


Universidad de las Ciencias Informáticas

Facultad 10

GUANO, ENTORNO DE ESCRITORIO CUBANO, LIBRE Y DE CÓDIGO ABIERTO

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autores: Adisleydis Rodríguez Pino
Abel Fírvida Donéstevez

Tutor: Lic. Jose Jorge Lorenzo Vila.

Ciudad de la Habana
Mayo 2009


DECLARACIÓN DE AUTORÍA

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

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

Adisleydis Rodríguez Pino	Abel Alfonso Fírvida Donéstevez	Jose Jorge Lorenzo Vila
_____	_____	_____
Firma del Autor	Firma del Autor	Firma del Tutor

AGRADECIMIENTOS.

A la Revolución que nos ha permitido estudiar en este lugar tan lleno de oportunidades.

A las personas que han contribuido con su esfuerzo a la realización de este trabajo.

A los familiares y amigos que nos han ayudado a palear estos años.

AGRADECIMIENTOS.

A mi mamita chula y linda y a mi papa que siempre han estado a mi lado ayudándome en todo momento y dándome todo su amor; a ellos les debo todo lo que soy.

A mi hermano celoso grandotote (jeje) por ser tan especial y quererme tanto como yo a el.

A mi enana linda por ser la mejor sobrina del mundo.

A mis abuelos, tíos y primos por apoyarme tanto en todo momento.

A mi bito por ser la personita que robo mi corazón; gracias por apoyarme y enseñarme tanto en todos los momentos que hemos estado juntos; sabes que este trabajo tiene parte de ti.

A mi grupo x04 por tantos momentos de alegría.

A mis amigas locas; Yuli, Baby y Vivi, por compartir tantos momentos de felicidad y de tristezas durante todo este tiempo y por ser tan jodedoras y tan feas todas las mañanas (jejej).

A mi tutor por su tiempo y dedicación.

A mi compañero de tesis, gracias por enseñarme tanto durante todo este tiempo y por ser como eres.

A todos los muchachos del proyecto Nova, especialmente a Roman, Miguel, Felix, Micha, Vladimir, Angel, Miranda y Anielkis.

A todas mis amistades, especialmente a Yusi, Yeilan, Angel y Geykel por su apoyo y continua preocupación.

A todos que me han formado durante estos años.

A todos los que se preocuparon por este trabajo y pusieron su granito de arena.

Gracias

Adis.

A todas las personas que me han ayudado a ser quien soy ..

Gracias

Abel

A nuestros familiares.

RESUMEN

La migración de los sistemas informáticos cubanos que actualmente funcionan sobre plataformas privativas, a sistemas operativos libres y de código abierto, es uno de los pasos más importantes que dará el país en los próximos años, en pos de alcanzar la necesaria independencia tecnológica.

En este trabajo se presenta Guano que es una propuesta de entorno de escritorio que integra la usabilidad y funcionalidad de entornos de escritorios de la familia Windows y Linux. Con él se pretende dar solución a la necesidad de llevar la migración a Software Libre en Cuba al hardware de bajas prestaciones, brindando la posibilidad de extender la vida útil de las máquinas, lo cual puede aportar beneficios al proceso de informatización de la sociedad cubana.

ÍNDICE

ÍNDICE.

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El software libre está conquistando América. La oleada de triunfos de los partidos de la izquierda se está traduciendo, en el plano tecnológico, en la sustitución de los programas informáticos privativos por otros de código abierto. Los gobernantes en América Latina creen que el camino más viable para impulsar la sociedad de la información como palanca para el desarrollo consiste en usar tecnologías de Software Libre y de Código Abierto (Free and Open Source Software, FOSS, por sus siglas en inglés).

En este marco, Cuba se ha planteado con valentía el objetivo de comenzar a transitar por el camino de la informatización, diseñando e iniciando la aplicación de estrategias que permitan convertir los conocimientos y las Tecnologías de la Información y las Comunicaciones (TIC) en instrumentos a disposición del avance del proceso de transformaciones emprendido por la Revolución Socialista. El país se encuentra trabajando para organizar la migración progresiva a software libre de las entidades y Organismos de la Administración Central del Estado (OACE), para ello se ha creado el grupo de trabajo para el ordenamiento del tema, que es coordinado por la Universidad de las Ciencias Informáticas (UCI) y la Oficina para la Informatización. En la UCI se elabora, a su vez, la distribución de Linux cubana (Nova) para contribuir y apoyar la migración a tecnologías de Software Libre y la guía de migración a sistemas de tipo FOSS, entre otras acciones para minimizar la dependencia tecnológica (1).

La migración de la informática en Cuba a sistemas operativos de tipo FOSS, proporcionará aumentar la productividad y la agilidad para responder a cambios en el mercado. Permitirá reducir la dependencia, tanto en el plano económico-comercial como en el de la seguridad, del software privativo sobre el cual se articulan en el país transacciones y desarrollos de inmensos volúmenes de información de gran valor. Se mantendrá una solución efectiva en costos y se ofrecerá mejores servicios o productos. Además, se amortiguará el costo asociado a la compra de licencias para el uso del software privativo que se utiliza actualmente en el país, el cual en su gran mayoría, debido al bloqueo económico impuesto por Estados Unidos a la isla, se ejecuta de manera pirata o ilegal y el pago por este concepto asciende a cientos de millones de dólares (Ver **Anexo 1**).

La necesidad de independencia tecnológica cubana se enmarca en un contexto regido mundialmente por las normas capitalistas de producción y mercado, donde los grandes monopolios invierten en ideas innovadoras cada día y la competencia obliga a tener siempre alguna novedad a la venta. Esto ha traído como consecuencia que la capacidad de procesamiento de las computadoras se haya

incrementado en un 400 %[1] en solo cinco años y por tanto que en los países altamente desarrollados se descarten grandes cantidades de máquinas, con solo uno o dos años de uso (2). Paralelamente la industria del software, siguiendo los hitos del mercado de hardware, aumenta cada vez más los requerimientos de los programas -el FOSS no está exento de esto- provocando que para tener acceso a las últimas tecnologías de software sea necesario disponer de los elementos de hardware más avanzados del mercado.

Desde el año 2007 la Empresa Industrial para la Informática, las Comunicaciones y la Electrónica (GEDEME), en coordinación con entidades comercializadoras de equipos de cómputo como COPEXTEL, desarrolla un programa de reciclaje de computadoras en Cuba. El proyecto pretende agrupar aquellas computadoras en estado de obsolescencia y tratar de alargarles su vida útil (3). Este programa puede aportar beneficios al proceso de informatización de la sociedad cubana, sin embargo la mayoría de estas máquinas al ser reincorporadas a la sociedad caen en las mismas tendencias preestablecidas de instalación y uso de sistemas operativos y software privativo. Esto en gran medida, es debido a la inexistencia de un sistema operativo libre que aúne buen rendimiento en hardware de bajas prestaciones y las funcionalidades necesarias para la interacción con el usuario final. El desarrollo eficiente de este programa nacional permitiría reinsertar estas computadoras de bajas prestaciones a la sociedad, logrando que puedan ser utilizadas durante mucho más tiempo. Además implicaría para el país un ahorro sustancial en gastos por nuevas adquisiciones. Sin embargo, la solución sería mucho más completa si se reincorporaran estas computadoras personales (PC, por sus siglas en inglés) con sistemas operativos libres, contribuyendo así a extender el proceso de migración de software libre en Cuba y con esto la independencia tecnológica.

El mundo del FOSS posee varias soluciones para acrecentar el rendimiento de los sistemas operativos sobre hardware obsoleto, las cuales se conocen como entornos de escritorio ligero (Lightweight Desktop Environment, LDE). Los LDE están creados con el fin de consumir poca cantidad de recursos de un computador, pero estos, en la mayoría de los casos, logran un buen desempeño eliminando de su diseño funcionalidades que son muy necesarias para la interacción con un usuario final como pudiera ser una secretaria, un contador, un maestro o cualquier persona que inicia su experiencia frente a una PC.

De la combinación de los factores anteriormente expuestos se puede inferir la siguiente situación problémica: La ejecución del proceso de migración de la totalidad de los sistemas informáticos cubanos a tecnologías de Software Libre y de Código Abierto, como parte de la estrategia del

[1] Este comportamiento se designa como Ley de Moore y se expresa en los siguientes términos: La capacidad de proceso de los equipos se duplica cada dos años y medio y los costos se mantienen. O en su forma equivalente, misma capacidad por la mitad de precio al término de 2 y medio de años (32).

Gobierno Cubano para lograr la necesaria independencia tecnológica, puede verse afectada en una parte considerable del equipamiento cubano de bajas prestaciones, debido a la inexistencia de entornos de escritorio ligero que aúne buen rendimiento en hardware de bajas prestaciones y las funcionalidades necesarias para la interacción con el usuario final.

A partir de ésta situación problémica surge el siguiente problema científico: ¿Cómo construir un entorno de escritorio ligero que mantenga un buen desempeño en computadoras de bajas prestaciones técnicas y las funcionalidades necesarias para la interacción con el usuario final?

El objeto de estudio de este trabajo son los entornos de escritorio y como objetivo general se propone desarrollar un entorno de escritorio ligero de código libre, orientado a computadoras de bajas prestaciones técnicas, que sea amigable al usuario final, para extender la vida útil de las computadoras en Cuba, favoreciendo la migración a Software Libre en la isla y con ella a la independencia tecnológica. Para dar cumplimiento a este objetivo se definió como campo de acción a los entornos de escritorio ligero más usados que existen en el marco del Software Libre y los elementos y herramientas que estos contienen.

La idea a defender que se plantea este trabajo:

El desarrollo de un entorno de escritorio ligero orientado al usuario final contribuye a facilitar la migración a Software Libre en Cuba al hardware de bajas prestaciones, extendiendo la vida útil de este equipamiento y el ahorro consecuente de recursos financieros.

Para dar cumplimiento al objetivo general se desarrollarán los siguientes objetivos específicos:

- Seleccionar la metodología adecuada para el desarrollo del entorno de escritorio ligero y las tecnologías aplicables a los principios de construcción de este.
- Desarrollar el entorno de escritorio ligero con las funcionalidades y facilidades de uso necesarias para la interacción con el usuario final.
- Realizar pruebas que garanticen la calidad del entorno de escritorio.

Para el cumplimiento de los objetivos descritos anteriormente se han definido las siguientes tareas de investigación:

- Estudiar la evolución y tendencias de los entornos de escritorio ligero y las herramientas que se puedan adaptar a la propuesta de solución.
- Seleccionar la metodología y los lenguajes de programación, para guiar y llevar a cabo el desarrollo de la propuesta de solución.
- Realizar el proceso de Ingeniería de Software al entorno de escritorio ligero.

Para el desarrollo de esta investigación se utilizaron diferentes métodos y técnicas los cuales se describen a continuación.

Histórico - Lógico: Mediante este método se pudo percibir la evolución de los DE (ligeros y no), las metodologías y los lenguajes, así como las tendencias de sus desarrollos en el tiempo. Lo cual facilitó la selección de las herramientas que conforman la propuesta de solución.

Analítico - Sintético: El uso de este método permitió el análisis de los diferentes entornos de escritorio ligero, para así poderlos descomponer en partes y entender mejor sus funcionalidades y características. La síntesis viabilizó el descubrimiento de las múltiples relaciones que guardan entre sí los componentes de los distintos entornos de escritorio ligero, posibilitando su unión en la propuesta de solución.

Análisis bibliográfico: A través de este método se escrutó gran cantidad de bibliografía en formato duro y digital que permitieron obtener elementos que apoyaron el desarrollo de esta investigación.

Observación: Mediante este método se pudo percibir de manera directa los problemas existentes en la migración de los sistemas informáticos cubanos a plataformas FOSS y las deficiencias funcionales de los entornos de escritorio ligero. Ayudando así a obtener una mejor visión del problema y los objetivos de esta investigación.

La solución que se propone desarrollar tiene como antecedentes a los proyectos de creación de escritorios ligeros y gestores de ventanas para PC de bajas prestaciones, por ejemplo: XFCE, LXDE, IceWM, Fluxbox, Epiewm, Blackbox, Enlightement, Openbox además de, los proyectos de humanización de la informática como GNOME. Este proyecto pretende ser una combinación de ellos, siendo así una solución que permita a computadoras de bajos recursos integrarse a la migración nacional a sistemas FOSS.

Aportes prácticos esperados del trabajo:

- Obtención de componentes reutilizable para cualquier entorno de escritorio.
- Versión inicial del producto.

Posibles aportes sociales de ser aplicada la solución propuesta:

- Aminorar las barreras técnicas que dificultan la independencia tecnológica.
- Reducir el impacto del bloqueo y la crisis sistémica mundial durante la generalización del proceso de informatización de la sociedad cubana.
- Mejorar la usabilidad de las TIC (Tecnologías de la Informática y las Comunicaciones) sobre Software Libre.

- Facilitar la adaptación de los usuarios a los sistemas de tipo FOSS.

El trabajo de diploma está estructurado en tres capítulos organizados de la siguiente manera:

Capítulo 1. Fundamentación teórica del tema: En este capítulo se realiza un estudio del estado del arte sobre los entornos de escritorio ligero, esto comprende el análisis de algunos de los entornos de escritorio que están en explotación actualmente, pudiendo ser soluciones a tener en cuenta, por lo que se deja plasmado el por qué no se consideran soluciones factibles para el problema. Además, se hace un estudio de las tecnologías y metodologías de desarrollo de software, seleccionando el método más viable para emplear en la construcción del entorno de escritorio ligero.

Capitulo 2. **Descripción y desarrollo ágil de la propuesta de solución:** En este capítulo se inicia el desarrollo de la propuesta de solución haciendo uso de la metodología seleccionada, para ello se hace hincapié en sus primeras fases, recogiéndose los principales artefactos generados de estas.

Capítulo 3. Validación de la solución propuesta: En este capítulo se describen algunos casos de pruebas o pruebas de aceptación que se le han efectuado a Guano en cada una de las iteraciones, para comprobar el funcionamiento de acuerdo a las historias de usuario. Además se exponen una relación de las funcionalidades con las que cuenta el DE hasta la fecha.

1

CAPÍTULO

1. FUNDAMENTACIÓN DEL TEMA

En este capítulo se realiza un estudio del estado del arte sobre los entornos de escritorio ligero, esto comprende el análisis de algunos de los entornos de escritorio (Desktop Environment, DE) que están en explotación actualmente, pudiendo ser soluciones a tener en cuenta, por lo que se deja plasmado el por qué no se consideran soluciones factibles para el problema. Además, se hace un estudio de las tecnologías y metodologías de desarrollo de software, seleccionando el método más viable para emplear en la construcción del LDE.

1.1. Entorno de Escritorio.

En su comienzo las computadoras no tenían pantalla, sus resultados eran plasmados en tarjetas de papel perforadas y luego hojas impresas en tinta. El uso de monitores ya fue todo un avance, pero durante bastante tiempo se limitaron a ofrecer sólo caracteres y acceso por teclado. Hubo un tiempo en que el primer trazo de una circunferencia en pantalla fue todo un acontecimiento. La metáfora del escritorio fue técnicamente posible a mediados de los años setenta gracias a los avances realizados en la creación de una interfaz gráfica de usuario (denominada con frecuencia GUI, del inglés Graphical User Interface). Hoy en día es una realidad evidente que la mayoría de los usuarios están acostumbrados a interactuar en la pantalla del monitor con ventanas, íconos, menús y el puntero del ratón, los cuáles conforman elementos de una abstracción. El escritorio informático es una consolidada ficción que los programadores crearon para facilitar la interacción con el usuario final, la cual evoca a los escritorios reales, con su superficie bidimensional y rectangular equivalente a la mesa, una foto de fondo, una papelera, notas, carpetas de archivos y documentos visualizados como hojas de papel.

Así surgieron los Entornos de Escritorio, conceptualizados como un conjunto de software orientado a ofrecer al usuario de un ordenador una interacción amigable y cómoda con el sistema operativo (4).

En Linux el entorno gráfico en sí, es la suma de un servidor, llamado X11 y los clientes (o componentes) que se conectan a él, los cuales se encargan de dar un ambiente amigable para el uso del sistema operativo. Los clientes para el servidor gráfico X11 que forman un escritorio completo

son, entre otros, los controladores de ventanas (Window-manager), los manejadores de ficheros (File-manager) y los escritorios (Desktop) (5); los cuales interactuando de forma coordinada entre sí y con el servidor X11 agrupan las funcionalidades generales del DE.

Se puede puntualizar entonces que en los sistemas operativos libres el DE no es un programa o componente sencillo, sino la composición integrada de un conjunto de estos que, debido a su interoperabilidad, ofrecen al usuario todas las herramientas necesarias para poder desenvolverse en el entorno gráfico.

A continuación se hace mención de algunas de las estructuras y componentes fundamentales de los DE en el marco de los sistemas operativos libres.

1.1.1. Aspectos principales de un Entorno de Escritorio.

Términos como escritorio, íconos, carpetas, ventanas, barras de tareas o menús son muy corrientes en los DE. A pesar de las diferentes nomenclaturas e incluso de las variaciones de funcionalidades, las partes fundamentales de cualquier DE se pueden resumir de manera general como se muestra a continuación.

- Íconos: En entornos gráficos, pequeña imagen gráfica mostrada en la pantalla que representa un objeto manipulable por el usuario. Por ejemplo, una papelera representa un comando para borrar textos o archivos no deseados. Los íconos permiten controlar ciertas funciones de la computadora sin tener que recordar comandos ni escribirlos con el teclado. Son un elemento importante de las interfaces gráficas de usuario, ya que facilitan el manejo de las distintas funciones (6).
- Ventana: Sector de la imagen en la pantalla del monitor que muestra distintas posibilidades de operar u opciones, el desarrollo de ciertos procesos, etcétera. Su uso permite superponerlas para ver aquello que se desea, manteniendo a la vez otras ventanas en superposición o en otros sectores de la pantalla. El sistema de ventanas ha proporcionado al usuario una mayor facilidad y rapidez de uso al no tener que recurrir a comandos difíciles de recordar, puesto que se transmiten las órdenes a través de íconos (7). Por tanto en informática, una ventana es un área visual, normalmente de forma rectangular, que contiene algún tipo de interfaz de usuario, mostrando la salida y permitiendo la entrada de datos para uno de varios procesos que se ejecutan simultáneamente.
- Menús: Es el catálogo o relación de programas y procedimientos que aparece en pantalla con el fin de que, usando un teclado, un dispositivo táctil, un lápiz óptico o un ratón, el operador pueda elegir qué opción desea ejecutar. Los denominados «desplegables» son el tipo más

habitual en la actualidad, muestran un Menú Principal y, tras una sencilla operación, sus diferentes opciones (7). De acuerdo a la definición en informática un menú es una serie de opciones que el usuario puede elegir para realizar determinada tarea.

En la Figura 1 se muestra la representación gráfica de algunos de los elementos antes mencionados.

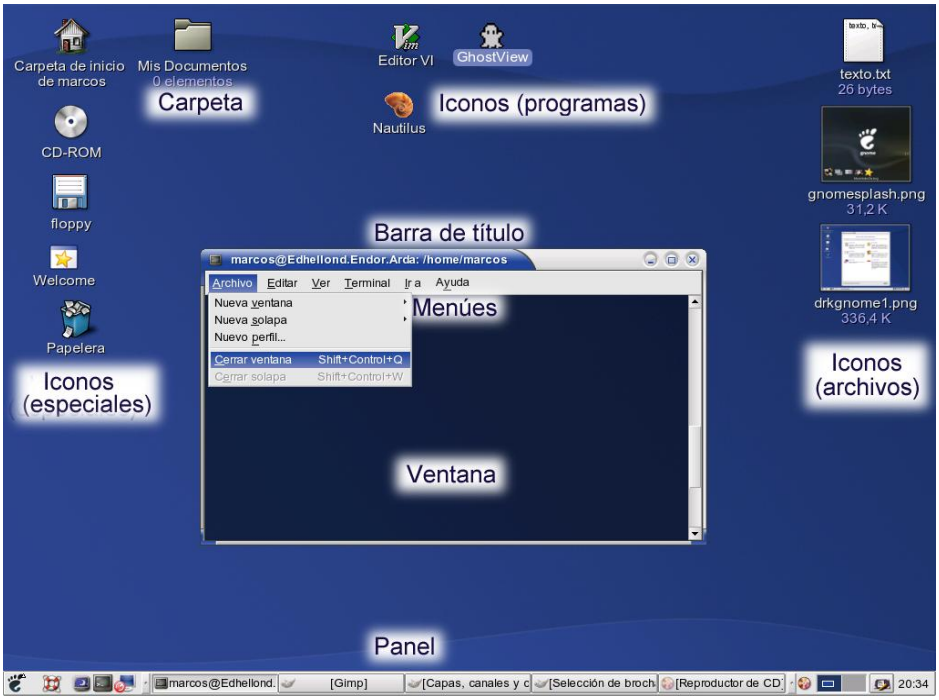


Figura 1: Partes más importantes de un Entorno de Escritorio.

Según la concepción de DE en los sistemas Linux, además de los elementos antes descritos, estos están compuestos por un conjunto de componentes que agrupan diferentes funcionalidades. A pesar de que este conjunto tiende a variar para cada DE en particular, existe sin embargo, un grupo constante que conforman el núcleo básico de la mayoría de ellos. Un resumen de estos componentes fundamentales se lista a continuación:

- **Manejador de Ficheros**: Es el encargado de explorar las carpetas, relacionar ficheros con programas para su reproducción, gestionar los eventos de arrastrar y soltar íconos además de administrar la papelera de reciclaje.
- **Manejador de ventanas**: Es un programa que controla la ubicación y apariencia de las aplicaciones bajo el sistema X Windows (8).
- **Panel**: Es el encargado de suministrar un menú de aplicaciones, un espacio para el reloj, los íconos de notificaciones, accesos directos a aplicaciones, el paginador de escritorio, applets para el control de volumen, el acceso a red y la importantísima barra de tareas donde se

sitúan las aplicaciones minimizadas.

- **Gestor de sesiones**: Encargado de salir de la sesión, hibernar, suspender, cambiar de usuario, apagar y reiniciar.

Un aspecto a considerar en el estudio de los DE es su rendimiento en PC de bajas prestaciones. Sobre todo si se tiene en cuenta de q el desarrollo de software ha seguido el ritmo de evolución tecnológica del hardware que se experimenta mundialmente en la actualidad y que los sistemas FOSS no han quedado exentos de esto. Esto ha repercutido en que el aspecto rendimiento sea rara vez tenido en cuenta o que el proceso de maximizarlo ha ido en detrimento de algunas funcionalidades básicas. Dentro de los sistemas operativos libres los LDE han constituido precisamente soluciones dirigidas a responder a estas dificultades.

A continuación se hace un estudio comparativo de las soluciones principales que existen en el marco del Software Libre y que pudieran dar solución a los aspectos de este trabajo.

1.2. Entornos de Escritorios Ligero.

Los LDE son una combinación de aplicaciones livianas que posibilitan la interacción del usuario con el sistema operativo y están diseñados para consumir poca cantidad de memoria y microprocesador. Ellos son usados frecuentemente por desarrolladores o personas con cierta experiencia en el trabajo con sistemas operativos Linux pues en la mayoría de los casos no brindan una vía fácil para su configuración, incluso muchos no cuentan con opciones de personalización como establecer imágenes de fondo o íconos. Un análisis de aquellos que se consideran importantes en el marco de este trabajo se muestra a continuación.

- **XFCE**

XFCE (X Free Choresterol Environment) es un entorno de escritorio ligero para sistemas de tipo Unix. Su creador, Olivier Fourdan, dice de él: «Diseñado para la productividad, se carga y ejecuta aplicaciones rápido, mientras conserva recursos de sistema» (9). XFCE además de ser ligero es estable, modular, visualmente atractivo y fácil de configurar. Este entorno de escritorio consume pocos recursos del sistema, por lo que es adecuado para computadoras con un mínimo recursos de hardware, como memoria RAM o CPU. También provee un Framework para el desarrollo de aplicaciones. XFCE está basado en GTK+. Cuenta con un manejador de ventana, un panel, un manejador de escritorio, un manejador de sesiones, un gestor de archivos, además de un conjunto de aplicaciones. Sin embargo, carece de algunas funcionalidades como por ejemplo, la posibilidad de establecer conexiones con otras PC de la red, además de poder compartir carpetas e impresoras o navegar y explorar distintos protocolos de red con el mismo explorador.

- **LXDE**

LXDE (Lightweight X11 Desktop Environment) es un entorno de escritorio rápido y ligero para equipos de bajos recursos. A diferencia de otros ambientes de escritorio, los componentes no se integran firmemente. Al contrario, los componentes son independientes, y cada uno de ellos se puede utilizar por separado con muy pocas dependencias (10). Esta desarrollado en el lenguaje C, utilizando la librería GTK+. Este escritorio está conformado por un panel, un manejador de ficheros, un manejador de sesiones, un gestor de ventanas y algunas aplicaciones que le aportan funcionalidades. Aún así, al igual que XFCE carece de la posibilidad de establecer conexiones con otras PC de la red, de poder compartir carpetas e impresoras o navegar y explorar distintos protocolos de red con el mismo explorador. Además, no posee soporte para la interacción con una papelera de reciclaje y una herramienta intuitiva que permita configurar el escritorio.

- **Fluxbox**

Es un gestor de ventanas (Windows Manager, WM por sus siglas en inglés) para el Sistema X Window basado en Blackbox 0.61.1. Su objetivo es ser ligero y altamente personalizable, con sólo un soporte mínimo para íconos, gráficos, y sólo capacidades básicas de estilo para la interfaz (11). Se considera un entorno de escritorio minimalista con el mínimo apoyo gráfico y con sólo una barra de tareas en todo el escritorio, que solamente tiene un reloj, un área para las notificaciones y una lista de aplicaciones. Es un WM que no soporta la arquitectura de componentes y carece de un explorador de ficheros.

- **Enlightenment**

Es un entorno de escritorio que une la rapidez y la ligereza de un gestor de ventanas con efectos visuales muy atractivos y la potencia de un completo entorno de escritorio (12). Uno de sus objetivos es llegar a ser un entorno de escritorio completo. Es muy configurable, modular y muy atractivo visualmente.

Fluxbox y Enlightenment son muy simples debido a que poseen funcionalidades básicas dirigidas a desarrolladores o a usuarios que poseen conocimiento medio de informática. En diferentes fuentes bibliográficas a ambos los describen como gestores de ventanas, aunque como proyecto tienen la visión de ser LDE, a pesar de todo esto, se analizan en este trabajo como DE debido a la estética minimalista que poseen, lo que hace posible convertirlos en entornos de trabajo muy eficientes.

De manera general existen diferentes proyectos libres dirigidos a proveer soluciones de LDE. Sin embargo, en su mayoría constituyen soluciones parciales y no se tienen referencias de alguna que

asuma la integración de todos los componentes y las funcionalidades necesarias desde el punto de vista de un usuario final (Ver **Anexo 3**). Por tanto se hace necesario el estudio de componentes independientes que a partir de su integración brinden las bases para el desarrollo de un LDE.

1.2.1. Componentes de Entornos de Escritorio Libres.

Dentro del marco del FOSS se conocen numerosos proyectos destinados a proveer distintos componentes, que integrados de forma coordinada permiten obtener una plataforma para el desarrollo de un LDE. A continuación se analizan en tres grupos (Manejadores de Ficheros, Paneles y Gestores de ventanas) los posibles elementos que conformarán la propuesta de solución.

1.2.1.1. Manejadores de Ficheros.

Uno de los elementos más importantes en la construcción de un DE es la concepción de un manejador de ficheros. Existe gran cantidad de estos en el mundo del Software Libre, la mayoría de los que poseen más potencia y funcionalidades ya pertenecen a un DE específico lo cual hace muy difícil importarlos como módulos para un entorno propio, ejemplo de esto son: Nautilus del proyecto Gnome, Konqueror o Dolphin del proyecto KDE, Rox perteneciente a Rox y Thunar de XFCE.

A continuación se muestran un conjunto de manejadores de ficheros que son independientes a cualquier DE.

PCManFM

PCMan File Manager (PCmanFM) es un administrador de ficheros, desarrollado por Hong Jen de Taiwán. Es el administrador estándar de LXDE, que también es desarrollado por el mismo autor en conjunto con otros desarrolladores. PCmanFM pretende seguir las especificaciones proporcionadas por FreeDesktop.org para interoperabilidad en el software libre.

Algunas de las funcionalidades que proporciona PCmanFM en su última versión estable son:

- 3 modos de vista (Íconos, Lista Compacta y Lista Detallada).
- Soporte de miniaturas de imágenes.
- Soporte de Manejo de dispositivos a través de hal.
- Interfaz de búsqueda de archivos.
- Exploración con pestañas al estilo Firefox.
- Arrastrar y Soltar.
- Se pueden arrastrar y soltar los archivos entre las diferentes pestañas de exploración.

- Soporte de Marcadores.
- Cumple el estándar Free Desktop.
- Soporte de asociaciones de archivos.
- Carga rápida de directorios de gran tamaño.
- Interfaz GTK2+.
- Internacionalización (idioma español incluido).
- Manejo de escritorio.
- Asignación de aplicaciones para determinados archivos.
- Posibilidad de abrir una sesión como root desde el propio navegador (13).

Sin embargo este carece de algunas funcionalidades como permitir a un usuario navegar y explorar carpetas a través de distintos protocolos de red, compartir carpetas e impresoras en la red e interactuar con la papelera de reciclaje.

Krusader

Krusader es un administrador de archivos avanzado para el escritorio KDE. Se caracteriza por el uso de dos paneles, como Midnight Commander. Las principales cualidades de Krusader son una interfaz gráfica e intuitiva, soporte para archivos comprimidos, sistemas de archivo montados, FTP, módulo de búsqueda avanzada, visor/editor, sincronización de directorios, comparación de contenido de archivos, renombrado recursivo de archivos y todo totalmente configurable.

El ambiente natural de Krusader es KDE, pero no necesita de este para funcionar en una PC porque se sostiene en servicios proporcionados por las librerías KDE. Esto significa que Krusader puede ejecutarse en Gnome, AfterStep, XFce u otro gestor de ventanas, siempre y cuando se cuente con las librerías necesarias en el sistema (14). Aún así este manejador de ficheros carece de algunas funcionalidades, como el manejo del escritorio, compartir carpetas en la red o proveer soporte para la papelera de reciclaje, además este requiere gran cantidad de dependencias de KDE.

XFE

Xfe es un gestor de archivos gráfico para el sistema X Window para Unix y sistemas similares a Unix, escrito por Roland Baudin. Sus cualidades son la simplicidad, ligereza y facilidad de uso (15). Este provee funcionalidades como la posibilidad de interactuar con los ficheros a través de accesos rápidos de teclado, también brinda una papelera de reciclaje, un visor de imágenes, un editor de texto integrado, un filtro de carpetas y la funcionalidad de buscar contenido dentro de estas. Al mismo

tiempo este gestor de ficheros, no permite manejar el escritorio, ni interactuar con carpetas a través de la red o con dispositivos extraíbles.

1.2.1.2. Gestores de ventanas.

Otro elemento importante del DE es el manejador de ventanas, hoy en día existen muchos de ellos, algunos bastante minimalistas que resultan muy ligeros, prácticos y potentes pero rara vez resultan totalmente configurables. A continuación se analizan algunos de los gestores de ventanas de uso actual.

OPenbox

Openbox está diseñado para ser rápido y consumir una mínima cantidad de recursos. Ofrece posibilidades tales como menús generados dinámicamente capaces de ofrecer información variada. Posee la peculiaridad de permitir cambiar casi todos los aspectos de cómo interactuar con el escritorio y completamente inventar nuevas formas de uso y control, es altamente configurable.

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 estético presente en la mayoría de los demás gestores de ventanas. Openbox permite al usuario disponer de un menú raíz sobre su escritorio al pulsar el botón derecho del ratón, y permite configurar el modo en el que las ventanas son gestionadas, además posee una aplicación para la configuración gráfica llamada obconf (16).

Twm

Twm es un administrador minimalista de ventanas, el cual proporciona el conjunto de herramientas básicas de cualquier gestor de ventanas. Este puede ser usado bien sea de forma independiente o con un entorno de escritorio y es instalado como parte del lanzamiento X11R7.1[2] (17).

1.2.1.3. Paneles.

A diferencia de los manejadores de ficheros, no existen en el mundo del FOSS paneles independientes al DE. A pesar de esto, se hizo un análisis de los paneles que forman parte de los escritorios más importantes en los sistemas operativos libres.

LXPanel (panel de LXDE)

LXPanel es un clon del panel de escritorio Fbpanel en el que se introducen mejoras en la generación

[2] X11R7.1 es el primer lanzamiento que aprovecha la modularización del sistema de ventanas X. Esta modularización, que divide X en módulos lógicamente distintos, permite que los desarrolladores contribuyan al sistema de una manera más fácil.

automática del menú en función de los archivos Desktop que se tenga en el sistema. Se añade otra aplicación denominada lxpanelctl mediante la cual se puede controlar las funciones de LXPanel a través de línea de comandos y por último se añaden dos plugins nuevos, Netstatus y Volume (18). Por tanto es un panel de escritorio fácil de usar y con todas las funciones que pueden esperarse de un panel. La configuración se realiza a través de una interfaz gráfica de usuario (GUI), está desarrollado en el lenguaje C y actualmente se encuentra en su versión 0.4 beta.

XFCE4-panel (panel de XFCE)

El panel es parte del entorno de escritorio de Xfce y ofrece lanzadores de programa, menús de panel, un reloj, un intercambiador de escritorio y mucho más (9). Muchos aspectos del panel se pueden configurar a través de diálogos gráficos.

Kicker (panel de KDE)

Kicker es el panel que permite iniciar las aplicaciones del entorno de escritorio KDE. De manera predeterminada, aparecerá a lo largo de la parte inferior de la pantalla, y ocupará todo el ancho de la pantalla, pero esto es configurable. Kicker es el punto de partida para (¡casi!) todo aquello a lo que se quiera acceder de forma rápida (19).

Según la experiencia que se ha alcanzado con el uso de este panel, los autores de este trabajo concluyen que Kicker es modular y se compone de los siguientes elementos: los applets, los botones de aplicaciones y los botones especiales. De forma predeterminada, el panel contiene los siguientes subprogramas: Paginador - muestra los escritorios virtuales, la barra de tareas, la bandeja del sistema, el reloj, y los siguientes botones: menú de KDE, botón de minimizar todo.

GNOME-panel

Gnome-panel es el lanzador y barra de tareas de GNOME. Forma parte del núcleo por defecto del escritorio GNOME. Este panel puede ser poblado con otros menús y botones completamente personalizables, incluidos los nuevos menús, cuadros de búsqueda, y los íconos, con los íconos en particular (llamados lanzadores) se desempeñan funciones similares a la de la “barra de acceso rápido” que se encuentra en la barra de tareas de Microsoft Windows. Posee gran cantidad de complementos que hacen de este un modelo atractivo y con muchas funcionalidades.

1.2.1.4. Selección de los componentes para el LDE.

Como conclusión del estudio del estado del arte sobre los LDE en el marco del FOSS se seleccionaron algunos componentes, cuyas características y futuras modificaciones conducirán a satisfacer el objetivo general de esta investigación; ellos son:

- PCmanFM, el cual fue seleccionado para el manejo de ficheros. Siendo este una alternativa atractiva de arquitectura ligera que además posee la funcionalidad de administrar el escritorio, lo cual brinda un ahorro de tiempo por concepto de análisis e implementación de herramientas que se encarguen de esta función.
- Openbox como gestor de ventanas, debido a que a pesar de ser ligero posee una muy amigable e intuitiva forma de configuración.
- XFCE4-panel como panel, debido a que puede ser incorporado al proyecto con un mínimo de dependencias. Este panel posee gran cantidad de complementos desarrollados por la comunidad, que le darían un valor agregado al escritorio, además de brindar un framework que agilizaría la programación de nuevas funcionalidades.

Para guiar la integración y desarrollo de estos componentes de una forma organizada y coherente es necesario el estudio de metodologías de desarrollo de software que se exponen a continuación.

1.3. Metodologías y tecnologías actuales en el campo del desarrollo de software.

1.3.1. Metodologías de desarrollo de software.

Las metodologías de desarrollo de software surgieron a raíz de la necesidad de controlar y documentar proyectos cada vez más complejos, impulsadas principalmente por instituciones económicamente importantes, con requisitos sumamente estrictos de seguridad y fiabilidad en sistemas.

Aplicar una metodología de desarrollo de software en una organización es una tarea difícil y compleja. El éxito en su utilización depende de múltiples factores, antes de decidirse por una en función de sus características se debe reflexionar acerca de sus aspectos determinantes, así como las ventajas sobre las otras.

Ventajas de tener una metodología:

Mejora de los procesos de desarrollo.

- Todas las personas del proyecto trabajan bajo un marco común.
- Estandarización de conceptos, actividades y nomenclaturas.
- Actividades de desarrollo apoyadas por procedimientos y guías.
- Resultados del desarrollo predecibles.
- Uso de herramientas de Ingeniería Software.

- Planificación de actividades en base a un conjunto de tareas definidas y a la experiencia en otros proyectos.
- Recopilación de mejores prácticas para proyectos futuros.

Mejora de los productos

- Se asegura que los productos cumplen con los objetivos de calidad propuestos.
- Detección temprana de errores.
- Se garantiza la trazabilidad de los productos a lo largo del desarrollo.

Mejora de las relaciones con el cliente

- El cliente percibe el orden en nuestros procesos.
- Facilita al cliente el seguimiento de la evolución del proyecto.
- Se establecen mecanismos para asegurar que los productos desarrollados cumplen con las expectativas del cliente (20).

Hoy en día existen en el mundo diferentes propuestas metodológicas para llevar a cabo el desarrollo de proyectos. Se identifican en la actualidad la existencia de dos clasificaciones, ambas contribuyen a un buen y organizado desarrollo de software aunque tengan sus marcadas diferencias, estas son:

- Las metodologías tradicionales, haciendo énfasis en la planeación.
- Las metodologías ágiles, haciendo énfasis en la adaptabilidad del proceso.

Las metodologías de desarrollo tradicionales modelan tres aspectos o dimensiones de los sistemas informáticos: estructura de la información, funciones y comportamiento. Cada aspecto requiere distintos modelos y los documentos resultantes pueden ser múltiples en los últimos dos aspectos; y, en general, son bastante costosos de mantener. En el desarrollo de sistemas pequeños o de mediana complejidad, la adopción de una metodología formal se percibe como un obstáculo más que como una ventaja puesto que las ventajas sólo se hacen notorias a largo plazo, y los objetivos primarios de cualquier institución van dirigidos a producir software en el menor tiempo y costo posibles (21).

Los métodos ágiles, nacen como respuesta a los problemas detallados anteriormente y se basan en dos aspectos puntuales, el retrasar las decisiones y la planificación adaptativa; permitiendo potenciar aún más el desarrollo de software a gran escala.

Como resultado de esta nueva teoría se crea el Manifiesto Ágil cuyas principales ideas son:

- Los individuos y las interacciones entre ellos son más importantes que las herramientas y los procesos empleados.
- Es más importante crear un producto software que funcione que escribir documentación exhaustiva.
- La colaboración con el cliente debe prevalecer sobre la negociación de contratos
- La capacidad de respuesta ante un cambio es más importante que el seguimiento estricto de un plan (22).

Estas sentencias se traducen en ventajas como son:

- La motivación y cohesión de los equipos de desarrollo aumenta.
- Los equipos de desarrollo aumentan la productividad.
- Los clientes siempre quedan satisfechos.
- El desarrollo es más adaptable a los cambios repentinos.
- Están pensadas para proyectos pequeños, reducidos de tiempo, requisitos volátiles y/o basados en nuevas tecnologías.

Debido a las grandes ventajas que proporcionan utilizar metodologías ágiles se propone como alternativa más viable el uso de estas para el desarrollo de este trabajo. A continuación se analizan algunos aspectos de las más usadas.

1.3.2. Metodologías Ágiles a usar.

Las metodologías de desarrollo de software ágiles permiten a los pequeños grupos de desarrollo concentrarse en la tarea de construir software fomentando prácticas de fácil adopción y un entorno ordenado que ayude a que las personas trabajen mejor y permita que los proyectos finalicen exitosamente.

Entre las metodologías más conocidas se encuentran:

- XP (Programación Extrema).
- Scrum.
- Crystal Metodologies.
- MSF (Microsoft Solution Framework).

A continuación se hace un análisis de dos de las metodologías antes mencionadas, XP y Scrum.

Programación Extrema (XP).

XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico (23).

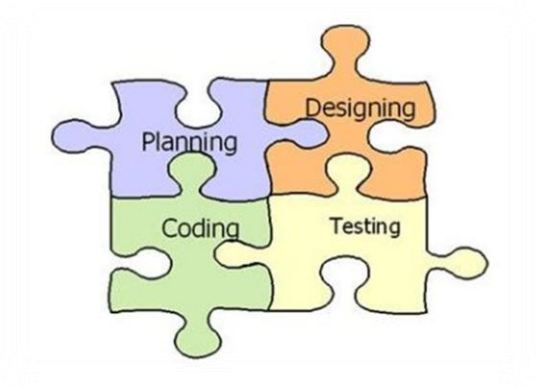


Figura 2: Metodología Extreme Programing.

A continuación se presentan las características esenciales de XP organizadas en los tres apartados siguientes: historias de usuario, roles, proceso y prácticas.

Historias de usuario

Son la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas. Las historias de usuario son descompuestas en tareas de programación (task card) y asignadas a los programadores para ser implementadas durante una iteración.

Roles XP.

Los roles de acuerdo con la propuesta original de Beck son:

- Programador. El programador escribe las pruebas unitarias y produce el código del sistema.

- Cliente. Escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio.
- Encargado de pruebas (Tester). Ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.
- Encargado de seguimiento (Tracker). Proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar futuras estimaciones. Realiza el seguimiento del progreso de cada iteración.
- Entrenador (Coach). Es responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente.
- Consultor. Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas.
- Gestor (Big boss). Es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es de coordinación.

Proceso y Prácticas XP.

El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.

2. El programador estima el esfuerzo necesario para su implementación.

3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.

4. El programador construye ese valor de negocio.

5. Vuelve al paso 1.

La principal suposición que se realiza en XP es la posibilidad de disminuir la mítica curva exponencial del costo del cambio a lo largo del proyecto, lo suficiente para que el diseño evolutivo función. Esto se consigue gracias a las tecnologías disponibles para ayudar en el desarrollo de software y a la aplicación disciplinada de las siguientes prácticas (24).

El juego de Planeamiento: Rápidamente determinar el alcance del próximo release mediante la combinación de prioridades del negocio y estimaciones técnicas. A medida que la realidad va cambiando el plan, actualizar el mismo.

Pequeños Releases: Poner un sistema simple en producción rápidamente, luego liberar nuevas versiones en ciclos muy cortos.

Metáfora: Guiar todo el desarrollo con una historia simple y compartida de cómo funciona todo el sistema.

Diseño Simple: El sistema deberá ser diseñado tan simple como sea posible en cada momento. Complejidad extra es removida apenas es descubierta.

Testing: Los programadores continuamente escriben pruebas unitarias, las cuales deben correr sin problemas para que el desarrollo continúe. Los clientes escriben pruebas demostrando que las funcionalidades están terminadas.

Refactoring: Los programadores reestructuran el sistema sin cambiar su comportamiento para remover duplicación, mejorar la comunicación, simplificar, o añadir flexibilidad.

Programación de a Pares: Todo el código de producción es escrito por dos programadores en una máquina.

Propiedad Colectiva del Código: Cualquiera puede cambiar código en cualquier parte del sistema en cualquier momento.

Integración Continua: Integrar y hacer builds del sistema varias veces por día, cada vez que una tarea se completa.

Semana de 40-horas: Trabajar no más de 40 horas semanales como regla. Nunca trabajar horas extras durante dos semanas consecutivas.

Cliente en el lugar de Desarrollo: Incluir un cliente real en el equipo, disponible de forma full-time para responder preguntas.

Estándares de Codificación: Los programadores escriben todo el código de acuerdo con reglas que enfatizan la comunicación a través del mismo (25).

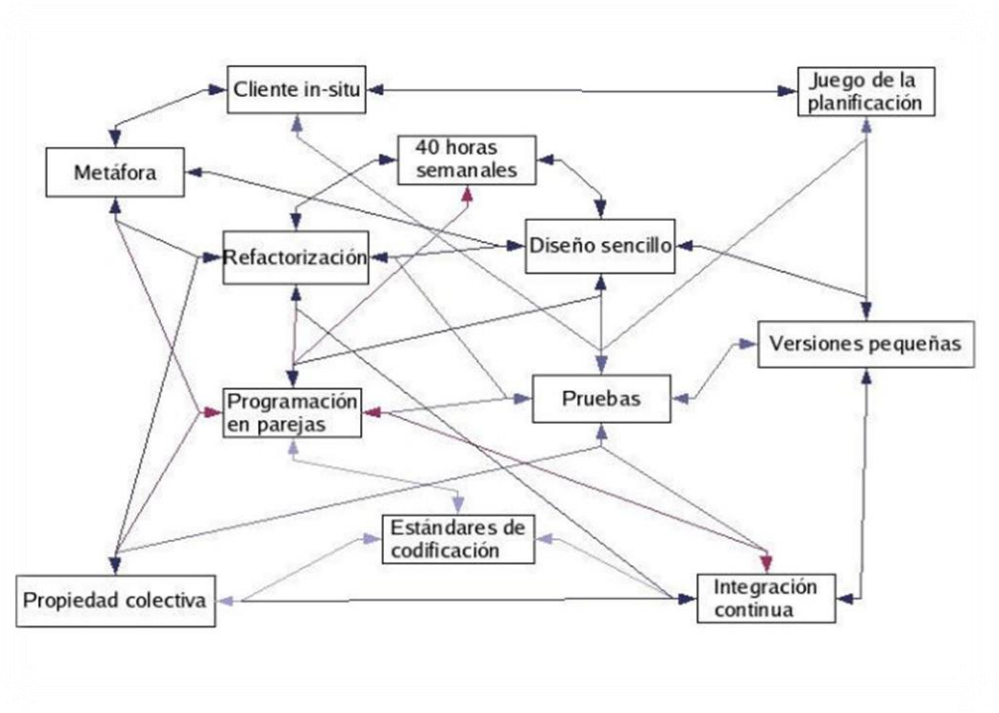


Figura 3: Las prácticas se refuerzan entre sí.

Scrum.

Desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente. La segunda característica importante de esta metodología, son las reuniones durante el período de vida del proyecto, destacándose la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración (24).

Scrum no está definida como metodología independiente, sino como un complemento a otras metodologías como lo son XP y MSF. Como método, Scrum enfatiza valores y prácticas de gestión, sin pronunciarse sobre requerimientos, implementación y demás cuestiones técnicas; de allí su deliberada insuficiencia y su complementariedad.

1.3.3. Justificación de la selección de la metodología: SXP.

Analizando las características propias de las metodologías expuestas anteriormente, y considerando

que el proyecto está integrado por un número reducido de estudiantes, se decidió establecer el uso de metodologías ágiles de desarrollo, específicamente las metodologías SCRUM, para la parte de planificación, y XP, para la parte de desarrollo, como alternativa más viable a la propuesta de solución. La unión de estas dos metodologías se conoce como SXP, metodología propuesta en el 2008 por la ingeniera Gladys Marsi Peñalver Romero; aprobada y ampliamente aplicada en los proyectos que trabajan con Software Libre en la Facultad 10 de la UCI.

Con la utilización de SCRUM para la gestión, se logra una planificación y organización inigualable; mientras que XP respalda con sus prácticas todo el proceso de desarrollo, obteniéndose de esta forma un proceso de software completo (26).

Actualmente no existe una metodología universal para hacer frente con éxito a cualquier proyecto de desarrollo de software. Toda metodología debe ser adaptada al contexto del proyecto, por tanto SXP se va a aplicar de manera dinámica durante el ciclo de vida del LDE porque se adapta a las condiciones reales del desarrollo del mismo. A continuación se hace una breve descripción de esta metodología.

SXP, como marco de apoyo.

SXP está especialmente indicada para proyectos de pequeños equipos de trabajo, rápido cambio de requisitos o requisitos imprecisos, muy cambiantes, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo claro, permitiendo además seguir de forma clara el avance de las tareas a realizar, de forma que los jefes pueden ver día a día cómo progresa el trabajo.

Fases definidas por la metodología:

- Planificación-Definición: Se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.
- Desarrollo: Se realiza la implementación del sistema hasta que esté listo para ser entregado.
- Entrega: Puesta en marcha.
- Mantenimiento: Fase donde se realiza el soporte para el cliente (26).

1.3.4. Lenguajes de programación.

A continuación se hace un análisis de algunos de los lenguajes de programación más utilizados en el desarrollo de componentes (herramientas o aplicaciones de escritorio) que integran a un DE.

ANSI C

ANSI C, o simplemente C, es un lenguaje de programación creado en 1972 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL. Al igual que B, es un lenguaje orientado a la implementación de Sistemas Operativos, concretamente Unix.

C es un lenguaje de programación de nivel medio ya que combina los elementos del lenguaje de alto nivel con la funcionalidad del ensamblador, pero es muy versátil y eficiente, que revolucionó las técnicas y estilo de programación. Su característica principal es ser portable, es decir, es posible adaptar los programas escritos para un tipo de computadora en otra. Otra de sus características principales es el ser estructurado, es decir, el programa se divide en módulos (funciones) independientes entre sí.

Sin embargo, la popularidad, eficacia y potencia de C se ha producido porque este lenguaje no está prácticamente asociado a ningún sistema operativo, ni a ninguna máquina en especial. Esta es la razón fundamental por la que C es conocido como el lenguaje de programación de sistemas por excelencia.

Es un lenguaje de programación ordenado, potente y muy fácil de adaptar a diferentes arquitecturas.

Características de C.

- Orientado a la programación de sistemas.
- Es altamente transportable.
- Es muy flexible.
- Genera código muy eficiente.
- Es muy expresivo (se pueden realizar muchas funciones escribiendo pocas líneas de código).
- Es muy poco modular.
- Hace pocas comprobaciones.
- Da poca disciplina al programador.
- Es difícil leer código escrito por otras personas (27).

C++

C++ es un potente lenguaje de programación que apareció en 1980. La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitan la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido.

C++ abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos. Las principales características radican en las facilidades que proporciona para la programación orientada a objetos y para el uso de plantillas o programación genérica.

Posee además una serie de propiedades difíciles de encontrar en otros lenguajes de alto nivel:

- Posibilidad de redefinir los operadores (sobrecarga de operadores).
- Identificación de tipos en tiempo de ejecución (RTTI).

En la actualidad, el C++ es un lenguaje versátil, potente y general. Su éxito entre los programadores profesionales le ha llevado a ocupar el primer puesto como herramienta de desarrollo de aplicaciones. El C++ mantiene las ventajas del C en cuanto a riqueza de operadores y expresiones, flexibilidad y eficiencia. Además, ha eliminado algunas de las dificultades y limitaciones del C original (28).

Python

Python es un lenguaje de programación poderoso y fácil de aprender. Ofrece eficaces estructuras de datos de alto nivel y un simple, pero efectivo, acercamiento a la programación orientada a objetos. La sintaxis elegante de Python y su escritura dinámica, junto con su naturaleza interpretada, lo convierte en un lenguaje ideal para el scripting y el desarrollo rápido de aplicaciones en muchas áreas en la mayoría de plataformas (29). Este lenguaje incluye una gran colección de módulos estándar que se pueden utilizar como base de los programas que proporcionan funcionalidades para el manejo de ficheros, llamadas al sistema, sockets y hasta interfaces a librerías gráficas como (GTK, QT, TK).

Python es un lenguaje de programación multiparadigma. Esto significa que más que forzar a los programadores a adoptar un estilo particular de programación, permite varios estilos: programación orientada a objetos, programación estructurada y programación funcional. Una característica importante de Python es la resolución dinámica de nombres, lo que enlaza un método y un nombre de variable durante la ejecución del programa (también llamado ligadura dinámica de métodos) (30).

Bash como lenguaje interpretado

Bash (Bourne Again Shell) es el intérprete de comandos más ampliamente difundido para la serie de sistemas operativos GNU y muchos otros sistemas tipo UNIX, deriva de Korn Shel y se ha convertido en un estándar de facto, incorporando la experiencia acumulada durante décadas por numerosos intérpretes desarrollados para sistemas tipo UNIX.

Bash es un intérprete de lenguaje de comandos compatible con sh que ejecuta comandos desde la entrada estándard o de un archivo. Bash también incorpora características útiles de los shells Korn y C (ksh y csh). El intérprete de comandos Bash es una herramienta poderosa, pensada para un uso

intensivo en la administración y operación habitual de los sistemas operativos tipo UNIX, incluyendo GNU/Linux (31) .

1.3.4.1. Justificación de la selección de los lenguajes.

Como base para el desarrollo del LDE se seleccionaron un conjunto de componentes, los cuales están escritos en código C. Debido a esto, para mantener una alta compatibilidad y facilidad de reutilización en las nuevas funcionalidades se hizo necesario el uso de este lenguaje. Por otro lado, para la realización de scripts que no demandaran de mucha capacidad de procesamiento, se seleccionó el lenguaje Bash, el cual está diseñado para realizar actividades y aplicaciones cotidianas y simples.

1.4. Conclusiones.

El estudio de los componentes de los entornos de escritorio, las metodologías y lenguajes de programación que se utilizan en el desarrollo de software, fueron necesarios para decidir cuáles son los más propicios a ser usados para el desarrollo de la propuesta de solución. Finalmente se decidieron utilizar los componentes: PCmanFM, Openbox y XFCE4-Panel, además de, los lenguajes C y Bash.

El no partir de cero en los componentes que integran el LDE, brinda la posibilidad de poder implementar nuevas funcionalidades según los requerimientos de los usuarios. Atendiendo a esto las modificaciones que se realizarán a los proyectos originales se plantean siguiendo dos estándares: las especificaciones de Freedesktop y la HIG, las cuales marcan las pautas de un diseño amigable e intuitivo para las interfaces gráficas.

Para guiar el proceso ingenieril, se seleccionó la metodología ágil de desarrollo SXP, recogiéndose a continuación los principales artefactos generados en cada una de sus iteraciones.

2

CAPÍTULO

2. DESCRIPCIÓN Y DESARROLLO ÁGIL DE LA PROPUESTA DE SOLUCIÓN.

En este capítulo se inicia el desarrollo de la propuesta de solución haciendo uso de la metodología seleccionada, para ello se hace hincapié en sus primeras fases, recogiéndose los principales artefactos generados de estas.

2.1. Fase inicial del Proceso de Ingeniería de Software.

Las metodologías de desarrollo de software, como se mostró en el capítulo anterior definen los pasos a seguir para el desarrollo de un software y SXP, metodología que se acordó seguir para guiar el proceso de desarrollo de Guano, en su primera fase, Planificación – Definición, plantea una serie de actividades que son las que generan todos los documentos que se encuentran relacionados con la concepción inicial del sistema, así como la definición del mismo, además los relacionados con el negocio, los requisitos y el diseño.

A continuación se recogen los principales artefactos que se concibieron durante el transcurso de esta fase inicial.

2.1.1. Concepción del Sistema.

Concluida la entrevista con el cliente se genera la plantilla concepción del sistema, en la cual se recogen los datos fundamentales que dan inicio al desarrollo de la propuesta de solución, incluyendo la descripción de esta.

Descripción de la solución propuesta: Guano v0.1.

Para darle solución a la situación problémica planteada se propone desarrollar un LDE orientado al usuario final, el cual contribuirá a llevar la migración a Software Libre en Cuba al hardware de bajas prestaciones, extendiendo así la vida útil de computadoras casi obsoletas y permitiendo el ahorro consecuente de recursos financieros.

El desarrollo del LDE lleva dos pasos fundamentales: el desarrollo de nuevas funcionalidades y la construcción de aplicaciones para manejar e integrar el funcionamiento de los componentes.

Partiendo de que Guano está compuesto de un manejador de ventanas, Openbox, un manejador de ficheros, PCmanFM y un panel, XFCE4-panel, este debe permitir a los usuarios nuevas funcionalidades según los requerimientos de estos. Algunas de estas nuevas características serán:

- Permitir al usuario enviar ficheros a la papelera de reciclaje.
- Soportar protocolos de red.
- Permitir al usuario la interacción con dispositivos extraíbles.
- Permitir al usuario la lanzar aplicaciones a través de comandos.
- Permitir al usuario compartir carpetas.

Por otro lado se hizo necesario una aplicación que maneje e integre el funcionamiento de los componentes, la cual es la base de la sesión, sobre ella se ejecutan todas las aplicaciones tanto predefinidas como las elegidas por el usuario.

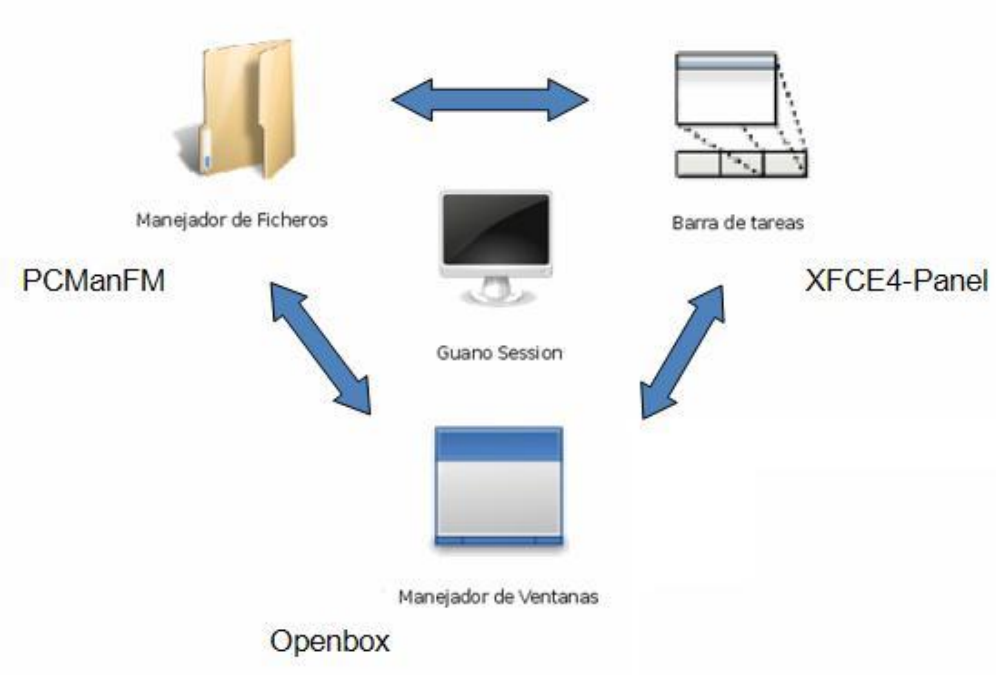


Figura 4: Propuesta de solución.

2.1.2. Modelo de dominio.

El Modelo de Dominio (o Modelo Conceptual) es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Representa clases conceptuales del dominio del problema. Es creado para documentar los conceptos claves y el vocabulario que será

utilizado por el equipo de desarrollo al referirse al sistema, y puede ser utilizado para garantizar o aumentar la comprensión del problema por los propios clientes (stakeholders), por lo que es muy útil como herramienta de comunicación en caso de que el proyecto implemente una división en equipos técnicos y de negocio. El modelo de dominio de este trabajo se muestra en la Figura 6.

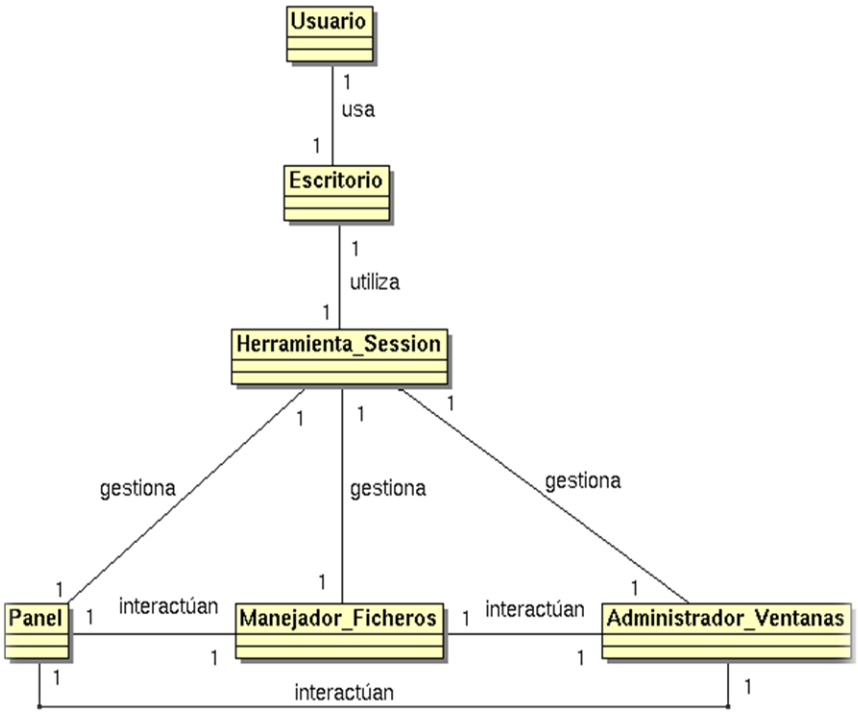


Figura 5: Diagrama del Modelo de dominio.

2.1.3. Lista de reserva del producto (LRP).

La LRP es el primer artefacto generado en la etapa de Captura de requisitos, está conformada por una lista priorizada que define el trabajo que se va a realizar en el proyecto. A continuación la LRP de la solución propuesta.

Asignado a	Ítem *	Descripción	Estimación	Estimado por
		Prioridad	***Muy Alta***	
Abel Fírvida Donéstevez	1	*Permitir integrar los componentes del DE.*	½	*AFD*
Adisleydis Rodríguez Pino	2	*Permitir al usuario interactuar con dispositivos extraíbles.*	4	*ARP*
Abel Fírvida Donéstevez Adisleydis Rodríguez Pino	3	*Permitir al usuario utilizar protocolos de red: smb.*	*8*	*AFD ARP*
		Prioridad	***Alta***	
Abel Fírvida Donéstevez	4	*Permitir al usuario escoger si desea salir de la sesión: suspendiendo, hibernando, apagando o reiniciando.*	*2*	*AFD*

Adisleydis Rodríguez Pino	5	*Permitir al usuario compartir carpetas en un dominio activo.*	*1*	*ARP*
		Prioridad	***Media***	
Abel Fírvida Donéstevez	6	*Permitir que se establezcan las configuraciones definidas por el usuario.*	*½*	*AFD*
Adisleydis Rodríguez Pino	7	*Permitir al usuario el lanzamiento de aplicaciones a través de comandos.*	*2*	*ARP*
Adisleydis Rodríguez Pino	8	*Permitir al usuario la opción de Mover a o Copiar a.*	*2*	*ARP*
		Prioridad	***Baja***	
Abel Fírvida Donéstevez	9	*Permitir al usuario bloquear la pantalla.*	*1/7*	*AFD*
Adisleydis Rodríguez Pino	10	*Permitir al usuario eliminar ficheros.*	*½*	*ARP*
Adisleydis Rodríguez Pino	11	*Permitir al usuario enviar ficheros a la papelera de reciclaje.*	*½*	*ARP*
Abel Fírvida Donéstevez Adisleydis Rodríguez Pino	12	*Permitir al usuario utilizar varios protocolos de red: ftp.*	*4*	*AFD ARP*
Abel Fírvida Donéstevez Adisleydis Rodríguez Pino	13	*Permitir al usuario utilizar vario protocolos de red: sftp.*	*4*	*AFD ARP*
		RNF (Requisitos No Funcionales)		
Adisleydis Rodríguez Pino Abel Fírvida Donéstevez	1	*El DE debe contar con las siguientes condiciones: - Simple de usar. - Interactivo. - Profesional.*		*ARP AFD*
Adisleydis Rodríguez Pino Abel Fírvida Donéstevez	2	*El DE debe garantizar la facilidad de uso por personas sin experiencia previa con las computadoras.*		*ARP AFD*
Adisleydis Rodríguez Pino Abel Fírvida Donéstevez	3	*El DE debe ser bien documentado de forma tal que en caso de mantenimiento el tiempo que se requiera sea el mínimo.*		*ARP AFD*
Abel Fírvida Donéstevez	4	*El DE deberá funcionar sobre cualquier plataforma GNU/Linux.*		*AFD*
Adisleydis Rodríguez Pino	5	*EL DE deberá funcionar sobre computadoras con al menos 800 MHZ de velocidad de procesamiento y 256 MB de RAM.*		*ARP*
Adisleydis Rodríguez Pino Abel Fírvida Donéstevez	6	*El DE debe tener una alta flexibilidad y escalabilidad para garantizar el mantenimiento periódico, posibilitando la inclusión de nuevas pruebas.*		*ARP AFD*
Abel Fírvida Donéstevez	7	*El DE permitirá su extensibilidad, permitiendo agregar nuevas*		*AFD*

		funcionalidades en un futuro.		
Adisleydis Rodríguez Pino Abel Fírvida Donéstevez	8	*El funcionamiento del DE debe ser rápido, y el tiempo de respuesta de sus aplicaciones no deberá exceder los 10 segundos.*		*ARP AFD*

Tabla 1: Lista de reserva del producto.

2.1.4. Análisis de las posibles implementaciones y componentes que pueden ser reutilizados.

La complejidad de los sistemas computacionales actuales ha llevado a buscar la reutilización del software existente y se proyecta hoy en día en diversas nuevas formas de hacer software de calidad con los costos más bajos del mercado y en tiempos que antes eran impensables.

La necesidad de reutilizar código proveniente de distintos proyectos exige una gran comprensión sobre la estructura de estos softwares. Para lo cual se generaron mediante técnicas de ingeniería inversa el siguiente diagrama, que describen la composición de las herramientas importadas para Guano.

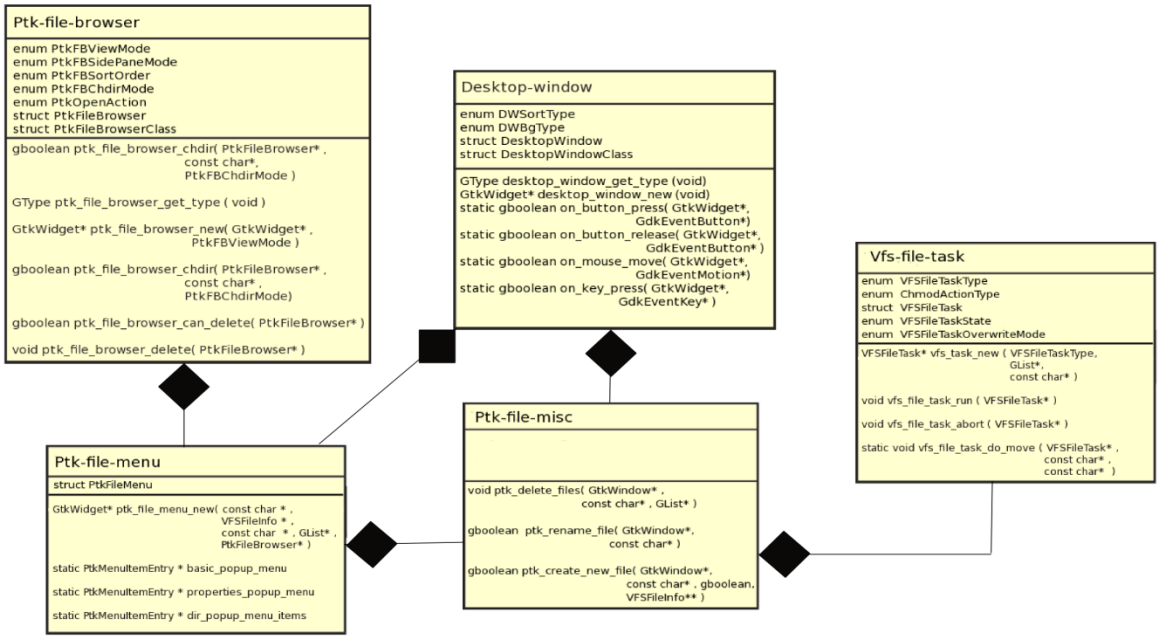


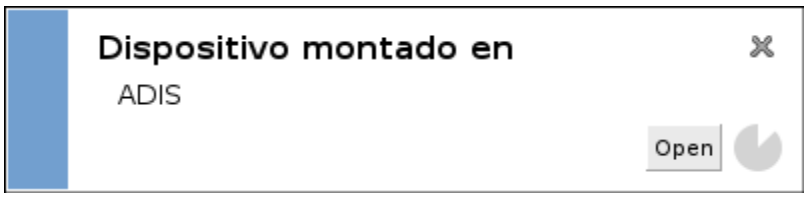
Figura 6: Diagrama de clases de PCmanFM.

2.1.5. Historias de usuario y prototipos de interfaz de usuario.

Para la implementación del entorno de escritorio se realiza la descripción de las historias de usuario y se muestran los prototipos de interfaz de usuario de algunas de estas. Las historias de usuario fueron escritas por el analista en conjunto con los clientes.

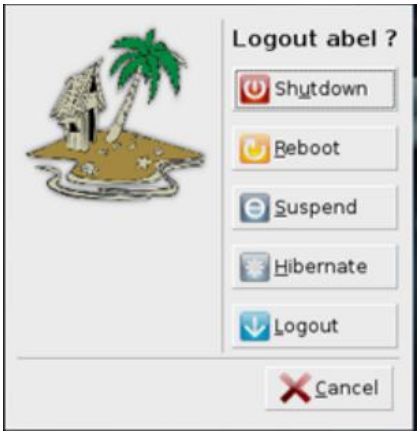
Historia de Usuario	
Número: HU-1	**Nombre Historia de Usuario:** Gestionar sesión.
Modificación de Historia de Usuario Número:	
Usuario: Abel Fírvida Donéstevez	**Iteración Asignada:** 1.
Prioridad en Negocio: Alta. (Alta / Media / Baja)	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Bajo. (Alta / Media / Baja)	**Puntos Reales:** 1/2.
Descripción: Es el desarrollo de la clase Guano-session. La cual es la base de la sesión, sobre ella se ejecutan todas las aplicaciones tanto predefinidas como las elegidas por el usuario. Al recibir la señal de salida cierra cada una de las aplicaciones que ejecutó al iniciar para que estas no pierdan datos.	
Observaciones: Guano-session se ejecuta como un demonio y recibe las órdenes de la interfaz gráfica que luego se las comunica a los clientes. La comunicación entre Guano-session y la interfaz gráfica se realiza usando sockets.	

Historia de Usuario	
Número: HU-2	**Nombre Historia de Usuario:** Gestionar la interacción con dispositivos extraíbles.
Modificación de Historia de Usuario Número:	
Usuario: Adisleydis Rodríguez Pino	**Iteración Asignada:** 1.
Prioridad en Negocio: Alta. (Alta / Media / Baja)	**Puntos Estimados:** 4.
Riesgo en Desarrollo: Bajo. (Alta / Media / Baja)	**Puntos Reales:** 4.

Descripción: Los dispositivos que son reconocidos por Hal son montados mostrando al usuario una notificación con la opción de explorarlos con el manejador de ficheros.
Prototipo de interface: 

Historia de Usuario	
Número: HU-3	**Nombre Historia de Usuario:** Permitir interacción con ficheros compartidos en la red.
Modificación de Historia de Usuario Número:	
Usuario: Abel Fírvida Donéstevez. Adisleydis Rodríguez Pino	**Iteración Asignada:** 1.
Prioridad en Negocio: Alta. (Alta / Media / Baja)	**Puntos Estimados:** 8.
Riesgo en Desarrollo: Bajo. (Alta / Media / Baja)	**Puntos Reales:** 8.
Descripción: Soporte para hacer que PCmanFM interactúe con direcciones de red tan fácilmente como con direcciones locales a través de GVFS.	

Historia de Usuario	
Número: HU-4	**Nombre Historia de Usuario:** Gestionar la salida de la sesión.
Modificación de Historia de Usuario Número:	
Usuario: Abel Fírvida Donéstevez	**Iteración Asignada:** 1.
Prioridad en Negocio: Alta. (Alta / Media / Baja)	**Puntos Estimados:** 2.

Riesgo en Desarrollo: Bajo. (Alta / Media / Baja)	**Puntos Reales:** 2.
Descripción: Desarrollo de una aplicación que maneje la salida de la sesión. Está basada en la aplicación lxsession-logout del proyecto LXDE, al igual que su progenitor soporta reiniciar, apagar, hibernar y suspender a través de GDM y tiene una imagen que la hace más agradable.	
Observaciones: Se utilizó para el desarrollo de esta aplicación el código fuente de lxsession-logout, agregándole funcionalidades para la comunicación a través de sockets permitiendo la interacción con Guano-session.	
Prototipo de interface: 	

Historia de Usuario	
Número: HU-5	**Nombre Historia de Usuario:** Establecer configuraciones al inicio de la sesión.
Modificación de Historia de Usuario Número:	
Usuario: Abel Fírvida Donéstevez	**Iteración Asignada:** 2.
Prioridad en Negocio: Media. (Alta / Media / Baja)	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Bajo. (Alta / Media / Baja)	**Puntos Reales:** 1/2.
Descripción: Desarrollo de un script que establece las configuraciones definidas por el usuario, como puede ser el reemplazo de cualquiera de las aplicaciones predeterminadas, vela por la existencia de todos los módulos en el sistema y de no ser así trata de encontrar alguna otra aplicación que pueda cubrir la necesidad.	
Observaciones: Es un script Bash que rige el inicio de la sesión.	

Historia de Usuario	
Número: HU-6	**Nombre Historia de Usuario:** Ejecutar aplicaciones.
Modificación de Historia de Usuario Número:	
Usuario: Adisleydis Rodríguez Pino	**Iteración Asignada:** 2.
Prioridad en Negocio: Media. (Alta / Media / Baja)	**Puntos Estimados:** 2.
Riesgo en Desarrollo: Bajo. (Alta / Media / Baja)	**Puntos Reales:** 2.
Descripción: Herramienta para ejecutar aplicaciones. Posee soporte para completar comandos mientras se está escribiendo sobre ella, recuerda aplicaciones que ya se escribieron, se integra con PCmanFM y Guano-authentication-manager para navegar tanto direcciones locales como remotas y se integra a Guano-session para reconocer la terminal preferida del usuario a la hora de ejecutar una aplicación con la opción "Lanzar en terminal".	

Historia de Usuario	
Número: HU-7	**Nombre Historia de Usuario:** Bloquear sesión.
Modificación de Historia de Usuario Número:	
Usuario: Abel Fírvida Donéstevez	**Iteración Asignada: 3**.
Prioridad en Negocio: Baja. (Alta / Media / Baja)	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Bajo. (Alta / Media / Baja)	**Puntos Reales:** 1/7.
Descripción: Programa para bloquear la pantalla, se integra a las preferencias del usuario y a múltiples programas de Salva Pantallas.	

Historia de Usuario	
Número: HU-8	**Nombre Historia de Usuario:** Envío de ficheros a la papelera de reciclaje.

Modificación de Historia de Usuario Número:	
Usuario: Adisleydis Rodríguez Pino	**Iteración Asignada:** 3.
Prioridad en Negocio: Baja. (Alta / Media / Baja)	**Puntos Estimados:** 1.
Riesgo en Desarrollo: Bajo. (Alta / Media / Baja)	**Puntos Reales:** 1/2.
Descripción: Soporte para papelera de reciclaje que permita enviar archivos a la papelera o eliminarlos.	
Observaciones: Para implementar esta funcionalidad fue necesario modificar código ya existente de PCmanfm, específicamente de las clases desktop_window, vfs_file_task y ptk_file_browser.	

2.1.6. Lista de riesgos.

En todo desarrollo de software actúan posibles riesgos, los cuales deben ser controlados a través de una estrategia de mitigación y si el riesgo se materializa se necesita de un plan de contingencia. Los riesgos que se pueden presentar en el desarrollo de Guano están en la planilla Lista de riesgos (Ver **Anexo 2**).

2.1.7. Metáfora del sistema.

"Al iniciar la sesión con el LDE se debe mostrar el escritorio, un manejador de ventanas y la barra de tareas que le permitan al usuario la interacción con la computadora"

2.2. Desarrollo ágil de Guano.

En esta segunda fase se describe toda la dinámica del proyecto a través de actividades que estarán asociadas a las historias de usuario y que permitirán su implementación, además de otros modelos auxiliares.

2.2.1. Planificación del proyecto por roles.

Rol	Miembro
Gerente (Management)	Adisleydis Rodríguez Pino
Cliente (Customer)	Lic. Jose Jorge Lorenzo Vila. Abel Maso Nuñes
Programadores (Programmers)	Abel Fírvida Donéstevez

	Adisleydis Rodríguez Pino Félix Alejandro Prieto Carratala
Analista (Analyst)	Abel Fírvida Donéstevez
Diseñadores (Designers)	Barbarita Ramírez Pérez Abel Fírvida Donéstevez
Encargado de Pruebas (Tester)	Abel Fírvida Donéstevez Adisleydis Rodríguez Pino
Arquitecto (Architect)	Adisleydis Rodríguez Pino

2.2.2. Historias de usuarios y tareas de ingeniería.

A continuación se relacionan las historias de usuario con sus tareas de ingenierías, basándose en la prioridad que tienen y los usuarios que se encargan de desarrollarlas. Debido a que el proceso es cambiante para ir adecuándolo a las necesidades y nuevas propuestas, esta es solo una planificación inicial.

La *historia #1 Gestionar sesión,* se dividió en tres tareas de la ingeniería:

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU-1
Nombre Tarea: Programar clase socket_server.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1/7
Fecha Inicio: 2/11/2008	**Fecha Fin:** 3/11/2008
Programador Responsable: Abel Fírvida Donéstevez.	
Descripción: Clase para el envío y recepción de mensajes a través de sockets.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** HU-1
Nombre Tarea: Programar la clase guano_session.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 2/7
Fecha Inicio: 3/11/2008	**Fecha Fin:** 5/11/2008
Programador Responsable: Abel Fírvida Donéstevez.	

Descripción: Clase para el manejo de la sesión.	
	Tarea de Ingeniería
Número Tarea: 3	**Número Historia de Usuario:** HU-1
Nombre Tarea: Programar la funcionalidad de ejecutar las aplicaciones definidas por el usuario al inicio.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 2/7
Fecha Inicio: 04/01/09	**Fecha Fin:** 5/11/2008
Programador Responsable: Abel Fírvida Donéstevez.	
Descripción: Agregar el método run-startup-application a la clase Guano-session.	

La *historia #2 Gestionar la interacción con dispositivos* extraíbles, tiene una tarea de la ingeniería:

	Tarea de Ingeniería
Número Tarea: 1	**Número Historia de Usuario:** HU-2
Nombre Tarea: Implementar la función correspondiente a la señal DeviceAdded de Hal.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 4
Fecha Inicio: 2/11/2008	**Fecha Fin:** 3/12/2008
Programador Responsable: Adisleydis Rodríguez Pino. Félix Alejandro Prieto Carratalá.	
Descripción: Esta función chequea las características del dispositivo conectado, lo monta y muestra una notificación con información del mismo.	

La *historia #3 Permitir interacción con ficheros compartidos en la red,* se dividió en tres tareas de la ingeniería:

	Tarea de Ingeniería
Número Tarea: 1	**Número Historia de Usuario:** HU-3.
Nombre Tarea: Investigar sobre como otros manejadores de ficheros interactúan con los ficheros compartidos.	
Tipo de Tarea : Investigación	**Puntos Estimados:** 2

Fecha Inicio: 2/12/2008	**Fecha Fin:** 17/12/2008
Programador Responsable: Abel Fírvida Donéstevez. Adisleydis Rodríguez Pino.	
Descripción: Investigar dentro del código fuente de los manejadores de ficheros: Mc, Krusader, Konqueror, Nautilus y Gnomba que tecnologías usan para la interacción con sistemas virtuales de ficheros. Escoger una de estas tecnologías para integrarla al PCmanFM.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** HU-3.
Nombre Tarea: Desarrollar clase guano-authentication-manager para administrar la interacción con sistemas virtuales de ficheros a través del API de GVFS.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1
Fecha Inicio: 5/01/2009	**Fecha Fin:** 13/01/2009
Programador Responsable: Abel Fírvida Donéstevez. Adisleydis Rodríguez Pino.	
Descripción: Desarrollar la clase vfs-network para la interacción de PCmanFM con los sistemas remotos de ficheros. Sobre esta clase se basa la aplicación guano-authentication-manager. Ella se encarga de montar, desmontar, resolver las necesidades de la conexion tales como: Autenticación (Nombres de usuarios, contraseñas y dominios) y el correcto formato de las direcciones *uri*[3].	

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** HU-3.
Nombre Tarea: Programar la clase Vfs-network que permita a PCmanFM el manejo interno de sistemas virtuales de ficheros.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 2
Fecha Inicio: 5/01/2009	**Fecha Fin:** 19/01/2009
Programador Responsable: Abel Fírvida Donéstevez. Adisleydis Rodríguez Pino.	

[3] URI (Uniform Resource Identifier): Identificador de recursos uniforme. Una dirección web.

Descripción: Basado en el código de la aplicación *gvfs-mount* crear una interfaz gráfica que permita al manejador de ficheros de Guano explorar sistemas virtuales de ficheros.

La *historia #4 Gestionar la salida de la sesión,* se dividió en tres tareas de la ingeniería:

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU-4
Nombre Tarea: Programar clase socket_client.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1/7
Fecha Inicio: 2/12/2008	**Fecha Fin:** 3/12/2008
Programador Responsable: Abel Fírvida Donéstevez.	
Descripción: Clase para el envío de mensajes a través de sockets.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** HU-4
Nombre Tarea: Integración de la clase socket_client con el código fuente de lxsession-logout.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1
Fecha Inicio: 3/12/2008	**Fecha Fin:** 10/12/2008
Programador Responsable: Abel Fírvida Donéstevez.	
Descripción: Se le agregó la función *send_message* a la función principal de lxsession-logout. Esta función específicamente se encarga de enviar la señal de salir de la sesión a Guano-session.	

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** HU-4
Nombre Tarea: Agregar imagen por defecto a guano-logout.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1/7

Fecha Inicio: 10/12/2008	**Fecha Fin:** 11/12/2008
Programador Responsable: Abel Fírvida Donéstevez.	
Descripción: Se le agregó una imagen por defecto en la función principal de guano-logout.	

La *historia #5 Establecer configuraciones al inicio de la sesión,* se dividió en dos tareas de la ingeniería:

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU-5
Nombre Tarea: Definir listas de aplicaciones.	
Tipo de Tarea: Investigación.	**Puntos Estimados:** 1/7
Fecha Inicio: 4/01/2009	**Fecha Fin:** 5/01/2009
Programador Responsable: Abel Fírvida Donéstevez.	
Descripción: Se definieron listas de aplicaciones para suplir las necesidades en caso de no tener instalado las aplicaciones por defecto de Guano. Estas listas son: • Manejadores de ventanas: "openbox xfwm4 metacity twm ratpoisom kwin" • Manejadores de ficheros:"PCmanFM nautilus thunar konqueror" • Salvapantalla: "xscreensaver gnome-screensaver" • Paneles: "xfce4-panel lxpanel gnome-panel" • Terminales: "terminal aterm mrxvt gnome-terminal rxvt urxvt konsole xterm"	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** HU-5
Nombre Tarea: Programar el script.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 2/7
Fecha Inicio: 5/01/2009	**Fecha Fin:** 7/01/2009
Programador Responsable: Abel Fírvida Donéstevez. Mijail Hurtado Fedorovich.	

Descripción: Programar el script en el lenguaje Bash que revisa las configuraciones del usuario, de no tener ninguna predefinida busca la primera aplicación de la lista para suplir los componentes por defectos. Finalmente ejecuta guano-session (comando para iniciar la sesión).

La *historia #6 Ejecutar aplicaciones*, se dividió en cuatros tareas de la ingeniería:

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU-6
Nombre Tarea: Investigar los métodos de completamiento de comandos de otros lanzadores de aplicaciones.	
Tipo de Tarea: Investigación.	**Puntos Estimados:** 1
Fecha Inicio: 2/11/2008	**Fecha Fin:** 9/11/2008
Programador Responsable: Adisleydis Rodríguez Pino.	
Descripción: Investigar el completamiento de código de las herramientas: ▪ Xfce4-run. ▪ Lanzador de aplicaciones de Gnome. ▪ Lanzador de aplicaciones de KDE. ▪ Lanzador de aplicaciones de Windows. ▪ Lanzador de aplicaciones de LXDE.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** HU-6
Nombre Tarea: Implementar la clase Gtk-run.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1/2
Fecha Inicio: 10/11/2008	**Fecha Fin:** 13/11/2008
Programador Responsable: Adisleydis Rodríguez Pino.	
Descripción: Clase que permite lanzar aplicaciones a partir de una cadena de texto.	

Tarea de Ingeniería	
Número Tarea: 3	**Número Historia de Usuario:** HU-6
Nombre Tarea: Integrar la clase Gtk-run al API GIO.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1/2
Fecha Inicio: 13/11/2008	**Fecha Fin:** 17/11/2008
Programador Responsable: Adisleydis Rodríguez Pino.	
Descripción: Integrar la clase Gtk-run al API de GIO para la interacción con sistemas virtuales de ficheros.	

Tarea de Ingeniería	
Número Tarea: 4	**Número Historia de Usuario:** HU-6
Nombre Tarea: Incorporar código para el completamiento de comandos.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1/2
Fecha Inicio: 13/11/2008	**Fecha Fin:** 17/11/2008
Programador Responsable: Adisleydis Rodríguez Pino.	
Descripción: Incorporar código para el completamiento de comandos en la clase Gtk-run.	

La *historia #7 Bloquear sesión*, se dividió en dos tareas de la ingeniería:

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU-7
Nombre Tarea: Programar el script guano-lock.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1/7
Fecha Inicio: 4/01/2009	**Fecha Fin:** 5/01/2009
Programador Responsable: Abel Fírvida Donéstevez.	
Descripción: Programar el script que reconozca cual es el salva pantalla activo y en dependencia de esto ejecute el comando de bloquear la pantalla.	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** HU-7
Nombre Tarea: Integrar el script guano-lock a la aplicación Guano-session.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1/7
Fecha Inicio: 4/01/2009	**Fecha Fin:** 5/01/2009
Programador Responsable: Abel Fírvida Donéstevez.	
Descripción: Agregar el script como herramienta al manejador de sesión Guano-session.	

La *historia #8 Envío de ficheros a la papelera de reciclaje*, se dividió en dos tareas de la ingeniería:

Tarea de Ingeniería	
Número Tarea: 1	**Número Historia de Usuario:** HU-8.
Nombre Tarea: Agregar función de enviar a la papelera a PCmanFM.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1
Fecha Inicio: 2/02/2009	**Fecha Fin:** 9/02/2009
Programador Responsable: Adisleydis Rodríguez Pino.	
Descripción: Función que permite enviar ficheros a la papelera de reciclaje, además de brindar la posibilidad de recuperar los ficheros eliminados. Esta función utiliza el API de gvfs en específico el método g_file_trash. (Ver http://library.gnome.org/devel/gio/unstable/GFile.html#g-file-trash).	

Tarea de Ingeniería	
Número Tarea: 2	**Número Historia de Usuario:** HU-8.
Nombre Tarea: Vincular la combinación de teclas Shift + Delete a la función de eliminar ficheros.	
Tipo de Tarea: Desarrollo.	**Puntos Estimados:** 1
Fecha Inicio: 2/02/2009	**Fecha Fin:** 9/02/2009
Programador Responsable: Adisleydis Rodríguez Pino.	

Descripción: Se vinculo la combinación de teclas Shift + Delete a la función de eliminar ficheros ya existente en la clase ptk_file_task. Para esto se modificaron las funciones *on_folder_view_key_press_event* de la clase ptk_file_browser y *on_key_press* de la clase desktop.window.c.

2.2.3. Plan de entregas.

Release	Descripción de la iteración	Orden de la HU a implementar	Duración total
1	Al concluir esta iteración el usuario tendrá la posibilidad de interactuar con dispositivos extraíbles, con ficheros compartidos en la red y una aplicación que le permita gestionar la salida de la sesión.	HU-1 HU-2 HU-3 HU-4	02/11/2008 – 04/03/2009
2	En esta iteración se incorpora una aplicación que permita ejecutar aplicaciones. Además de un script que permite establecer configuraciones definidas por el usuario siempre que inicie la sesión.	HU-5 HU-6	04/01/2009 – 22/01/2009
3	En esta iteración el usuario ya tendrá una aplicación para bloquear la pantalla y tendrá la posibilidad de eliminar ficheros con el uso de una papelera de reciclaje.	HU-7 HU-8	05/01/2009 – 14/01/2009

2.2.4. Diagrama de componentes.

Lo que distingue a un Diagrama de Componentes (DC) de otros tipos de diagramas es su contenido. Normalmente contienen componentes, interfaces y relaciones entre ellos. Y como todos los diagramas, también puede contener paquetes utilizados para agrupar elementos del modelo. Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes de software, sean éstos de código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo.

En las Figuras se muestran los diagramas de componentes de Guano.

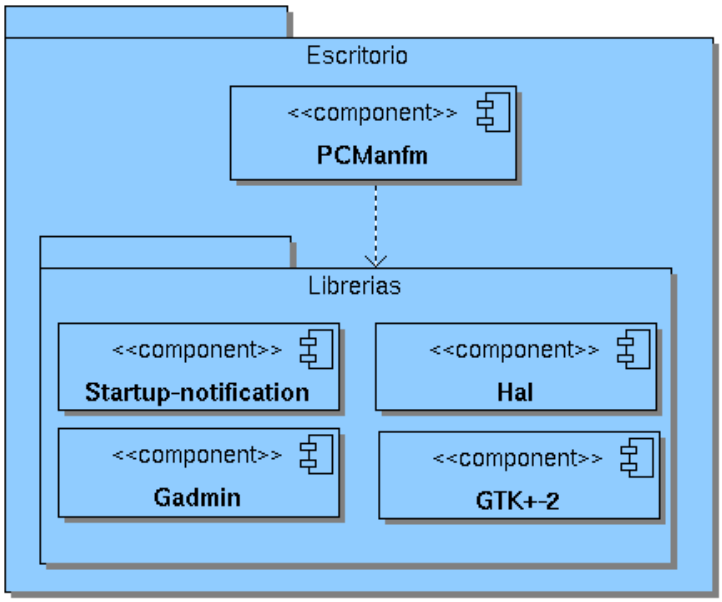


Figura 7: Diagrama de componentes del Escritorio.

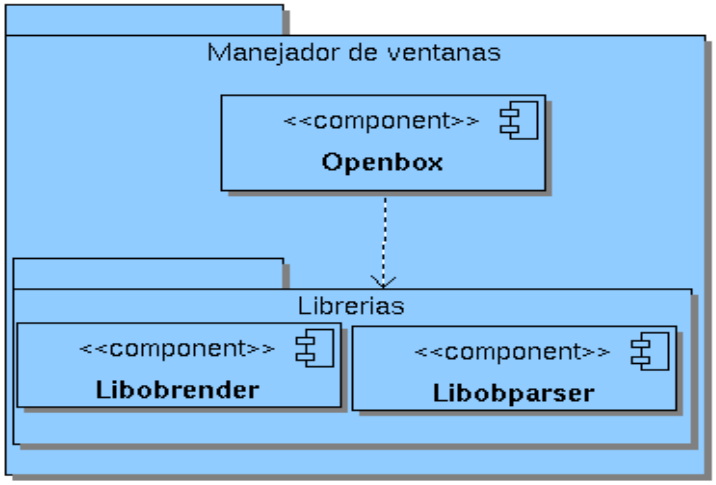


Figura 8: Diagrama de componentes del Manejador de ventanas.

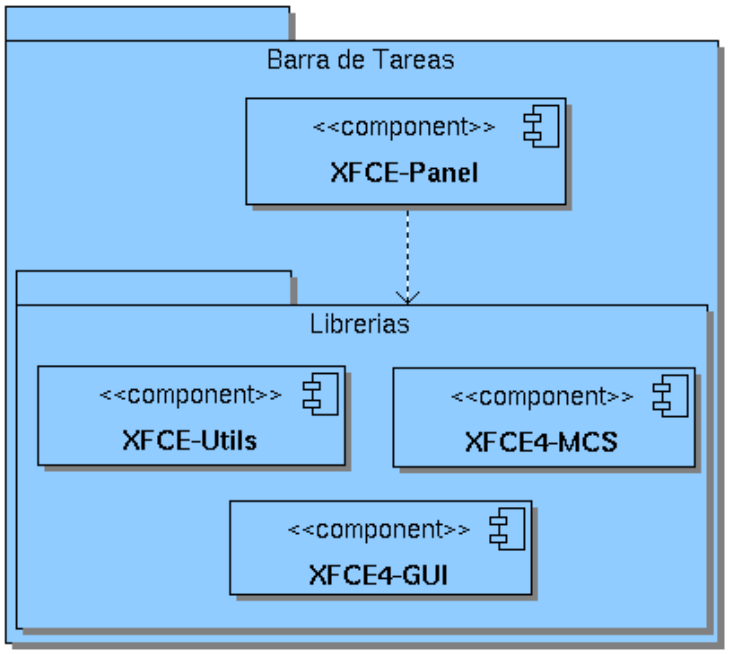


Figura 9: Diagrama de componentes de la Barra de tareas.

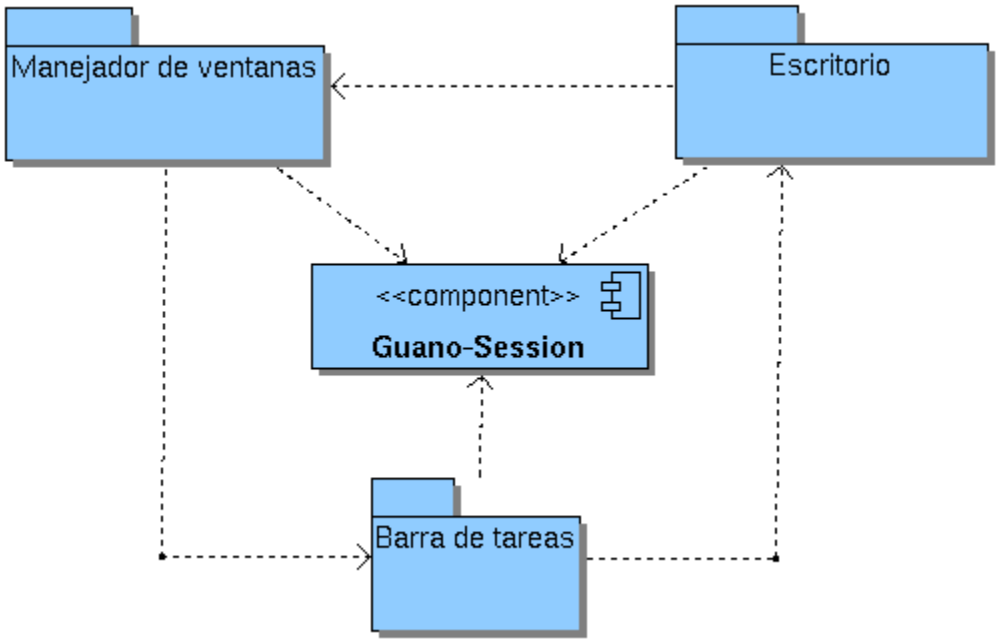


Figura 10: Diagrama de componentes de Guano.

2.3. Conclusiones.

Una vez descrita la solución propuesta, se puede concluir que esta consiste en un paquete de componentes que juntos suministran un escritorio ligero de funcionalidad completa, el cual combina la usabilidad y funcionalidad de los sistemas operativos de las familias Windows y Linux, conservando

como prioridad un buen rendimiento en PC de bajas prestaciones técnicas, adaptándose a las necesidades del proceso migratorio de la informática cubana a sistemas operativos de tipo FOSS en computadoras con dichas características.

En este capítulo se definieron además los requisitos funcionales y no funcionales, el modelo de dominio y se redactaron las historias de usuario, como resultado de la fase inicial de la metodología SXP. Para la segunda fase se realizó la planificación del proyecto, se establecieron para cada historia de usuario sus principales tareas asociadas para la primera planificación y se priorizaron. La planificación de iteraciones se hizo teniendo en cuenta la prioridad para el negocio de cada historia de usuario. Finalmente se realizó los diagramas de componentes el cual permitió dar una mejor visión del proyecto.

3

CAPÍTULO

3. VALIDACIÓN DE LA SOLUCIÓN PROPUESTA.

El objetivo de las pruebas de aceptación es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema que determine su aceptación, desde el punto de vista de su funcionalidad y rendimiento, según la metodología SXP.

En el presente capítulo se describen algunos casos de pruebas o pruebas de aceptación que se le han efectuado a Guano en cada una de las iteraciones, para comprobar el funcionamiento de acuerdo a las historias de usuario. Además se exponen una relación de las funcionalidades con las que cuenta el DE hasta la fecha.

3.1. Casos de Prueba.

3.1.1. Caso de Prueba Historia de Usuario: HU-1.

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: Gestionar sesión.

Para esta historia se comprueba que la aplicación Guano-session ejecute correctamente a todos los elementos del DE y las aplicaciones definidas en el fichero de inicio. También se debe verificar que el programa espere por la señal de salida para terminar la sesión, cerrando a su vez cada una de las aplicaciones que inició.

	Caso de Prueba de Aceptación
Código Caso de Prueba: 1	**Nombre Historia de Usuario:** Gestionar sesión.
Nombre de la persona que realiza la prueba: Abel Fírvida Donéstevez.	
Descripción de la Prueba: Se iniciará la sesión escogiendo como manejador de sesión a Guano. Una vez iniciado se comprueba que todas los componentes del DE (PcmanFM, XFCE4-Panel, Openbox) y las aplicaciones definidas en el fichero de inicio estén activas.	
Condiciones de Ejecución: No deben existir errores de compilación y deben estar ubicados en las carpetas	

/usr/bin y /usr/share/xsession/ los scripts startguano y Guano.desktop respectivamente.
Entrada / Pasos de ejecución: Iniciar cualquier gestor de acceso (GDM, XDM, KDM o SLIM), mediante este seleccionar a Guano como manejador de sesión y autenticar un usuario.
Resultado Esperado: Se inicia la sesión con todas las aplicaciones y componentes en correcto funcionamiento.
Evaluación de la Prueba: Satisfactoria.

3.1.2. Caso de Prueba Historia de Usuario: HU-2.

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: Gestionar la interacción con dispositivos extraíbles.

Para esta historia se comprueba que se reconozcan los dispositivos extraíbles mostrando una notificación en la esquina inferior derecha de la pantalla.

	Caso de Prueba de Aceptación
Código Caso de Prueba: 1	**Nombre Historia de Usuario:** Gestionar la interacción con dispositivos extraíbles.
Nombre de la persona que realiza la prueba: Adisleydis Rodríguez Pino.	
Descripción de la Prueba: Al introducir un dispositivo extraíble se debe mostrar una notificación con la opción de abrir el mismo y al activar dicha funcionalidad se debe abrir el manejador de ficheros mostrando su contenido.	
Condiciones de Ejecución: El demonio Hald debe estar activo y el usuario debe tener los permisos necesarios para interactuar con él.	
Entrada / Pasos de ejecución: Introducir un dispositivo extraíble y esperar unos segundos por la notificación de que ha sido leído, luego hacer clic en el botón “Abrir” para examinar el contenido del recurso.	
Resultado Esperado: Se muestra correctamente que el dispositivo fue montado y la opción “Abrir” lleva directamente al contenido del mismo.	
Evaluación de la Prueba: Satisfactoria.	

3.1.3. Caso de Prueba Historia de Usuario: HU-3.

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: Permitir interacción con ficheros compartidos en la red.

Para esta historia se comprueba que el manejador de ficheros pueda interactuar con ficheros compartidos en la red.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1	**Nombre Historia de Usuario:** Permitir interacción con ficheros compartidos en la red.
Nombre de la persona que realiza la prueba: Abel Fírvida Donéstevez. Adisleydis Rodríguez Pino.	
Descripción de la Prueba: Se introducirá en la barra de navegación del manejador de ficheros la uri de un recurso compartido en la red a través del protocolo samba (Ej. *smb://10.0.0.22/software*) acto seguido el navegador debe mostrar el contenido de esta carpeta.	
Condiciones de Ejecución: Para el correcto funcionamiento de esta funcionalidad debe estar ejecutándose el demonio gvfs-fuse-daemon y el usuario debe tener permisos sobre el módulo fuse del kernel, que también debe estar cargado.	
Entrada / Pasos de ejecución: Se introducirá en la barra de navegación del manejador de ficheros una uri de un recurso compartido en la red a través del protocolo samba.	
Resultado Esperado: El manejador de ficheros interactuará con el recurso compartido transparentemente.	
Evaluación de la Prueba: Satisfactoria.	

3.1.4. Caso de Prueba Historia de Usuario: HU-4.

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: Gestionar la salida de la sesión.

Para esta historia se comprueba que la aplicación permita correctamente salir de la sesión, cambiar de usuario, hibernar, reiniciar, suspender y apagar el ordenador.

Caso de Prueba de Aceptación	
Código Caso de Prueba: 1	**Nombre Historia de Usuario:** Gestionar la salida de la sesión
Nombre de la persona que realiza la prueba: Abel Fírvida Donéstevez.	
Descripción de la Prueba: Al pulsar el botón de “Salir de sesión” se debe cerrar la sesión.	
Condiciones de Ejecución: Guano sesión debe haberse ejecutado correctamente al inicio de la sesión.	
Entrada / Pasos de ejecución: Ejecutar Guano-logout y seleccionar la opción “Salir de sesión”.	
Resultado Esperado: La sesión se debe terminar satisfactoriamente.	
Evaluación de la Prueba: Satisfactoria.	

3.1.5. Caso de Prueba Historia de Usuario: HU-5.

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: Establecer configuraciones al inicio de la sesión.

Para esta historia se comprueba que al iniciarse el script Startguano se establezcan las configuraciones definidas por el usuario y se inicie la aplicación Guano-session.

	Caso de Prueba de Aceptación
Código Caso de Prueba: 1	**Nombre Historia de Usuario:** Establecer configuraciones al inicio de la sesión
Nombre de la persona que realiza la prueba: Abel Fírvida Donéstevez.	
Descripción de la Prueba: Se ejecuta el script Startguano por un usuario que no haya creado ningún fichero de configuración.	
Condiciones de Ejecución: El script Startguano debe estar en correcto estado y debe poseer permiso de ejecución por parte de cualquier usuario.	
Entrada / Pasos de ejecución: Iniciar cualquier gestor de acceso (GDM, XDM, KDM o SLIM), mediante este seleccionar a Guano como manejador de sesión y autenticar un usuario.	
Resultado Esperado: Se crean todas las configuraciones del usuario y se inicia correctamente la sesión.	
Evaluación de la Prueba: Satisfactoria.	

3.1.6. Caso de Prueba Historia de Usuario: HU-6.

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: Ejecutar aplicaciones.

Para esta historia se comprueba que se ejecuten los comandos o se conecten las direcciones de red que se escriben en la entrada de la aplicación Gtkrun.

	Caso de Prueba de Aceptación
Código Caso de Prueba: 1	**Nombre Historia de Usuario:** Ejecutar aplicaciones
Nombre de la persona que realiza la prueba: Adisleydis Rodríguez Pino	
Descripción de la Prueba: Se escribe un comando en la entrada de Gtkrun, al pulsar el botón "*Ejecutar"* se debe lanzar la aplicación.	
Condiciones de Ejecución: El comando escrito debe existir.	
Entrada / Pasos de ejecución: Ejecutar Gtkrun, escribir un comando en la entrada de texto y activar la acción	

"Ejecutar".
Resultado Esperado: Se debe ejecutar correctamente el comando escrito.
Evaluación de la Prueba: Satisfactoria.

3.1.7. Caso de Prueba Historia de Usuario: HU-7.

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: Bloquear sesión.

Para esta historia se comprueba que al ejecutar la aplicación Guano-lock se bloquee la sesión ejecutándose el programa salva pantallas.

	Caso de Prueba de Aceptación
Código Caso de Prueba: 1	**Nombre Historia de Usuario:** Bloquear sesión.
Nombre de la persona que realiza la prueba: Abel Fírvida Donéstevez.	
Descripción de la Prueba: Se ejecuta el script Guano-lock.	
Condiciones de Ejecución: Debe haber instalado al menos un programa salva pantallas (Gnome-screensaver, xscreensaver, xflock).	
Entrada / Pasos de ejecución: Se ejecuta el script Guano-lock a través del menú del panel.	
Resultado Esperado: La sesión debe bloquearse y ejecutar algún programa de salva pantallas.	
Evaluación de la Prueba: Satisfactoria.	

3.1.8. Caso de Prueba Historia de Usuario: HU-8.

Esta sección cubre el conjunto de pruebas funcionales relacionadas con la historia de usuario: Envío de ficheros a la papelera de reciclaje.

Para esta historia se comprueba que al eliminar un archivo no se borre completamente sino que sea enviado a la papelera de reciclaje.

	Caso de Prueba de Aceptación
Código Caso de Prueba: 1	**Nombre Historia de Usuario:** Envío de ficheros a la papelera de reciclaje
Nombre de la persona que realiza la prueba: Adisleydis Rodríguez Pino.	
Descripción de la Prueba: Se debe hacer clic derecho sobre un fichero y seleccionar la opción eliminar. Acto seguido el fichero debe moverse a la papelera de reciclaje.	

Condiciones de Ejecución: No debe haber errores en la compilación del programa.
Entrada / Pasos de ejecución: Se debe hacer clic derecho sobre un fichero y seleccionar la opción eliminar.
Resultado Esperado: El fichero debe moverse a la papelera de reciclaje.
Evaluación de la Prueba: Satisfactoria.

3.2. Pruebas de rendimiento y portabilidad.

Para corroborar el cumplimiento del objetivo principal planteado por esta investigación se realizaron pruebas de rendimiento al LDE que fueron recogidas en la *Tabla 2.*

CPU / Velocidad	RAM	Disco duro	Velocidad de respuesta	Memoria utilizada
Pentium III 800 MHz	192 MB	7200 rpm	20 s	20 MB
Celeron 500 MHz	256 MB	4500 rpm	10 s	28 MB
Pentium IV 1.4 GHz	256 MB	7200 rpm	0.28 s	10 MB
Pentium IV 3.0 GHz	1 GB	7200 rpm	0.032 s	12 MB
VirtualBox 3.0 GHz	128 MB	Disco Virtual	0.032 s	12 MB
VirtualBox 3.0 GHz	192 MB	Disco Virtual	0.032 s	12 MB
VirtualBox 3.0 GHz	512 MB	Disco Virtual	0.032 s	15 MB

Tabla 2: Pruebas de rendimiento de Guano sobre distintas configuraciones de hardware.

CPU / Velocidad	RAM	Disco duro	Velocidad de respuesta	Memoria utilizada
Pentium IV 1.4 GHz	256 MB	7200 rpm	20 s	70 MB
Pentium IV 3.0 GHz	1 GB	7200 rpm	10 s	90 MB
VirtualBox 3.0 GHz	512 MB	Disco Virtual	32 s	85 MB

Tabla 3: Pruebas de rendimiento de Gnome2.24 sobre distintas configuraciones de hardware.

Las pruebas de rendimiento son las pruebas que se realizan, desde una perspectiva, para determinar lo rápido que realiza una tarea un sistema en condiciones particulares de trabajo. Como se evidencia en la *Tabla 2* se logró obtener un buen rendimiento LDE para cada variante, ratificando que Guano es eficiente incluso en PC de bajas prestaciones.

Por otro lado haciendo un análisis comparativo entre los resultados obtenidos en las *Tablas 2* y *3* se puede concluir que Guano es mucho más eficiente que el DE Gnome. Nótese que Gnome no se pudo probar en todas las configuraciones de hardware en que se examinó el rendimiento de Guano debido a que estas no alcanzan los requerimientos mínimos de este DE.

Partiendo de aquí el LDE fue probado en los Sistemas Linux que a continuación se listan:

Nova Fedora Core 8 y 9

Gentoo

Red Hat Enterprises Linux

OpenSuse 10 y 11

Ubuntu

Arch Linux

OpenGeu.

Debian

En todos los casos se obtuvo un buen desempeño del software, lo cual demuestra la posibilidad de que no sea imprescindible tener una distribución específica para poder usar Guano y así se puedan incorporar al equipo de desarrolladores programadores de todo el país sin importar sus preferencias en materia de distribuciones.

3.3. Resultados obtenidos.

En este epígrafe se relacionan los resultados alcanzados hasta el momento por el equipo de desarrollo de Guano.

Como fruto de este trabajo Guano está disponible en su versión 0.1, que está siendo usada actualmente en algunos laboratorios de las Facultad 4, 3 y 10 de la UCI. Esto significa que se pueden esperar muchos más resultados en versiones superiores.

3.3.1. Acerca de las funcionalidades obtenidas.

Entre las principales funcionalidades que presenta Guano en su versión 0.1 se encuentran:

- Es fácil de utilizar incluso por usuarios inexpertos en el diseño.
- Permite la interacción con direcciones de red tan fácilmente como con direcciones locales a través de GVFS.
- Posee una herramienta que controla los elementos y todas las aplicaciones que se ejecutan.
- Posee una aplicación que gestiona la salida de la sesión.

3.4. Conclusiones.

En este capítulo como resultado de las pruebas que se realizaron a las historias de usuarios se obtuvo un conjunto de funcionalidades validadas que garantizan la entrega del producto con calidad, respondiendo así a las necesidades del cliente.

CONCLUSIONES GENERALES

El proceso de migración a Software Libre en Cuba, la creación de la Universidad de las Ciencias Informáticas con su proyecto Nova y otros proyectos que se desenvuelven dentro de ella, forman parte del camino trazado por la dirección de la Revolución para lograr la necesaria independencia tecnológica que debe ser forjada por el ingenio creativo cubano. Lo cual indica la necesidad de no incurrir en esfuerzos innecesarios. Guano es una muestra de ello, es un ejemplo de cómo adaptar el mundo del Software Libre a las especificidades del país.

El proceso de desarrollo de Guano arrojó como resultado un software que combina la usabilidad de los sistemas operativos de las familias Windows y Linux, conservando como prioridad un buen rendimiento en PC de bajas prestaciones técnicas, adaptándose a las necesidades del proceso migratorio de la informática cubana a sistemas operativos de tipo FOSS y demostró que los objetivos trazados en el presente trabajo se han cumplido satisfactoriamente.

El primer lanzamiento de una versión estable de este producto dará un vuelco a la situación del proceso de migración en el país, ya que se contará con una herramienta que permitirá generalizar el modelo FOSS a casi todas las instituciones y hogares de la isla, sin importar lo atrasadas que sean sus condiciones técnicas.

RECOMENDACIONES.

- A los estudiantes y profesores de las Universidades del país se recomienda el uso de Guano, en pos de crear una corriente de retroalimentación que mejore los ciclos de desarrollo del producto.
- Que el presente trabajo se continúe desarrollando para implementar nuevas funcionalidades y mejorar su calidad.
- Con vista a contribuir en la migración del país, los proyectos, trabajos de curso y tareas independientes incluidos en el plan de clases de las carreras relacionadas con la informática deben exhortar a los alumnos a modificar, corregir e internacionalizar código de fuente perteneciente a proyectos reales del mundo de Software Libre.
- Se debe extender el uso de Software Libre en carreras relacionadas con la informática, no solo por las ventajas que brinda económicamente, si no por como contribuyen favorablemente la herramientas libres en el proceso de desarrollo de software.

REFERENCIA BIBLIOGRÁFICA.

1. Cubaminrex-Cuba en la Cumbre Mundial sobre la Sociedad de la Inform. *Ministerio de Relaciones Exteriores de Cuba.* [En línea] 16 de noviembre de 2005. [Citado el: 13 de enero de 2009.] http://www.cubaminrex.cu/Sociedad_Informacion/Cuba_SI/Informatizacion.htm.

2. **Levy, Carlos Miranda.** Reciclaje de Computadoras. *Educar.org y eAprender.org.* [En línea] 29 de Junio de 2005. [Citado el: 15 de enero de 2009.] http://portal.educar.org/foro/reciclajedecomputadoras.

3. **Valle, Amaury E. del.** Alertan sobre efectos y tendencias de la basura electrónica-Cuba-Juventud Rebelde-Diario de la Juventud Cubana. *Juventud Rebelde--Diario de la Juventud Cubana.* [En línea] 2002 de octubre de 2002. http://www.juventudrebelde.cu/cuba/2008-05-22/alertan-sobre-efectos-y-tendencias-de-la-basura-electronica/.

4. **Barrera, Moisés Rodriguez.** Curso ISLA: Linux Básico. *Cursos ISLA.* [En línea] 9 de noviembre de 2007. [Citado el: 16 de diciembre de 2008.] http://cisla.osl.ull.es/_media/octubre07/iniciacion/linux_basico.pdf?id=marzo08%3Ainiciacion%3Aapuntes&cache=cache.

5. **Garcia, David Alvarez.** *Entornos graficos de bajo consumo.* s.l. : Linux+, 2007.

6. Biblioteca Digital de Informática. *Biblioteca Digital de Informática.* [En línea] 2006. http://www.ispcmw.rimed.cu/sitios/digbiblio/#.

7. MasterMagazine. *MasterMagazine.* [En línea] 2004. http://www.mastermagazine.info/.

8. Portada-GleduWiki. *DAEP El sistema X WIndow.* [En línea] 10 de agosto de 2006. http://wiki.gleducar.org.ar/wiki/Portada.

9. **Fourdan, Olivier.** XFce - Desktop Environment. *XFce - Desktop Environment.* [En línea] 1996. http://www.xfce.org/.

10. LXDE.org. *LXDE.org.* [En línea] LXDE Foundation e.V., octubre de 2008. [Citado el: 2 de diciembre de 2008.] http://lxde.org/.

11. fluxbox.org. *fluxbox.org.* [En línea] 2001. http://fluxbox.org/.

12. **Gardner, Laurence.** Instalación Enlightenment (e17) en Debian/Ubuntu. *CacahueTUX… Otro de Linux.* [En línea] http://cacahuetux.wordpress.com/2008/03/13/instalacion-enlightenment-e17-en-debianubuntu/.

13. **Jen, Hong.** PCMan File Manager. *PCMan File Manager.* [En línea] SourceForge.net. http://pcmanfm.sourceforge.net/.

14. krusader. *krusader.* [En línea] 2001. [Citado el: 2 de febrero de 2009.] http://krusader.org/.

15. XFE Home Page. *XFE.* [En línea] [Citado el: 2 de febrero de 2009.] http://roland65.free.fr/xfe/.

16. Main Page Openbox. *Main Page Openbox.* [En línea] 2007. [Citado el: 5 de febrero de 2009.] http://icculus.org/openbox/index.php/Main_Page.

17. Red Hat Enterprise Linux 5.0.0. *Red Hat Enterprise Linux 5.0.0.* [En línea] 2007. [Citado el: 7 de febrero de 2009.] http://www.redhat.com/docs/manuals/enterprise/RHEL-5-manual/es-ES/Deployment_Guide/index.html.

18. **Carrascal, José Luis Lara.** Utilidades de Escritorio - LXPanel. *Utilidades de Escritorio - LXPanel.* [En línea] 2006. http://manualinux.my-place.us/lxpanel.html.

19. **KDE, El equipo de documentación de.** La guía de usuario de KDE. *La guía de usuario de KDE.* [En línea] 2004. http://docs.kde.org/stable/es/kdebase-runtime/userguide/index.html.

20. **Barrio, C. López.** *Metodología de Desarrollo:Programación Extrema.* 2005.

21. **Servetto, Lic. Arturo Carlos.** *Ambiente Integrado de Ingeniería Automática de Sistemas.* Buenos Aires : s.n.

22. **Camilo Javier Solis Álvarez, Roberth Gustavo Figueroa Díaz.** *Metodologías Tradicionales vs. Metodologías Ágiles.* Loja : s.n.

23. **Beck, Kent.** *Extreme Programming Explained. Embrace Change.* 1999.

24. **José H. Canós, Patricio Letelier y Ma Carmen Penadés.** *Métodologías Ágiles en el Desarrollo de Software.* Valencia : s.n.

25. **Hernán., Schenone Marcelo.** *Diseño de una Metodología Ágil de Desarrollo de Software.* Buenos

Aires : s.n., 2004.

26. **Romero, Gladys Marsi Peñalver.** *Metodología ágil para proyectos de software libre.* Habana : s.n., 2008.

27. **Espino, José Miguel Santos.** *Introducción al lenguaje C.* [Documento]

28. **Stroustrup, Bjarne.** *El lenguaje de programación C++.* Madrid : Addison Wesley, 1998. ISBN 84-7829-019-2.

29. **H, Swaroop C.** *A Byte of Python.* 2004.

30. Overview-Python v3.0. *Python Programming Language -- Official Website.* [En línea] Python Software Foundation, 2008. [Citado el: 25 de enero de 2009.] http://www.python.org/.

31. **Rizzo, Pablo Manuel.** Referencias Indirectas de Variables en Bash. *Pablo Manuel Rizzo |Main/Web Personal.* [En línea] 2008. [Citado el: 2009 de febrero de 2.] http://pablorizzo.com/Articulos/BashReferenciasIndirectasDeVariables.

32. **Council, National Research.** Funding a Revolution: Government Support for Computing Research. *National Academies Press.* [En línea] 1999. [Citado el: 5 de Noviembre de 2008.] http://www.nap.edu/catalog/6323.html. ISBN: 0-309-06278-0.

33. **Sommerville, I.** *Software engineering.* s.l. : Addison-Wesley Pub Co, 2000.

34. **Sametinger., J.** *Software engineering with.* s.l. : Springer Verlag, 1997.

35. **J. Sodhi, P. Sodhi.** *Software reuse: Domain analysis ande design process.* s.l. : McGraw-Hill, 1999.

BIBLIOGRAFÍA.

1. Castro, Fidel. Mentiras deliberadas, muertes extrañas y agresión a la economía mundial. [En línea]. Septiembre 2007 Disponible en: http://www.granma.cubaweb.cu/secciones/reflexiones/esp-049.html
2. Álvarez, David. Entorno gráfico de bajo consumo. [Libro digital en formato PDF]. Disponible en: http://casidiablo.net/entornos-graficos-de-bajo-en-linux/
3. Entornos de escritorios y gestores de ventanas más populares. [En línea]. Octubre 2007. Disponible en: http://facilware.es/?p=18
4. Wikipedia. GNOME. [En línea]. Mayo 2008. Disponible en: http://enciclopedia.us.es/index.php/GNOME
5. Sardon, Enrique. Manejadores de Ventanas Livianos en Linux. [En línea]. Disponible en: http://supra-net.org/158/manejadores-de-ventanas-livianos-en-linux/
6. Thinway. LXDE, un entorno de escritorio ligero para Linux. [En línea]. Diciembre 2006. Disponible en: http://bitelia.com/2006/12/19/lxde-un-entorno-de-escritorio-ligero-para-linux/
7. Wikipedia. Openbox. [En línea]. Octubre 2008. Disponible en: http://es.wikipedia.org/wiki/Openbox.
8. Huijsmans, Jasper. Sitio Web de Xfce. [En línea]. Año 1996-2008. Disponible en: http://www.xfce.org/projects/xfce4-panel/
9. Lara, José Luís. Administradores de Archivos XFE. [En línea]. Año 2006-2008. Disponible en: http://manualinux.my-place.us/xfe.html

ANEXOS

Anexo 1: Costo de uso por concepto de las licencias de Windows.

Cálculo aproximado de costo de uso por concepto de las Licencias de Windows.

Las licencias de Windows varían entre $49 y $2000 USD con un costo promedio (CP) de $150.

En Cuba existen alrededor de 509 000 PC.

$$CL(Costo\ de\ Licencia) = CP * PCg$$

$$CL = 150 * \$509\,000$$

$$CL = \$76\,350\,000$$

Anexo 2: Lista de riesgos.

Riesgo	Tipo de Riesgo	Impacto	Descripción	Probabi-lidad	Efectos	Mitigación del riesgo
Estabilidad	Requerimientos	Calidad Funcionalidad Diseño Cronograma	Los requerimientos pueden cambiar en las diferentes etapas del proyecto.	media	serios	-Realizar una buena captura de requisitos aplicando diferentes estrategias. -Brindarle a los clientes diferentes perspectivas y diferentes soluciones a un problema para evitar “sorpresas” e insatisfacciones. -Aclararles a los clientes los impactos negativos que pueden efectuar estos cambios sobre todo a la calidad y el cronograma del proyecto.
Mantenibili-dad	Especialidades de ingeniería	Calidad Funcionalidad	La implementación es difícil de	bajo	insignific ante	-Establecer estándares de código.

			entender y de mantener.			-Realizar revisiones técnicas. -Utilizar herramientas automáticas para el chequeo del código fuente y el ajuste a los estándares de codificación. -Comentar todo el código generado. -Generar toda la documentación asociada al código de la aplicación.
Fiabilidad	Especialidades de ingeniería	Calidad Funcionalidad Diseño	Los requerimientos de fiabilidad son difíciles de encontrar	alto	serio	-Registro y control de errores. -Registro y control del tiempo de corrección de errores.
Robustez	Especialidades de ingeniería	Calidad Funcionalidad Diseño	Los requerimientos de robustez son difíciles de satisfacer.	bajo	tolerable	-Realizar un estudio de las posibles interrupciones y comportamientos inesperados y como debe reaccionar el sistema frente a ellos.
Seguridad	Especialidades de ingeniería	Calidad Funcionalidad Diseño Cronograma	No existe experiencia suficiente en seguridad de aplicaciones.	medio	serio	-Realizar talleres y cursos donde se trate el tema de seguridad en específico para tipo de aplicación que se desarrolla.
Familiaridad	Sistema de desarrollo	Calidad Cronograma	La metodología no es familiar para los miembros del proyecto.	alto	tolerable	Explicar en etapas tempranas a todos los miembros del proyecto cada uno de los procesos y mostrar ejemplos hipotéticos.
Docencia	Entorno de trabajo	Cronograma	Carga docente de los integrantes del proyecto	alto	tolerable	-Elaborar un horario de producción tanto para todos los miembros del proyecto.

						-Realizar planificaciones teniendo en cuenta la carga docente y las horas de trabajos que tienen los miembros del equipo.
Actividades extracurricu-lares	Entorno de trabajo	Cronograma	Participación en actividades extracurriculares. (marchas, juegos deportivos, festivales)	alto	tolerable	-Tener en cuenta cuantos miembros del proyecto participan en estas actividades, revisar las planificaciones y ajustarlas.
Perdida	Recursos	Calidad Funcionalidad Diseño Cronograma	Perdida de integrantes del equipo de desarrollo	medio	serios	-Tener a más de una persona con conocimientos de los procesos y las actividades claves del proyecto que pueden verse afectadas en mayor medida si se pierde a un integrante considerado prácticamente indispensable.
Roturas	Recursos	Cronograma	Rotura de los componentes de hardware. (computadoras, módems)	medio	serios	-Cumplir con diferentes medidas de seguridad y cuidado de los medios para alargar su vida útil.
Energía	Recursos	Cronograma	Fallas eléctricas.	alto	Tolerable	-Informar a los miembros del equipo sobre posibles interrupciones planificadas.
Tecnológico	Recursos	Tiempo Desarrollo	No existe computadoras para todo el personal	alto	serias	-Tratar de obtener un Laboratorio para el proyecto
Personal	Personal	Retraso en la entrega de productos	Ausencia de un programador, analista, documentador o un diseñador.	medio	Serios	-Tratar de establecer las tareas de cada cual sin crear dependencia extrema de un trabajo con otro y teniendo más de

						una persona preparada en cada tema
Cooperación	Entorno de trabajo	Calidad Funcionalidad Diseño Cronograma	Los miembros del equipo no cooperan entre ellos, no existe espíritu de equipo.	medio	tolerable	-Agrupar el personal según su afinidad. -Seleccionar jefes que ejerzan verdadero liderazgo en el equipo.

Anexo 3: Tabla comparativa de los Entornos de Escritorio según sus funcionalidades principales.

Funcionalidades	Entornos de Escritorio			
	XFCE	LXDE	Fluxbox	Enlightement
Cambiar la apariencia y configuración del escritorio.	X	X	X	X
Funcionalidades para la interacción con dispositivos extraíbles.	X	X		
Navegar y explorar distintos protocolos de red con el mismo explorador.				
Funcionalidades para establecer conexiones con otras PC de la red.				
Accesos rápidos de teclado.	X	X	X	X
Funcionalidades para compartir carpetas e impresoras en mi red.				
Tener el Panel de Control.	X			
Barra de accesos rápidos.	X	X		X
Conservar la estructura del menú y que recuerde las aplicaciones recientes y preferidas.				
Papelera de reciclaje.	X			

Glosario de Términos

B: Nombre de un lenguaje de programación desarrollado en los Bell Labs, predecesor del lenguaje de programación C.

BCPL: Sigla en inglés de Basic Combined Programming Language (Lenguaje de Programación Básico Combinado).

Blackbox: Es un gestor de ventanas minimalista para sistemas de tipo UNIX, escrito completamente de cero por Brad Hughes bajo el lenguaje de programación C++.

Bourne shell (sh): Uno de los primeros intérpretes importantes de Unix.

D-BUS: Es el nombre de un sistema software que proporciona una forma simple de comunicación entre distintas aplicaciones, desarrollado como parte del proyecto freedesktop.org.

Fbpanel: Es un panel de escritorio.

Framework: es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Freedesktop: Es un proyecto que trabaja por la interoperatibilidad y la compartición de la tecnología base de los entornos de escritorio para X Window System (X11), tanto en Linux como en otros sistemas operativos Unix.

FTP (sigla en inglés de File Transfer Protocol - Protocolo Transferencias Archivos): Es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP.

GDM: (GNOME Display Manager) es un gestor de acceso para el X Window System. Es el reemplazo, elaborado por el proyecto GNOME, al XDM básico.

Gentoo: Gentoo Linux es una distribución GNU/Linux orientada a usuarios con cierta experiencia en este sistema operativo, fue fundada por Daniel Robbins, basada en la inactiva distribución llamada Enoch Linux.

Gestor de acceso: Es una parte opcional del sistema X Window que se usa para el manejo de sesiones.

GIO: Nueva biblioteca compartida que es parte de GLib y proporciona la API para gvfs.

GLib: GLib es una biblioteca de propósito general que se usa para implementar muchas funciones no gráficas. Aunque GTK+ lo necesita, GLib puede usarse de forma independiente. Por eso, algunas aplicaciones usan GLib sin usar la biblioteca GTK+.

GNOME: Entorno de escritorio para sistemas operativos de tipo Unix bajo tecnología X Window.

GLOSARIO DE TÉRMINOS.

GNU/Linux: GNU/Linux es, a simple vista, un Sistema Operativo. Es una implementación de libre distribución UNIX para computadoras personales, servidores, y estaciones de trabajo. Fue desarrollado para el i386 y ahora soporta los procesadores i486, Pentium, Pentium Pro y Pentium II, así como los clones AMD y Cyrix. También soporta máquinas basadas en SPARC, DEC Alpha, PowerPC/PowerMac, y Mac/Amiga Motorola 680x0. Como sistema operativo, GNU/Linux es muy eficiente y tiene un excelente diseño.

GPL: La GNU General Public License (inglés: Licencia Pública General) es una licencia creada por la Free Software Foundation y orientada principalmente a los términos de distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es Software Libre.

GTK+: GTK+ o The GIMP Toolkit es un conjunto de bibliotecas multiplataforma para desarrollar interfaces gráficas de usuario (GUI), principalmente para los entornos gráficos GNOME, XFCE y ROX aunque también se puede usar en el escritorio de Windows, MacOS y otros. GTK+ se ha diseñado para permitir programar con lenguajes como C, C++, C#, Java, Ruby, Perl, PHP o Python.

Gvfs: Es un sistema de archivos virtual de espacio del usuario en el que mount se ejecuta como un proceso separado que le permite comunicarse por medio de D-Bus. También contiene un módulo GIO que proporciona, de forma integrada, soporte Gvfs para todas las aplicaciones que usen el API GIO. También permite exponer Gvfs a las aplicaciones no-GIO que usen fuse.

Hal: Es una capa de abstracción de hardware software que permite a las aplicaciones de escritorio tengan acceso a información de hardware.

Hald: Demonio encargado de gestionar los eventos en la capa de abstracción Hal.

Historias de Usuario HU: Secuencias de acciones que el sistema puede llevar a cabo interactuando con sus actores, incluyendo alternativas dentro de las secuencias.

Human interface guidelines (HIG): Son documentos que ofrecen a los desarrolladores de aplicaciones un conjunto de recomendaciones destinadas a mejorar la experiencia para los usuarios, haciendo interfaces de uso más intuitivos, aprendibles, y constantes.

Interfaces (API): API (del inglés Application Programming Interface - Interfaz de Programación de Aplicaciones) es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Interfaz gráfica (GUI): En el contexto del proceso de interacción persona-ordenador, la interfaz gráfica de usuario, es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.

KDM: (K Display Manager) es un gestor de acceso para el X Window System. Es el reemplazo, elaborado por el proyecto KDE, al XDM básico.

KDE: Entorno de Escritorio e infraestructura de desarrollo para sistemas Unix/Linux.

GLOSARIO DE TÉRMINOS.

Libre: O Software Libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software.

Librería: Término informático para referirse a las bibliotecas de vínculos dinámicos.

Metodología: Un conjunto de métodos eficientes orientados a conseguir un objetivo propuesto. Son un conjunto de procesos que organizados dan una secuencia de pasos a seguir para obtener los hitos propuestos y finalmente el producto.

Midnight Commander (abreviado mc): Es un administrador de archivos ortodoxo para sistemas Unix y derivados (también existe para Windows).

Minimalista: El término minimalista, en su ámbito más general, es referido a cualquier cosa que se haya desnudado a lo esencial, despojada de elementos sobrantes, o que proporciona solo un esbozo de su estructura, y minimalismo es la tendencia a reducir a lo esencial.

Rendimiento: Un término usado en la informática para dar medida de cuán eficaz puede ser un software en el uso de los recursos de hardware.

Shell: Una shell es un procesador de macros que ejecuta comandos.

Sistema operativo: Un sistema operativo es un software de sistema, es decir, un conjunto de programas de computadora destinado a permitir una administración eficaz de sus recursos. Comienza a trabajar cuando se enciende el computador, y gestiona el hardware de la máquina desde los niveles más básicos, permitiendo también la interacción con el usuario.

SLIM: Es un gestor de acceso ligero.

Terminal: Término derivado del inglés que significa consola.

UNIX: Unix (registrado oficialmente como UNIX®) es un sistema operativo portable, multitarea y multiusuario; desarrollado, en principio, en 1969 por un grupo de empleados de los laboratorios Bell de AT&T.

XDM: Es el gestor de acceso del sistema X Window.