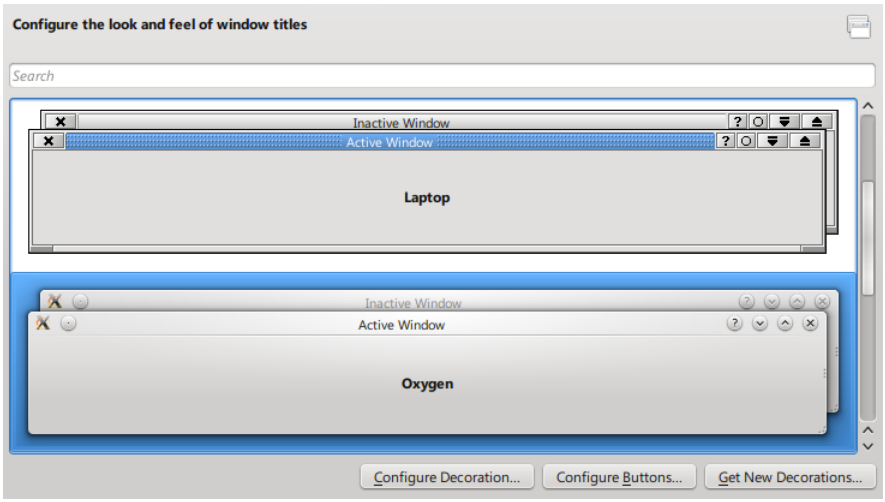


Figura 17. Ejemplos de decoración de Kwin

Los principales efectos que incluye KDE son los siguientes:

Efectos de apariencia:

- Explosión: Hace que la ventana explote cuando se cierra.
- Desvanecimiento: Las ventanas se desvanecen suavemente cuando aparecen o desaparecen.
- Desvanecimiento del escritorio: Entre varios escritorios virtuales cuando se intercambia el foco entre cada uno de ellos.
- Caída: Hace que las ventanas se deshagan en varias piezas.
- Login-logout: Efecto de desvanecimiento al logarse.
- Lampara mágica: Simula una lámpara mágica al minimizar ventanas.
- Animación al minimizar ventanas.
- Marcas de ratón: Permite dibujar líneas en el escritorio.
- Deslizar: Desliza ventanas al cambiar entre escritorios virtuales.

- Transparencias en distintas condiciones.
- Deformación de las ventanas al moverse.
- Simula que cae nieve en el escritorio.

Efectos de accesibilidad:

- Invertir colores de escritorio y ventanas.
- Aumentador de pantalla que simula el efecto de una lupa.
- Agrandar la sección de la pantalla cercana al cursor.
- Ayuda a la localización del centro de la pantalla al mover una ventana.
- Mostrar efector del cursor para ayudar a localizarlo.
- Agrandar el escritorio completo.

Efectos de foco:

- Oscurece la ventana principal.
- Oscurece las ventanas inactivas.
- Oscurece toda la pantalla al pedir privilegios de root.

Figura 18. Escritorio ejecutando KWin con efecto cubo (OpenSuse – KDE)

Análisis

KWin es un gestor con gran potencia gráfica, lo que a su vez hace que su consumo de recursos sea elevado, especialmente si hacemos un uso intensivo de sus efectos en 3D.

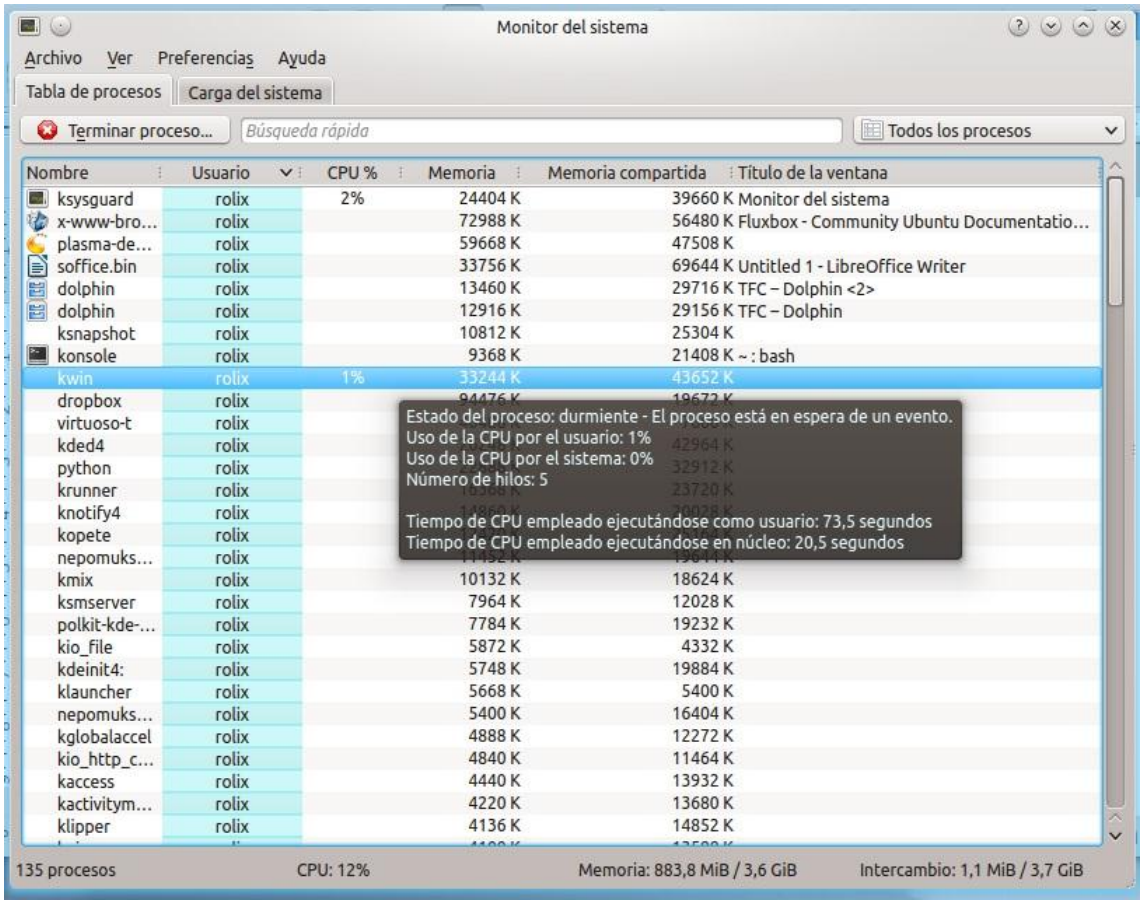
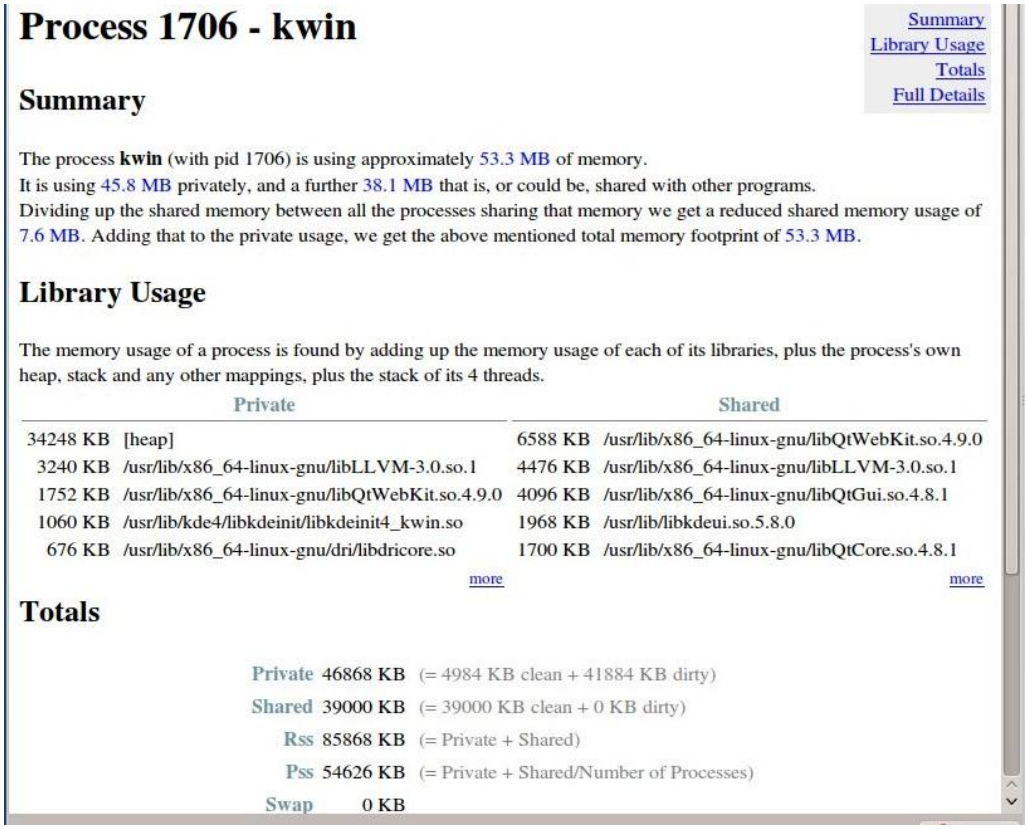


Figura 19. Monitor del sistema con proceso Kwin

En esta captura podemos encontrar los datos arrojados por el monitor de sistema y correspondientes al sistema prácticamente en reposo. El consumo de memoria en ese instante es de 33244KB + 43000KB de memoria compartida, lo cual no es considerable para un equipo actual, pero si puede ser representativo para un sistema en el que la memoria RAM sea un recurso limitado. También el consumo de CPU es limitado, pero como hemos comentado anteriormente se encontraba prácticamente en reposo, sin ejecutar ninguna aplicación y sin forzar al gestor a tener que ejecutar ninguna tarea aplicable a su cometido.

A continuación, en la siguiente captura, se observan las características del proceso ejecutándose, y donde ya podemos observar datos más ajustados. Por un lado, observamos que la memoria que el sistema informa sobre el proceso

es de 53,3MB, y también obtenemos datos de las librerías principales, teniendo gran protagonismo el uso de las librerías Qt características de este gestor.



Process 1706 - kwin

Summary
Library Usage
Totals
Full Details

Summary

The process **kwin** (with pid 1706) is using approximately 53.3 MB of memory.
It is using 45.8 MB privately, and a further 38.1 MB that is, or could be, shared with other programs.
Dividing up the shared memory between all the processes sharing that memory we get a reduced shared memory usage of 7.6 MB. Adding that to the private usage, we get the above mentioned total memory footprint of 53.3 MB.

Library Usage

The memory usage of a process is found by adding up the memory usage of each of its libraries, plus the process's own heap, stack and any other mappings, plus the stack of its 4 threads.

Private		Shared	
34248 KB	[heap]	6588 KB	/usr/lib/x86_64-linux-gnu/libQtWebKit.so.4.9.0
3240 KB	/usr/lib/x86_64-linux-gnu/libLLVM-3.0.so.1	4476 KB	/usr/lib/x86_64-linux-gnu/libLLVM-3.0.so.1
1752 KB	/usr/lib/x86_64-linux-gnu/libQtWebKit.so.4.9.0	4096 KB	/usr/lib/x86_64-linux-gnu/libQtGui.so.4.8.1
1060 KB	/usr/lib/kde4/libkdeinit/libkdeinit4_kwin.so	1968 KB	/usr/lib/libkdeui.so.5.8.0
676 KB	/usr/lib/x86_64-linux-gnu/dri/libdricore.so	1700 KB	/usr/lib/x86_64-linux-gnu/libQtCore.so.4.8.1
	more		more

Totals

Private	46868 KB	(= 4984 KB clean + 41884 KB dirty)
Shared	39000 KB	(= 39000 KB clean + 0 KB dirty)
Rss	85868 KB	(= Private + Shared)
Pss	54626 KB	(= Private + Shared/Number of Processes)
Swap	0 KB	

Figura 20. Características del proceso Kwin

Valores de ejecución de Gtkconf:

```
GtkEntry - time:  0,21
GtkComboBox - time:  2,55
GtkComboBoxEntry - time:  2,78
GtkSpinButton - time:  0,60
GtkProgressBar - time:  0,40
GtkToggleButton - time:  0,85
GtkCheckButton - time:  0,84
GtkRadioButton - time:  0,89
GtkTextView - Add text - time:  2,45
GtkTextView - Scroll - time:  2,40
GtkDrawingArea - Lines - time:  0,91
GtkDrawingArea - Circles - time:  1,32
GtkDrawingArea - Text - time:  1,57
GtkDrawingArea - Pixbufs - time:  0,24
 ---
Total time: 18,01
```

Los datos de benchmarking, no pueden ser tomados como una prueba de rendimiento fiable para testear Kwin, ya que está basado en la ejecución de pruebas Gtk, pero si son útiles para intuir en qué valores pruebas Kwin es más eficaz respecto al resto de gestores de ventanas, ya que las resultados de las pruebas específicas de dibujo no resultan diferentes a los obtenidos en los restantes gestores.

Como conclusión, hemos encontrado en las pruebas que KWin es un gestor que reúne características interesantes para equipos en los que los recursos del sistema no se encuentren muy limitados. Por un lado, los efectos visuales y su capacidad de configuración lo hacen una opción muy potente dentro del abanico de posibilidades que hemos analizado, mientras que su consumo de memoria, sin ser bajo, ha respondido de manera positiva durante la prueba, sin llegar a picos altos de consumo de CPU aun ejecutando cambios de escritorio en 3D durante un periodo continuado de tiempo, y respondiendo en todo momento de manera muy rápida.

4.2 Enlightenment

Enlightenment (también llamado E) es un gestor de ventanas de tipo pila, que puede usarse sólo o conjuntamente con un entorno de escritorio como GNOME o KDE y que es usado a menudo como sustituto de un gestor completo de escritorio.

Su principal característica es el bajo consumo de recursos que requiere para funcionar, aunado a que es bastante personalizable y permite la instalación de temas visuales llamativos. Todas estas virtudes han logrado convertirlo en una opción muy popular.

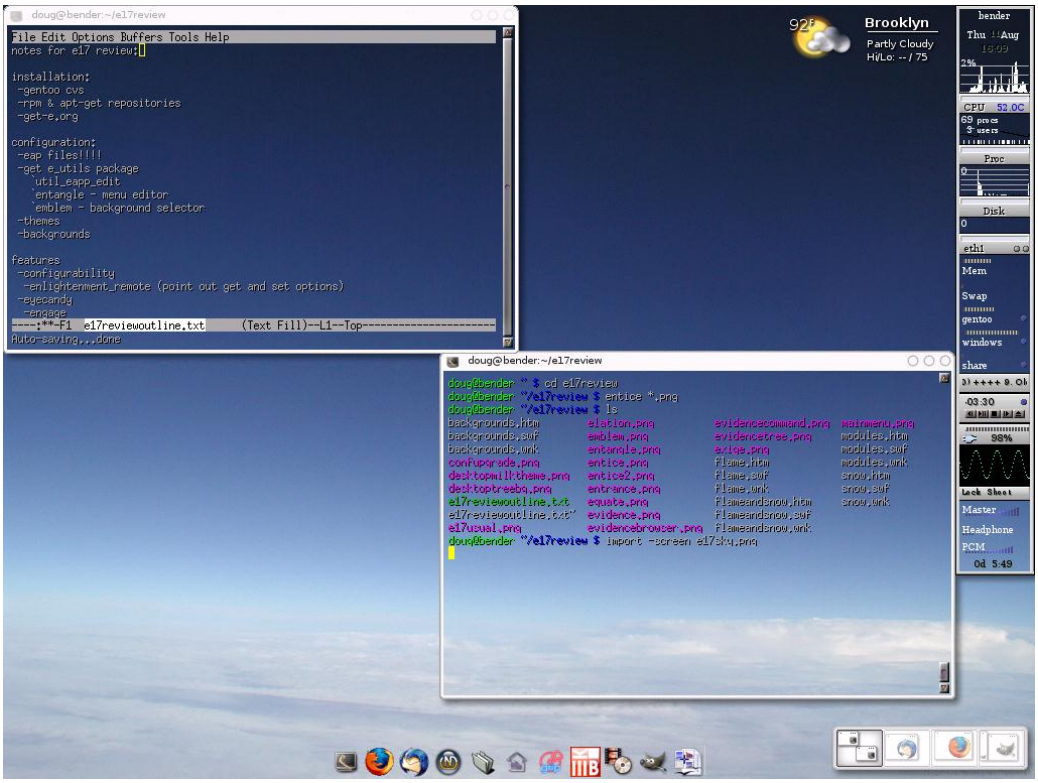


Figura 21. Escritorio de equipo ejecutando Enlightenment

Análisis

En esta primera captura observamos el árbol de procesos del sistema con la ejecución de Enlightenment, y cómo su proceso principal cuelga del proceso kdm, ya que en esta prueba el gestor se está ejecutando en un sistema operativo Kubuntu (durante el proceso de instalación del gestor, el instalador pregunta si se desea instalar bajo gdm o kdm) y del que a su vez también parte el proceso Xorg.

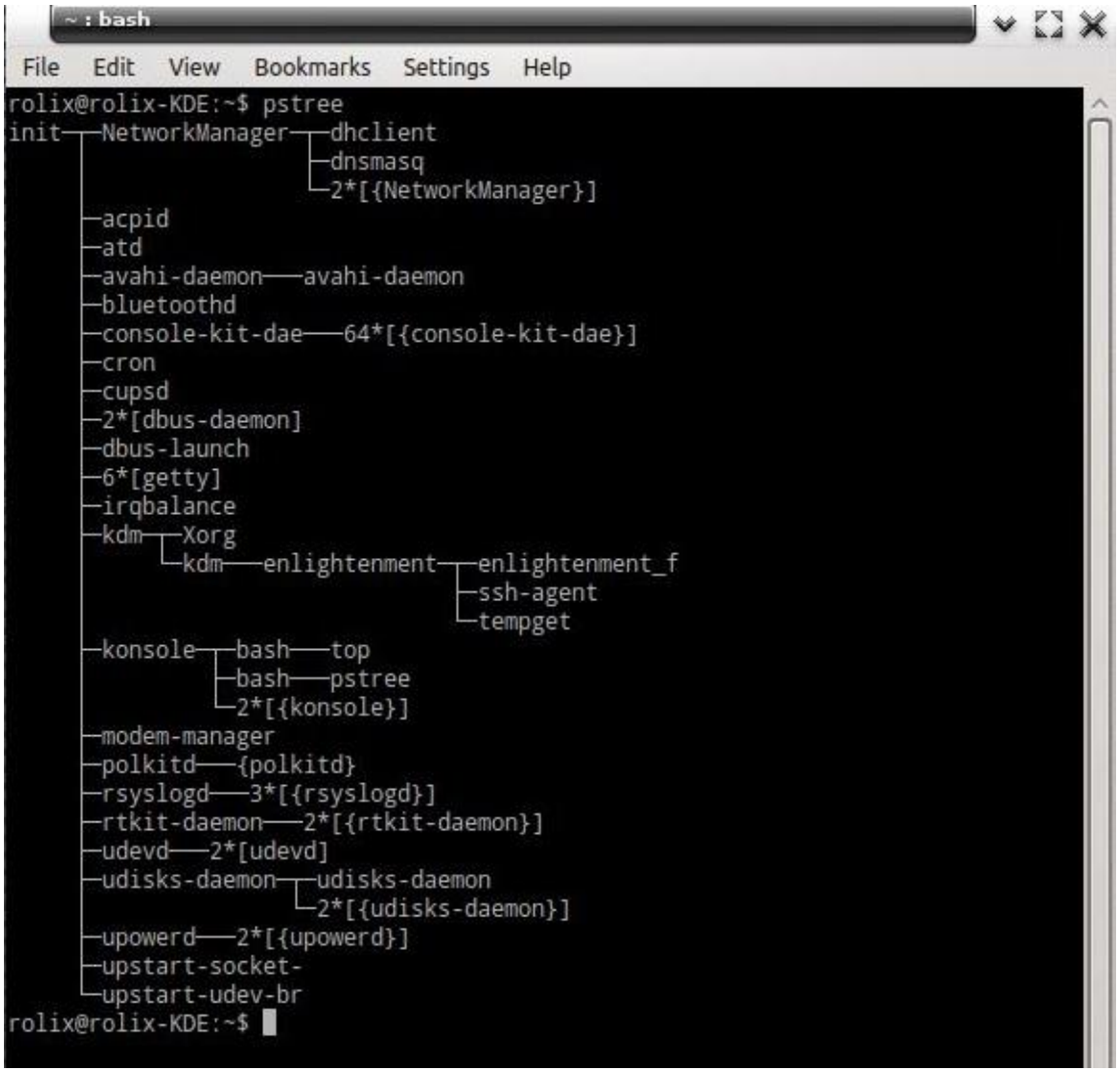


Figura 22. Pstree de equipo ejecutando Enlightenment

En esta captura del monitor del sistema del equipo de prueba podemos observar el bajo consumo de recursos de que hace gala este gestor de ventanas. El consumo de memoria principal es de 13,5 MB, mientras que el uso de memoria compartida del proceso es de únicamente 12,6 MB.

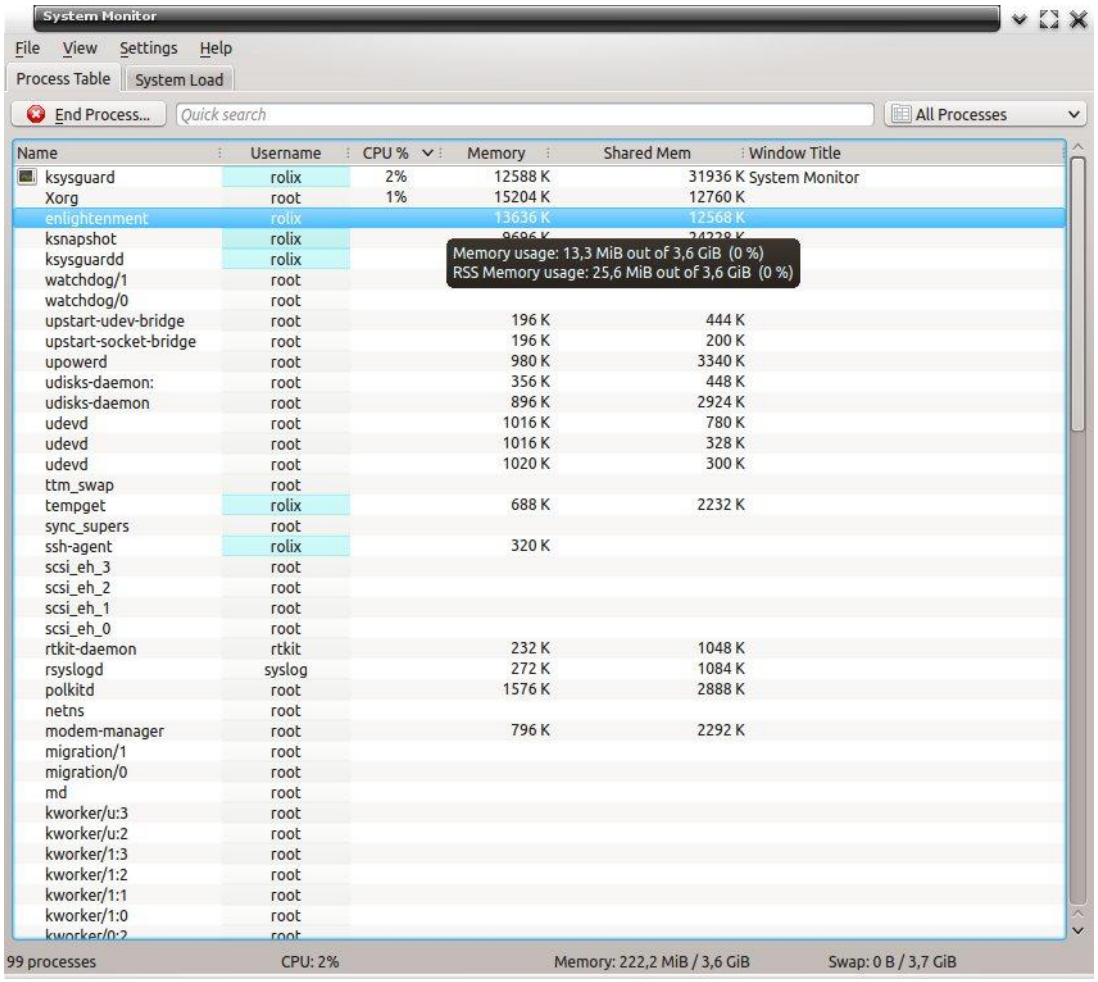


Figura 23. Monitor de sistema de equipo ejecutando Enlightenment

Por otro lado, en la siguiente captura se nos muestra la información detallada del proceso enlightenment, con un consumo total aplicable de 20,8MB, mientras que el consumo principal de las librerías usadas por este gestor es aplicable a las libc y X11 de manera compartida, y al uso de su tema visual aplicado desde el punto de vista de memoria no compartida. En ambos casos destaca que en ningún momento observamos un consumo mayor a 1MB por parte de estas librerías, lo que nos vuelve a confirmar el bajo gasto de recursos que caracteriza a este gestor.

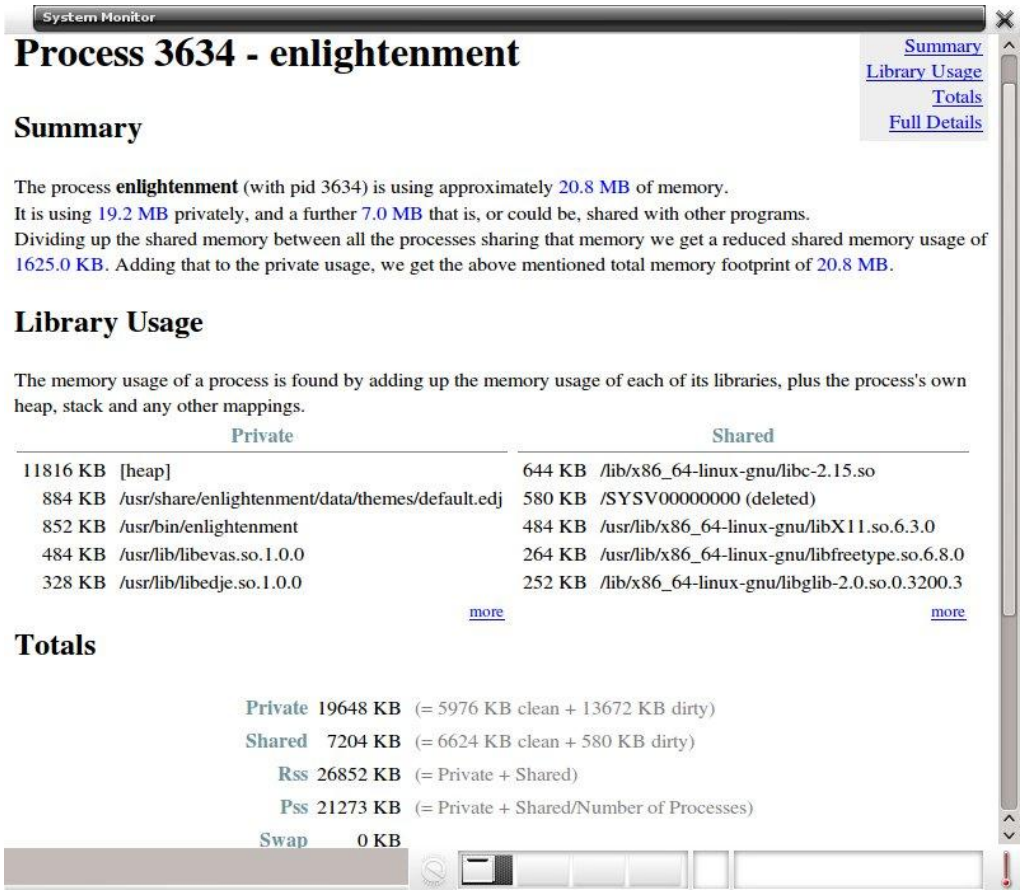


Figura 24. Características del proceso enlightenment

Valores de ejecución de GtkConf:

```
GtkEntry - time:  0.03
GtkComboBox - time:  0.70
GtkComboBoxEntry - time:  0.60
GtkSpinButton - time:  0.06
GtkProgressBar - time:  0.03
GtkToggleButton - time:  0.07
GtkCheckButton - time:  0.06
GtkRadioButton - time:  0.10
GtkTextView - Add text - time:  0.82
GtkTextView - Scroll - time:  0.30
GtkDrawingArea - Lines - time:  0.92
GtkDrawingArea - Circles - time:  1.25
GtkDrawingArea - Text - time:  1.90
GtkDrawingArea - Pixbufs - time:  0.26
 ---
Total time:  7.11
```

Enlightenment es un gestor recomendado para equipos en los que no sea necesario hacer uso de animaciones con efectos complejos en 3D, pero que no por ello deseen un gestor con pocas capacidades. Este gestor, con un consumo de recursos muy limitado, visualmente no desmerece con otros gestores más potentes, pero su gran virtud se encuentra en la eficiencia y su gestión de recursos.

4.3 Mutter + Compiz

Mutter es un gestor de ventanas utilizado en el entorno de escritorio GNOME3 (Metacity ha sido reemplazado en esta última versión). Extensible con diversos plugins y con soporte para numerosos efectos visuales.

El nombre de Mutter hace referencia a Metacity + Clutter, siendo esta última una librería gráfica que añade soporte para Open GL.

La controversia ha rodeado a este cambio de filosofía en GNOME3, ya que no permite la utilización de otro gestor de ventana sin tener que volver a escribir el código, por otro lado, su excesivo consumo de recursos ha sido también muy criticado.

Figura 25. Escritorio Mutter + Compiz con efectos de transparencia

Compiz es un gestor de ventanas de tipo composición que se caracteriza por usar hardware gráfico 3D para crear efectos de escritorio complejos. Suele ser usado como sustituto de los gestores por defecto como Kwin o Metacity.

Debido al uso intensivo de la aceleración 3D, los requerimientos necesarios son elevados respecto a otros gestores de ventanas, incluyendo hardware de aceleración 3D que soporte Xgl.

La mayoría de los efectos de Compiz (excepto transparencia y desaturación) son creados mediante añadidos externos, incluyendo el efecto más famoso, el cubo utilizado para la presentación de los diferentes escritorios virtuales que se están ejecutando.

Otra característica de Compiz es que utiliza pequeños programas llamados decoradores que dibujan los bordes de las ventanas y los botones de cierre, minimizar y maximizar, que son diferentes en función de los temas que se utilizarán.

Figura 26. Ejemplo de efectos en Compiz (Fedora)

Análisis

Una de las primeras características destacables de este gestor, es que es necesario la instalación de drivers propietarios para que se puedan ejecutar todas las características disponibles, lo cual no lo hace recomendable si no poseemos una tarjeta gráfica que podamos dedicar a la ejecución de estas capacidades.

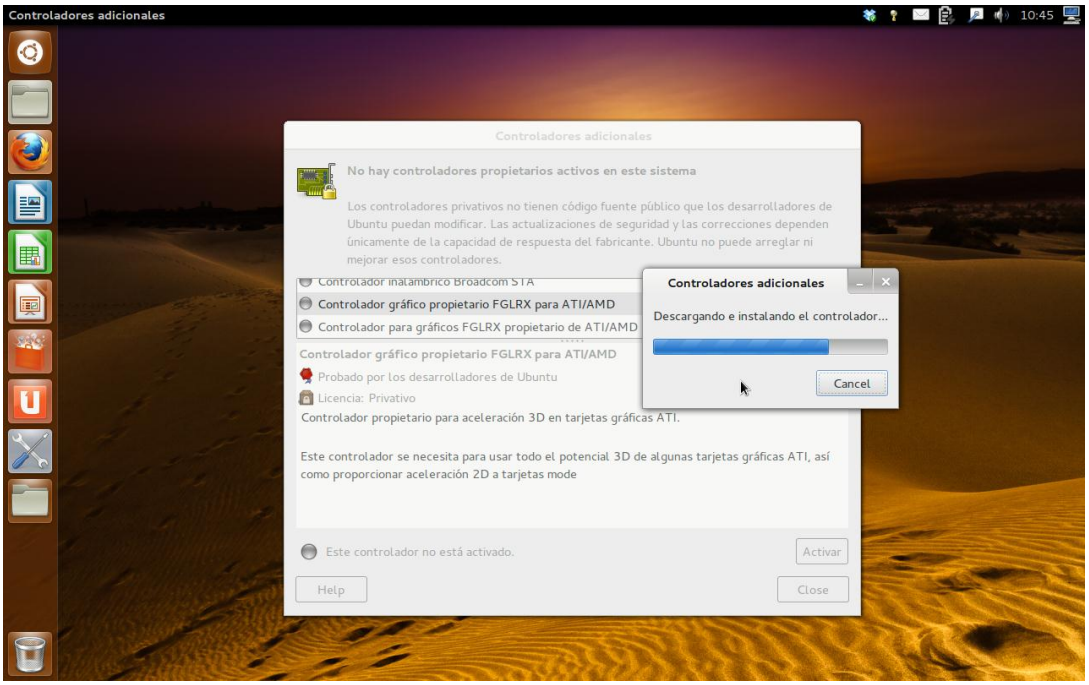


Figura 27. Instalación de drivers dedicados para la ejecución de Compiz

En la siguiente captura hacemos referencia al árbol de ejecución de procesos del sistema con el sistema en reposo. Vemos que es muy llamativo, que en comparación con el resto de gestores de ventanas analizado, el árbol que observamos posee muchos más procesos en ejecución, y que la mayoría de estos procesos hacen referencia al entorno de escritorio (donde vemos que se integra Compiz, ya que efectúa la ejecución de ciertas rutinas).

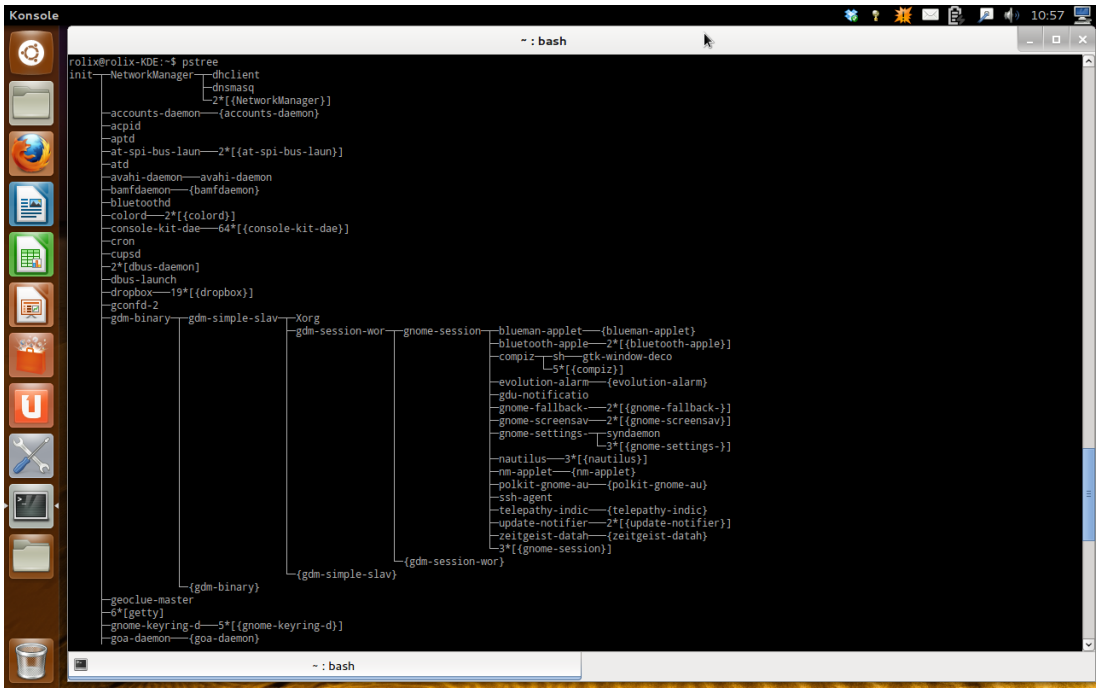


Figura 28. Pstree con Mutter y Compiz

En la siguiente captura observamos el consumo de recursos del sistema en reposo, y en particular la asignación de memoria del proceso Compiz, donde resulta llamativo su consumo de memoria (58 MB), memoria compartida (51 MB) y la memoria virtual que tiene asignada tras su uso durante una sesión normal de funcionamiento del equipo (1,4 GB).

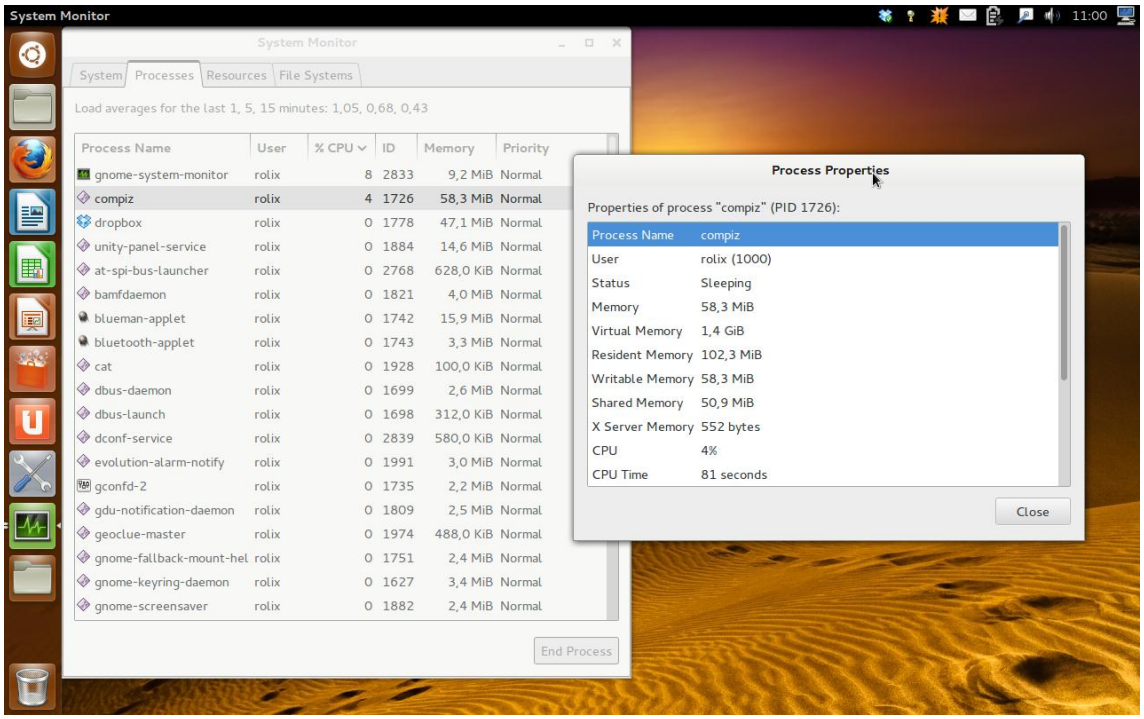


Figura 29. Monitor del sistema con detalle del proceso Compiz

Valores de ejecución de GtkConf:

```
GtkEntry - time:  0,07
GtkComboBox - time:  1,60
GtkComboBoxEntry - time:  1,52
GtkSpinButton - time:  0,25
GtkProgressBar - time:  0,11
GtkToggleButton - time:  0,21
GtkCheckButton - time:  0,15
GtkRadioButton - time:  0,31
GtkTextView - Add text - time:  0,30
GtkTextView - Scroll - time:  0,37
GtkDrawingArea - Lines - time:  1,78
GtkDrawingArea - Circles - time:  2,29
GtkDrawingArea - Text - time:  1,19
GtkDrawingArea - Pixbufs - time:  0,16
 ---
Total time: 10,32
```

Estos datos ya son lo suficientemente reveladores sobre el consumo de recursos que son necesarios para ejecutar este gestor de ventanas, lo que no lo hace recomendable para sistemas con recursos ajustados o en los que los efectos visuales no sean una prioridad. Por otro lado, la carga del sistema

durante la ejecución de las pruebas ha sido excesiva en algunos casos, llegando el proceso Compiz a mostrarnos un consumo de recursos de más del 60% de la CPU simplemente por efectuar cambios de escritorio animados, durante un breve lapso de tiempo.

4.4 Fluxbox

Fluxbox es un gestor de ventanas para X Window basado en Blackbox. Su objetivo es ser ligero y altamente personalizable, con sólo un soporte mínimo para iconos, gráficos, y sólo capacidades básicas de estilo para la interfaz.

Se utilizan atajos de teclado, tabs, y menús simples como interfaces, los cuales pueden ser editados. Algunos usuarios prefieren Fluxbox sobre otros gestores de ventanas debido a su velocidad y simplicidad.

La apariencia visual de las decoraciones de las ventanas en Fluxbox es personalizable mediante la edición de archivos de textos. Los temas de Fluxbox son compatibles con los de Blackbox los cuales se pueden editar. Se pueden especificar colores, gradientes, bordes, y otros atributos básicos de apariencia; versiones recientes de Fluxbox soportan esquinas redondeadas y elementos gráficos.

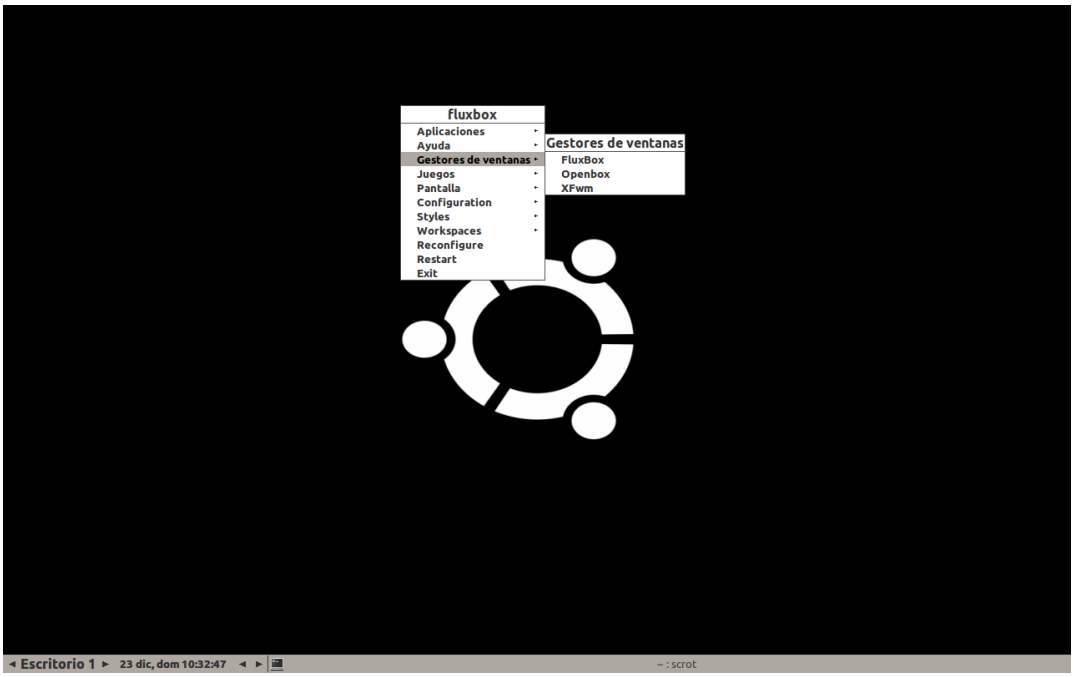


Figura 30. Escritorio fluxbox con selector de gestores

Análisis

En esta primera captura con la ejecución del comando pstree que nos muestra el árbol de procesos del sistema en ejecución, podemos observar la posición del proceso fluxbox, y su relación con el proceso kdm. A su vez, también observamos la ejecución del comando top, que nos muestra los procesos que más memoria y carga de ejecución se están llevando a cabo en el momento de su invocación, y donde tanto el proceso Xorg como el fluxbox, ocupan las primeras posiciones.

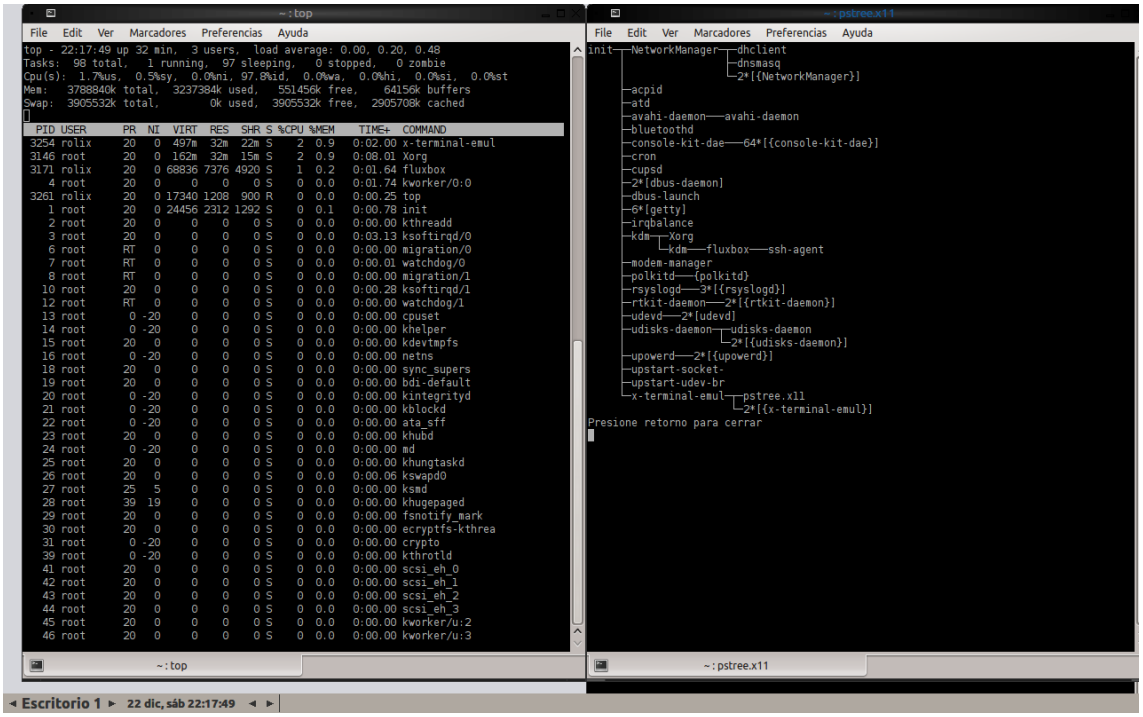


Figura 31. Top y pstree de equipo ejecutando fluxbox

En esta segunda captura, ejecutamos el monitor del sistema, donde se nos muestra que la memoria asociada al proceso fluxbox es de sólo 2,5MB, lo que nos demuestra la baja carga de recursos necesaria para la ejecución de este gestor de ventanas, mientras que a su lado tenemos un detalle del proceso, donde ya podemos observar por separado la asignación de memoria compartida (5 MB) de las librerías de fluxbox, reafirmando de nuevo que nos encontramos ante un gestor de mínimos requisitos.

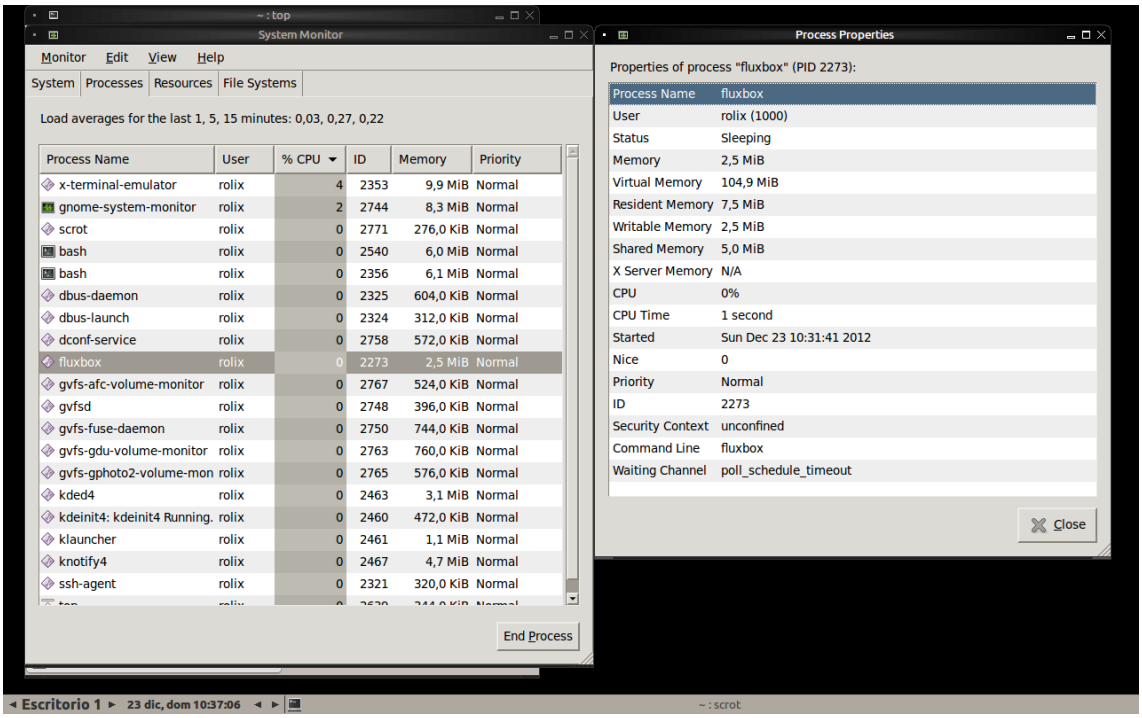


Figura 32. Monitor de sistema y detalle del proceso fluxbox

Valores de ejecución de GtkConf:

```
GtkEntry - time:  0,03
GtkComboBox - time:  0,67
GtkComboBoxEntry - time:  0,59
GtkSpinButton - time:  0,05
GtkProgressBar - time:  0,03
GtkToggleButton - time:  0,07
GtkCheckButton - time:  0,06
GtkRadioButton - time:  0,10
GtkTextView - Add text - time:  0,80
GtkTextView - Scroll - time:  0,30
GtkDrawingArea - Lines - time:  0,91
GtkDrawingArea - Circles - time:  1,30
GtkDrawingArea - Text - time:  1,89
GtkDrawingArea - Pixbufs - time:  0,25
 ---
Total time:  7,05
```

Fluxbox es recomendable para equipo con problemas de recursos, ya que nos encontramos ante un gestor que tiene una necesidad mínima tanto de memoria como de gasto de CPU. Por otro lado, sus características son limitadas, y la apariencia gráfica hace que se asemeje a un gestor anticuado.

4.5 Openbox

Openbox es un gestor de ventanas para X Window, disponible bajo licencia GPL y que cumple plenamente las especificaciones ICCCM y EWMH. Está diseñado para ser rápido y consumir una mínima cantidad de recursos. Para conseguir esa ligereza sacrifica algunas funciones típicas en buena parte de los gestores de ventanas como por ejemplo barra de menú, lista de aplicaciones en ejecución o bordes redondeados en las ventanas. Pero a cambio ofrece otras posibilidades tales como menús generados dinámicamente capaces de ofrecer información variada.

El sistema de menú de Openbox es dinámico. Esto se consigue utilizando la salida de un script como fuente del menú. Cada vez que el usuario apunte con su ratón al submenú, el script vuelve a ejecutarse y el menú es regenerado. Esta capacidad ofrece a usuarios y desarrolladores más flexibilidad que el típico menú estático presente en la mayoría de los demás gestores de ventanas.

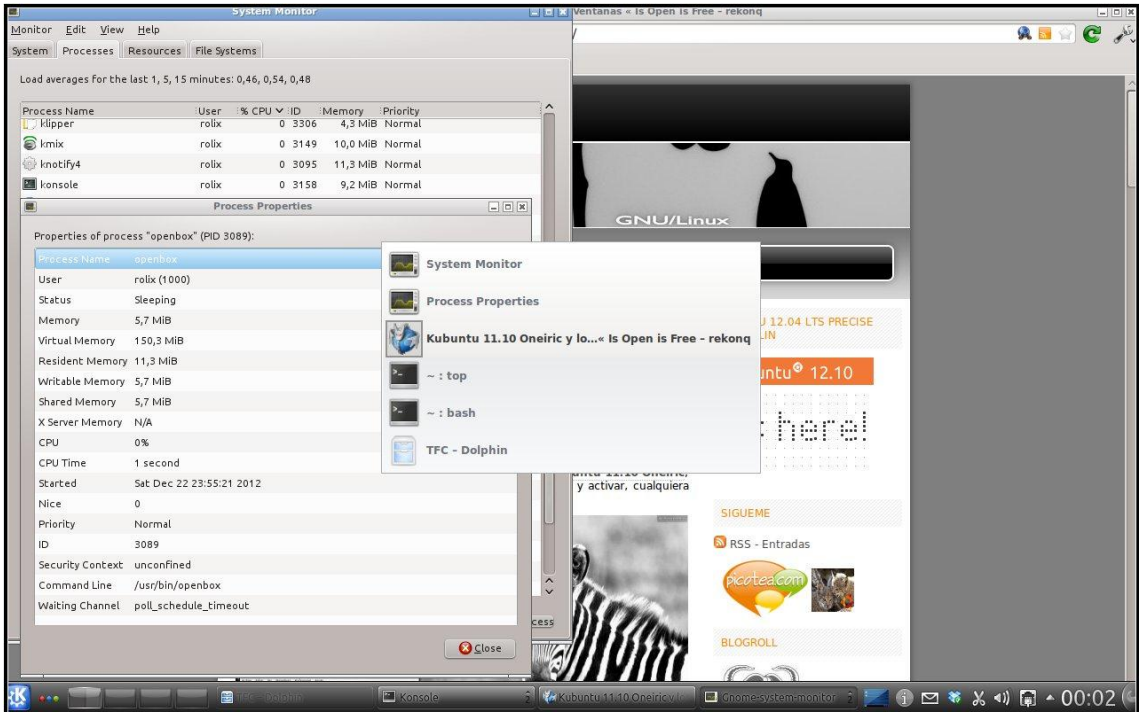


Figura 33. Escritorio ejecutando Openbox

Análisis

En esta primera captura con la ejecución del comando pstree, que nos muestra el árbol de procesos del sistema en ejecución podemos observar la posición del proceso openbox, para lograr una mejor compresión de sus características. Vemos que al contrario que la mayoría de los gestores analizados hasta ahora, el proceso openbox no tiene como padre el proceso kdm o gdm, sino que cuelga directamente de kdeinit4 y del proceso ksmserver.

Figura 34. Pstree de equipo ejecutando Openbox

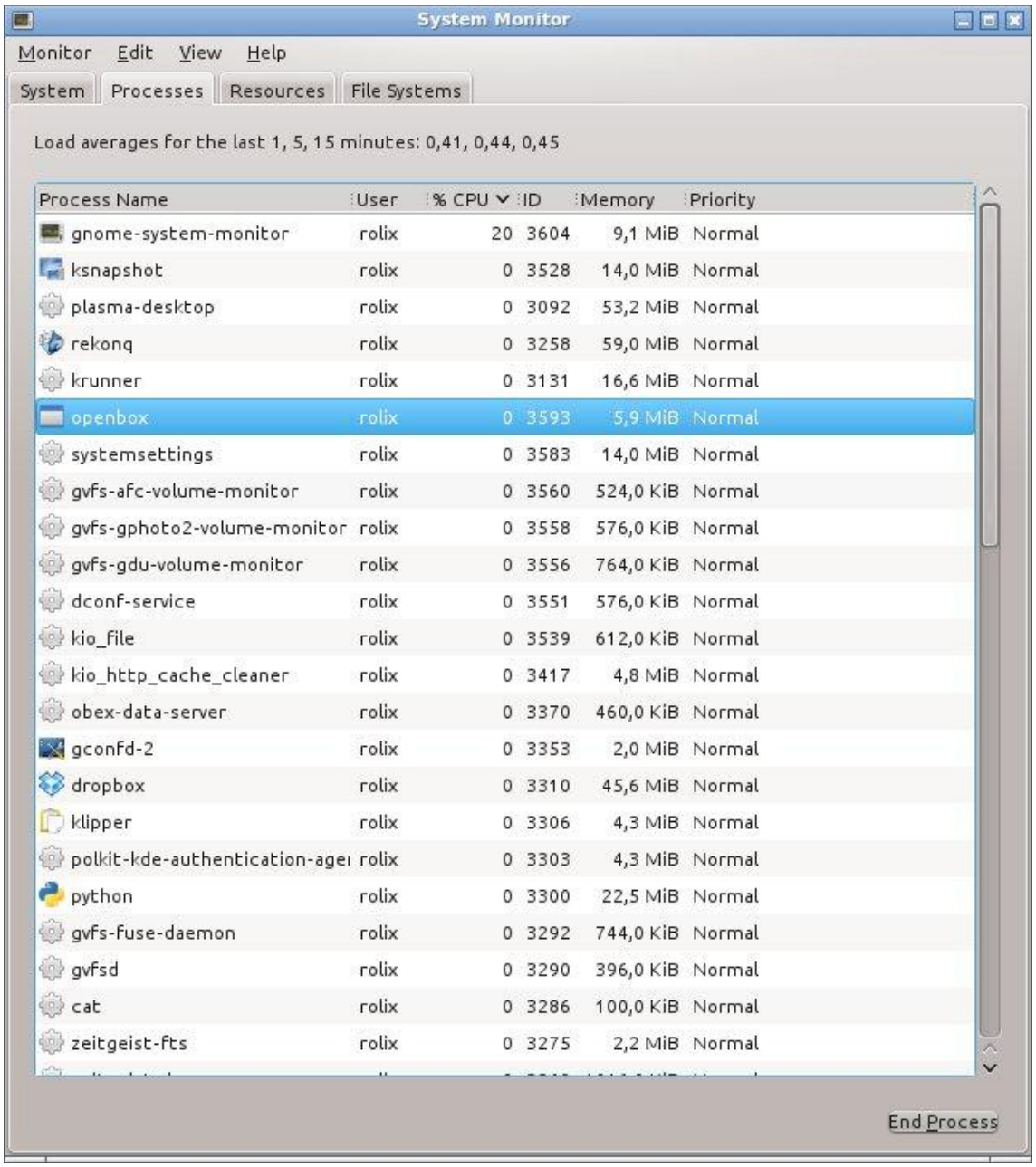


Figura 35. Monitor de sistema ejecutando Openbox

En la captura del monitor de sistema podemos confirmar que la información sobre su bajo consumo de memoria es correcta. Según los datos recogidos en nuestro equipo al momento de realizar esta prueba, el proceso openbox únicamente estaba consumiendo 5,9MB de memoria, mientras que la siguiente captura con las propiedades del proceso más concretas, no hace sino confirmar los datos obtenidos anteriormente.

Figura 36. Propiedades del proceso Openbox en ejecución.

Valores de ejecución de GtkConf:

```
GtkEntry - time:  0,03
GtkComboBox - time:  0,66
GtkComboBoxEntry - time:  0,52
GtkSpinButton - time:  0,05
GtkProgressBar - time:  0,03
GtkToggleButton - time:  0,07
GtkCheckButton - time:  0,05
GtkRadioButton - time:  0,10
GtkTextView - Add text - time:  0,81
GtkTextView - Scroll - time:  0,31
GtkDrawingArea - Lines - time:  0,89
GtkDrawingArea - Circles - time:  1,23
GtkDrawingArea - Text - time:  1,87
GtkDrawingArea - Pixbufs - time:  0,25
 ---
Total time:  6,86
```

Openbox podemos considerarlo como una evolución de fluxbox más accesible “out of the box”. El diseño no parece tan limitado como en el caso de fluxbox, y por otro lado comparte gran parte de las características desde el punto de vista de su bajo consumo de requisitos. Su funcionamiento ha sido positivo, no ha habido grandes cambios en el nivel de uso de CPU, tanto a la hora de abrir y cerrar nuevas ventanas, como realizando cambios de temas visuales. Por todo esto, Openbox se presenta como una opción totalmente recomendable, sobre todo si nos encontramos con un equipo con bajas capacidades de CPU y memoria.

4.6 Xfwm

Xfwm es el gestor de ventanas del entorno de escritorio Xfce. Este gestor sigue siendo uno de los más utilizados por los usuarios, ya que aúna características de interfaces gráficos más potentes, con un consumo moderado de recursos. Además existe una comunidad desarrolladora muy amplia detrás del proyecto que logra que este proyecto tenga un ciclo de desarrollo destacadamente dinámico.

Está basado en las bibliotecas Gnome 2.x y provee su propio gestor de composición, el primero de su tipo cuando fue lanzado. Xfwm también soporta atajos de teclado y es completamente modificable gráficamente.

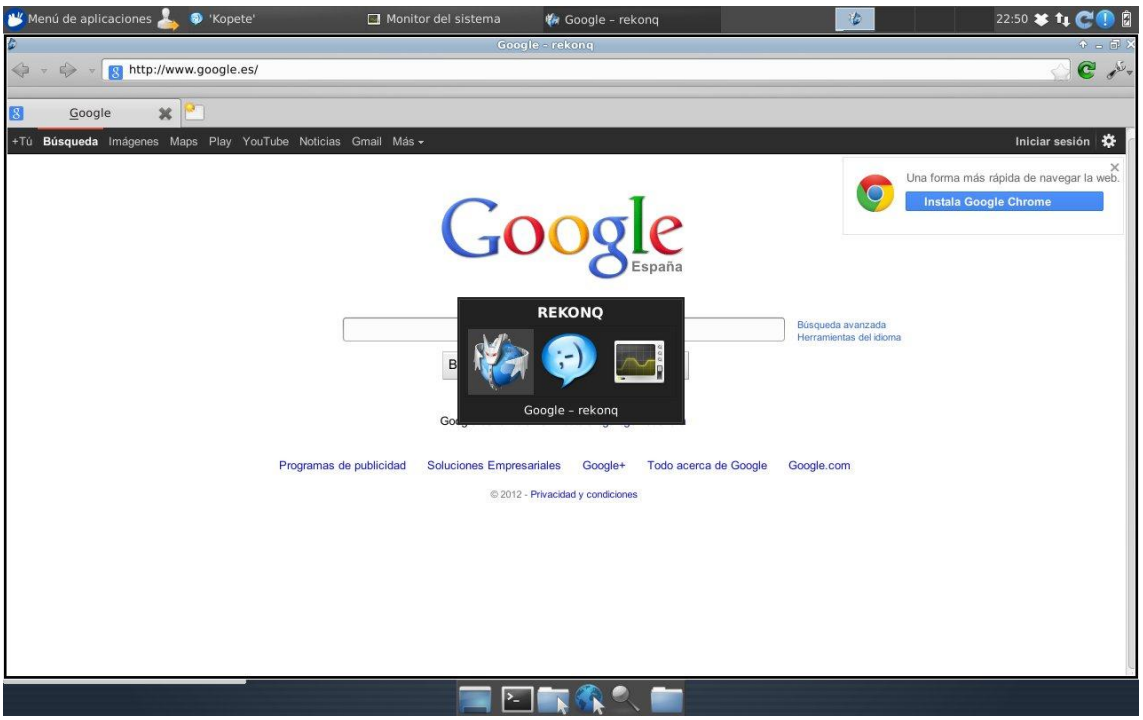


Figura 37. Escritorio ejecutando Xfwm realizando selección de escitorios virtuales.

Análisis

La ejecución de pstree nos muestra como el proceso xfwm4 (última versión de este gestor) no tiene ninguna relación con el proceso kdm que también se encuentra en ejecución en el sistema de prueba, así como el resto de procesos de xfce, en el que se pueden observar por su nombre las diferentes funciones que tienen cada uno de ellos.

Figura 38. Pstree de equipo ejecutando Xfwm

El consumo de memoria de este gestor también se encuentra entre los más moderados de los estudiados en este análisis. El proceso xfwm4 únicamente consume 2,7 MB de memoria y 9 MB si observamos la parte de memoria compartida, lo que lo hace un candidato ideal para sistemas con bajos recursos, pero que a su vez deseen usar un gestor con capacidades gráficas aceptables.

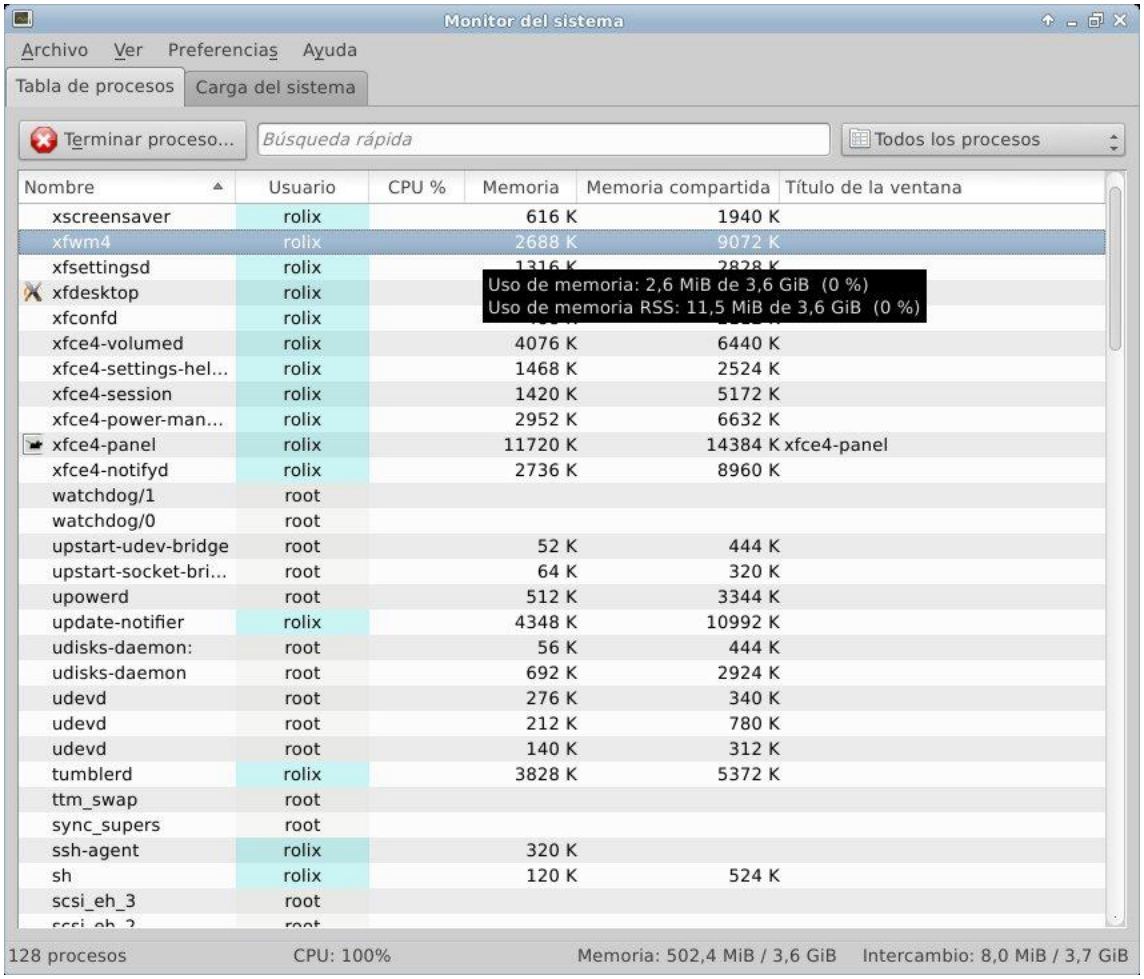


Figura 39. Monitor del sistema de equipo ejecutando Xfwm

Valores de ejecución de GtkConf:

```
GtkEntry - time:  0,04
GtkComboBox - time:  0,93
GtkComboBoxEntry - time:  0,74
GtkSpinButton - time:  0,15
GtkProgressBar - time:  0,10
GtkToggleButton - time:  0,17
GtkCheckButton - time:  0,12
GtkRadioButton - time:  0,21
GtkTextView - Add text - time:  0,81
GtkTextView - Scroll - time:  0,37
GtkDrawingArea - Lines - time:  0,90
GtkDrawingArea - Circles - time:  1,36
GtkDrawingArea - Text - time:  1,78
```

```
GtkDrawingArea - Pixbufs - time:  0,25
 ---
Total time:  7,93
```

Xfwm es un gestor que aúna las dos características más destacadas de los gestores de ventanas. Por un lado, su consumo es ajustado, y no se ha producido en ningún momento picos significativos de recursos, mientras que a su vez el diseño de efectos y del propio escritorio es fácilmente configurable y adaptado a un diseño moderno.

5 Conclusiones

La idea principal de este proyecto ha sido establecer una base sobre el conocimiento del mundo de los gestores de ventanas de GNU/Linux, desde sus inicios hasta las distintas opciones que se nos ofrecen actualmente.

El objetivo ha sido estudiar la evolución de los gestores desde su base, mostrando primero una aproximación al concepto de gestor de ventanas y sus características más destacadas, para después continuar con el sistema que ha hecho desarrollar estas aplicaciones hasta lo que conocemos actualmente, el sistema X Window, estudiando su funcionamiento y mostrando hacia dónde se tiende a evolucionar este sistema.

Posteriormente se han mostrado varias opciones actuales de gestores de ventanas, algunas muy usadas, ya que acompañan a los entornos de escritorios más utilizados, y también otras alternativas menos conocidas, pero que con la característica diversidad del mundo GNU/Linux, ayudan a que cualquier funcionalidad pueda quedar cubierta de un modo apropiado.

Kwin y Mutter, si contamos con un equipo lo suficientemente potente, muestran lo que un gestor de ventanas moderno puede llegar a hacer. Sus características visuales y su integración con los entornos de escritorios actuales así lo demuestra, pero desde el punto de vista de consumo de recursos, vemos que son aplicaciones pesadas y que pueden llegar a ralentizar un equipo, por lo que no las recomiendo en un sistema en que la calidad visual no sea un imperativo.

En el lado opuesto de este estudio, nos encontraríamos con Fluxbox y Openbox. Ambas opciones son recomendables si estamos usando un equipo en el que los recursos sean limitados, o si queremos que el gasto asociado a los gestores de ventanas sea el mínimo posible. En el caso de tener que escoger entre uno de los dos, me decantaría por Openbox, ya que las capacidades que nos muestra no son tan limitadas como en el caso de Fluxbox, y sin embargo su consumo es prácticamente el mismo en ambos casos, mostrándose así como la opción más recomendable para este tipo de situaciones.

Enlightenment y Xfwm son las dos opciones intermedias de este estudio. Ambas nos dan opciones gráficas interesantes (sin llegar a las opciones más “extremas” quep odemos observar en Kwin o Mutter), pero mantienen un compromiso más que interesante con el consumo de recursos tal y como se puede observar en la tabla adjunta a continuación. Xfwm me parece la opción más completa de las dos, ya que aunque ambas opciones se mantienen bastante parejas en cuanto a consumo y capacidades, aporta el desarrollo de una aplicación como Xfce, entorno de escritorio consolidado, con un desarrollo muy estable y que se ha convertido en la imprescindible de muchas distribuciones GNU/Linux.

La ganadora de esta comparativa de gestores es Xfwm, por su condición de usabilidad para todas las opciones. Su consumo de memoria ha sido incluso

inferior a opciones más ligeras como Fluxbox, los resultados que han arrojado los tests ejecutado por Gtkperf lo sitúan en valores similares a las opciones más ligeras, pero a su vez aporta características gráficas que sin llegar a ser tan impactantes como las que muestran los gestores más potentes, sí que mantienen opciones visuales agradables. Por otro lado es destacable que su ciclo de desarrollo sea estable y dinámico, junto con una comunidad de uso amplia, lo que le garantiza una evolución continuada.

Tabla comparativa de los gestores analizados

	Kwin	Enlightenment	Mutter	Fluxbox	Openbox	Xfwm
GtkEntry	0,21	0,03	0,07	0,03	0,03	0,04
GtkComboBox	2,55	0,7	1,6	0,67	0,66	0,93
GtkCombo Entry	2,78	0,6	1,52	0,59	0,52	0,74
GtkSpinButton	0,6	0,06	0,25	0,05	0,05	0,15
GtkProgressBar	0,4	0,03	0,11	0,03	0,03	0,1
GtkToggleButton	0,85	0,07	0,21	0,07	0,07	0,17
GtkCheckButton	0,84	0,06	0,15	0,06	0,05	0,12
GtkRadioButton	0,89	0,1	0,31	0,1	0,1	0,21
GtkTextView Text	2,45	0,82	0,3	0,8	0,81	0,81
GtkTextView Scroll	2,4	0,3	0,37	0,3	0,31	0,37
GtkDrawing Lines	0,91	0,92	1,78	0,91	0,89	0,9
GtkDrawing Circles	1,32	1,25	2,29	1,3	1,23	1,36
GtkDrawing Text	1,57	1,9	1,19	1,89	1,87	1,78
GtkDrawing Pix	0,24	0,26	0,16	0,25	0,25	0,25
GtkConf Total	18,01**	7,11	10,32	7,05	6,86	7.93
Memoria proceso	53,3MB	20,8MB	58MB	2,5MB	5,8MB	2,6MB
Memoria compartida	39MB	12,6MB	50,9MB	5MB	N/A	9MB

** Hay que tener en cuenta que los datos obtenidos en este test se han basado en la ejecución de rutinas específicas del framework Gtk, distintas a las Qt usadas por Kwin.

6 Bibliografía

http://www.x.org/wiki/

http://www.icccm.org/

http://qt-project.org/

http://www.gtk.org/

http://freedesktop.org/wiki/

http://wayland.freedesktop.org/

http://www.**microxwin**.com

http://insitu.lri.fr/metisse/

http://www.xynth.org/

http://directfb.org/

http://userbase.kde.org/KWin/es

http://www.enlightenment.org/

https://live.gnome.org/Clutter

http://www.compiz.org/

http://fluxbox.org/

http://openbox.org/

http://wiki.xfce.org/

http://www.kde.org/

http://www.wikipedia.org/

http://www.ubuntu-guia.com

http://distrowatch.com/

http://www.kubuntu.org/

http://gtkperf.sourceforge.net/