UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD I

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

Título: Módulo de arranque de estaciones ligeras para OSplugger.

Autores

Leiser Pérez Matos

Sergio Alejandro Velázquez Leyva

Tutor

Ing. Mijail Hurtado Fedorovich

Ciudad de la Habana, Cuba, junio 2011

DECLARACIÓN DE AUDITORÍA

Declaramos que somos los únicos 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 firmamos la presente a los días del mes del año .

_____

Ing. Mijaíl Hurtado Fedorovich

_____ _____

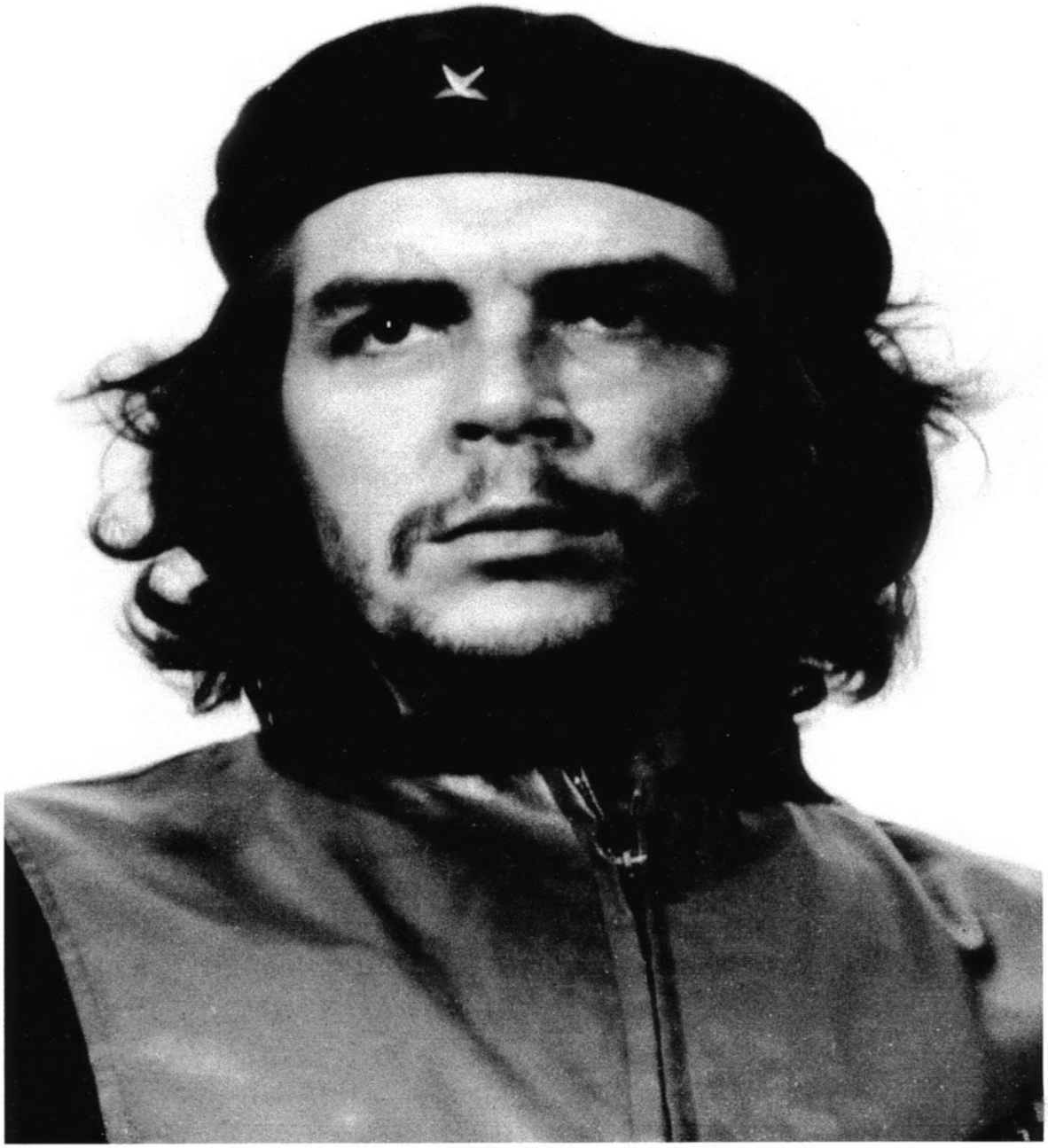
Sergio Alejandro Velázquez Leyva. Leiser Pérez Matos

DATOS DE CONTACTO DEL TUTOR

Nombre y Apellidos: Ing. Mijaíl Hurtado Fedorovich

Email: mfedorovich@uci.cu

Síntesis del Tutor: Profesor graduado en Ingeniería de las Ciencias Informáticas en el año 2008, en la Universidad de las Ciencias Informáticas. Profesor instructor. Líder del proyecto OSplugger, cofundador y desarrollador de Nova.



"La única lucha que se pierde es la que se abandona."

Ernesto Che Guevara.

Agradecimientos

Dedicatoria

Índice

Resumen

En la actualidad Cuba se encuentra inmersa en la compra de clientes ligeros puesto que en el mercado mundial son vendidos más baratos que las computadoras convencionales y es por ello que Cuba siendo un país bloqueado y limitado por el imperio yanqui se ve en la necesidad de buscar alternativas para el desarrollo informático de la isla. La Universidad de las Ciencias Informática (UCI) en el proyecto Nova se encuentra un subproyecto llamado Osplugger que no tiene la vía de configurar y automatizar el arranque de los clientes ligeros siendo muy engorroso poner en funcionamiento los mismos ya que se dependería de un personal que no solo tenga conocimientos básicos de GNU/Linux, sino conocimientos avanzados en el tema previsto puesto que la configuración sería a puras líneas de comandos, la solución propuesta ahorraría todo este trabajo. Por las razones antes señaladas los autores de este trabajo de diploma se propusieron como objetivo principal desarrollar un módulo de arranque que permita el arranque de los clientes ligeros de forma automatizada y dinámica utilizando tecnologías libres. Para la realización de este trabajo se utilizaron los métodos científicos el histórico-lógico, analítico-sintético y modelación. El resultado final logrado por los autores fue la liberación de la versión 1.0 del módulo de arranque para OSplugger que permite configurar y automatizar el arranque de los clientes ligeros pudiendo ser utilizado en cualquier institución del país. Cada una de las funcionalidades del módulo propuesto fue evaluada obteniéndose resultados satisfactorios.

Introducción

Desde el surgimiento de los primeros ordenadores se ha venido evolucionando en la forma de iniciar los sistemas operativos de los mismos. Los Sistemas Operativos tienen más de una forma de iniciar o arrancar mediante discos floppy, CD o DVD, dispositivos de almacenamiento USB, a través de los típicos discos duros montados localmente en los computadores e incluso desde una red cableada o inalámbrica.

En los ordenadores modernos, el proceso de arranque comienza con la Central Processing Unit (CPU) ejecutando los programas contenidos en la memoria Read-Only Memory (ROM) en una dirección predefinida (se configura la CPU para ejecutar este programa, sin ayuda externa, al encender el ordenador). El inicio por la red, básicamente consiste en que un ordenador posee una rutina de arranque en una memoria permanente, por ejemplo en una Read-Only Memory (ROM), que le permite contactar con el servidor y obtener el sistema de ficheros a través de un enlace de red. Para poder arrancar por red el ordenador debe obtener: 1º una identidad, 2º la imagen de un sistema operativo y 3º normalmente, un sistema de ficheros con el que trabajar.

El cliente ligero, es un ordenador que no posee componentes de almacenamiento interno. Los mismos son diferentes pero relativamente pasivos y de bajo mantenimiento, son fáciles de instalar y operar. El hardware para los clientes ligeros es por lo general económico al no contener discos duros, memoria de aplicaciones y un procesador potente. Por otro lado, desde la perspectiva del usuario, la interacción por medio del monitor, el teclado, y el ratón cambia poco en cuanto al uso de un cliente pesado (cómputo con capacidad de almacenar datos y procesarlos).

Años atrás, Cuba enfrentó el dilema de que miles de computadoras ubicadas principalmente en todas las escuelas, instalaciones de la salud y otras entidades, comenzaron a fallar. Se transitaba por el envejecimiento tecnológico, los cortes de electricidad y hasta el maltrato de los usuarios. También cientos de computadoras autónomas con configuraciones completas, eran ineficientemente aprovechadas en muchos sectores, lo que indicaba un innecesario sobredimensionamiento del hardware. El modelo de informatización de la sociedad cubana, basado en máquinas convencionales, mostró sus debilidades siendo poco eficaz, limitado y muy costoso a la hora de adquirir hardware de última generación.

INTRODUCCIÓN

Ante la necesidad de dar continuidad al proceso de una manera más racional, especialistas del Ministerio de la Informática y las Comunicaciones concibieron el proyecto de desarrollo de clientes ligeros, a partir de la experiencia internacional en este campo y acorde a las posibilidades de una economía en vías de desarrollo y bloqueada por el imperialismo yanqui. El uso de los clientes ligeros sin duda alguna es una alternativa importante para el proceso de informatización de la sociedad cubana siendo hoy en día una prioridad para el país.

Actualmente en la UCI, existe un proyecto llamado OSplugger en el que se desarrolla una herramienta de administración, gestión y control de clientes ligeros o máquinas sin disco de almacenamiento que permite compartir prácticamente cualquier sistema operativo por la red. El mismo carece de una vía para configurar el arranque de las estaciones ligeras de forma automatizada y dinámica, por lo que es engorroso realizar este proceso, pues para poner en funcionamiento un cliente ligero hay que configurarlo de forma manual acarreando demora y deficiencia con el servicio del mismo.

Teniendo en cuenta las deficiencias antes planteadas se declara el siguiente **problema científico:** ¿Cómo lograr que OSplugger configure el arranque de las estaciones ligeras de forma automatizada y dinámica?

Para comprender los límites de este trabajo de diploma se define como **objeto de estudio:** el funcionamiento de los clientes ligeros, delimitando como **campo de acción:** la configuración del arranque de los clientes ligeros.

Para el correcto desarrollo de esta investigación se define como **objetivo general**: Desarrollar un módulo de arranque para estaciones ligeras administradas por OSplugger.

Con el fin de darle cumplimiento al mismo se trazan los siguientes **objetivos específicos:**

- Analizar el proceso de configuración de arranque de los clientes ligeros.
- Seleccionar herramientas, lenguajes y tecnologías necesarias para el desarrollo del trabajo.
- Elaborar el análisis y diseño de la propuesta de solución.
- Implementar el módulo propuesto y verificar los resultados obtenidos mediante pruebas de funcionalidad.

El trabajo de diploma se sustenta en la siguiente **idea a defender:** El desarrollo de un módulo de arranque para estaciones ligeras administradas por OSplugger, puede posibilitar el uso eficiente de los clientes ligeros flexibilizando el arranque de los mismos.

Con el propósito de dar validez a lo anteriormente planteado se formulan las siguientes tareas científicas:

- Elaboración del marco teórico a partir del estado del arte del tema existente en la actualidad a nivel mundial para alcanzar un mayor conocimiento sobre los clientes ligeros y su configuración de arranque.

- Selección de la metodología de desarrollo, las tecnologías, herramientas y lenguajes adecuados para el desarrollo de la solución.

- Construcción del diseño de la propuesta a implementar para darle facilidades al desarrollador a la hora de implementarla.

- Implementación del módulo propuesto para darle solución a los requisitos que se definan en el transcurso de la investigación.

- Ejecución de los casos de prueba de funcionalidad para asegurar la calidad del resultado final.

Para la realización de este trabajo de tesis se utilizaron diferentes métodos científicos, ellos constituyen un *"Conjunto de reglas que señalan el procedimiento para llevar a cabo una investigación"* **[1].** Este conjunto de reglas parten de principios claros, razonables e incuestionables, que sirven para dar validez a las reglas del método científico.

Los métodos científicos a utilizar son:

Analítico-Sintético: Divide el objeto de estudio en conceptos que son examinados por separado, estudiándose rigurosamente cada uno de ellos de manera independiente, y confrontando el criterio de disímiles autores, y las correspondencias entre ellos. Se realiza un análisis sobre los clientes ligeros, dividiéndose específicamente en la configuración de su sistema de arranque y en los protocolos usados para la transferencia de datos.

También se analiza las diferentes herramientas, tecnologías y metodología que son necesarias para el desarrollo de este módulo, de las mismas se tendrá en cuenta los

aspectos que la definen y que de alguna forma las hace desiguales, tratando de determinar las características genéricas de las mismas y sus mejoras potenciales.

Modelación: para una mejor comprensión del módulo a desarrollar, se modelan los diferentes procesos o actividades a través de la herramienta CASE para representar los diferentes diagramas UML.

Histórico–Lógico: permite la revisión bibliográfica de los colectivos de autores especializados en el tema de Gestores de Arranque utilizando estos como punto de referencia y analizando los resultados alcanzados. Viendo así las diferentes tendencias y estrategias actuales en dicho tema.

El presente trabajo de diploma se divide en 3 capítulos, a continuación se presentan sus nombres y objetivos de forma general:

Capítulo # 1: "Fundamentación teórica de la configuración de arranque en los clientes ligeros". El objetivo de este capítulo es analizar conceptos generales y básicos que permiten comprender temas relacionados con los gestores de arranque existentes en la actualidad.

Capítulo # 2: "Planificación y definición del módulo de arranque para OSplugger". Este capítulo tiene como principal objetivo delimitar la modelación del negocio y la selección de los requerimientos. Para dar cumplimiento a lo precedentemente planteado se tienen en cuenta la herramienta, lenguaje y metodología a utilizar, se explica el flujo actual de los procesos y se definen las historias de usuarios y la lista de reserva del producto a desarrollar.

Capítulo # 3: "Implementación y pruebas del módulo de arranque para OSplugger". El objetivo de este capítulo es detallar como los elementos del modelo de diseño se implementan en términos de componentes. Además de validar y probar el módulo de acuerdo a los requisitos que debe cumplir.

Capítulo1: Fundamentación teórica de la configuración de arranque en los clientes ligeros

En el desarrollo de este capítulo se exponen datos, comparaciones e informaciones resultantes de la investigación realizada sobre todo lo relacionado con el proceso de arranque en las computadoras, profundizando en el proceso de arranque mediante la red para los clientes ligeros. Se estudia los distintos gestores de arranque existentes en el mundo tanto para SO Windows como Linux mostrando una visión general de los conceptos vinculados a los mismos.

1.1 Proceso de Arranque.

La secuencia o proceso de arranque (boot) es el camino que se realiza desde que se enciende el ordenador hasta que se carga el sistema operativo. La fase de arranque se considera completa cuando el equipo está preparado para dar respuesta a las órdenes del usuario.

Al encenderse el equipo, la CPU (Unidad Central de Procesamiento) ejecuta el código que se encuentra en la BIOS (Sistema Básico Entrada/Salida), ya sea cargándolo primero en la memoria principal o leyéndolo directamente desde la memoria de la BIOS. Este código se encarga de realizar la rutina POST (Power On Self Test), que verifica la integridad de la memoria, los controladores, y dispositivos del sistema. Después se realiza la práctica de arranque, la BIOS busca un código de arranque en el dispositivo de almacenamiento indicado (en BIOS o por opción del usuario), lo carga en memoria y transfiere el control del equipo a éste.

Para localizar el código de arranque, el BIOS lee el primer bloque del dispositivo y si tiene la firma adecuada, lo carga y le cede el control de la CPU. En dispositivos particionados, este primer bloque es un MBR (Registro Maestro de Arranque), en dispositivos no particionados es un VBR (Volumen de Registro de Inicio). La única función del "Código de Arranque Maestro" (Master Boot Code(MBR)) es localizar en su tabla de particiones una partición con la marca (flag) de arranque, verificar que la firma del primer registro es correcta, cargarlo en memoria y cederle el control.

Este sector inicial es un Volumen de Registro de Inicio (VBR) también llamado en este caso PBR (Partición Registro de Inicio). El código del VBR es un "cargador de sistema" (boot loader) o un "gestor de arranque" (boot manager) que se encarga de

iniciar la carga de un sistema operativo. El código del MBR y el VBR utiliza instrucciones en código máquina. El "Código de Arranque Maestro" del MBR y el VBR puede ser sustituido por un gestor de arranque, o una parte del mismo.

Una vez que el cargador de sistema tiene el control, se ocupa de que comience la carga del sistema operativo anteriormente especificado cediendo el control al núcleo del sistema operativo. El mismo se encarga de obtener información sobre el equipamiento (hardware) del sistema, así como los controladores (drivers) asociados a los dispositivos.

1.2 Gestores de Arranque.

Habitualmente se mezclan los términos "gestor de arranque" (boot manager) y "cargador de sistema" (boot loader), debido a que aplicaciones como GRUB (Gran Gestor de Arranque Unificado) brindan ambas funciones. Sin embargo existe una marcada diferencia entre estos, ya que un cargador de sistema es un programa sencillo diseñado exclusivamente para cargar en memoria un sistema operativo como su nombre lo indica.

Un gestor de arranque en cambio es un programa que permite opciones previas a la carga del sistema. Así puede por ejemplo, permitir el arranque desde dispositivos no detectados en el inicio o desde particiones sin la marca de arranque. También puede ofrecer al usuario opciones de arranque o incluso cargar otros gestores de arranque, incluyendo las funciones de cargador de sistema.

Un gestor de arranque puede sustituir el "Código de Arranque Maestro" del MBR, aunque dada la limitación de espacio en el MBR, pueden instalar solo una parte o etapa (para los que poseen arranque multi-etapas). Esta primera etapa se ocupa de cargar el resto del programa. Algunos de los más conocidos son GRUB, LiLo, NTLDR (todos ellos incluyen la función de cargador de sistema) y BootMgr (llama al cargador de sistema WinLoad).

Llegado a este punto se puede observar que los gestores de arranque pasan a formar parte esencial a la hora de facilitar el arranque y la interacción con el usuario durante el mismo. Es esta característica la que hace realidad la existencia de diversos gestores de arranque para los diferentes sistemas operativos y plataformas. En la familia de los sistemas de Windows hay varios gestores, unos más versátiles o más usados que otros, entre esos se pueden mencionar los siguientes:

- NTLDR (NT Loader)

Es el gestor de arranque usado en Ms-Windows desde la versión NT hasta XP y 2003. Está diseñado para sistemas que únicamente disponen de instancias de Windows. Según este diseño, el "Código Maestro de Arranque" del MBR levanta mediante carga encadenada ("chain load") el código del VBR (Volumen de Registro de Inicio) de la partición de Windows. Este código del VBR busca en la raíz de la partición de sistema el fichero NtLdr (C:\NtLdr). Una vez en memoria NtLdr puede ofrecer al usuario seleccionar entre distintas instancias de Windows como gestor de arranque- basándose en el fichero de configuración boot.ini y cargar la opción elegida -cargador de sistema-.

- BOOTMGR

El sistema operativo Ms-Windows a partir de la versión Vista utiliza BootMgr como gestor de arranque y WinLoad como cargador de sistema. Igual que NtLdr, está diseñado para sistemas que únicamente disponen de instancias de Windows, aunque puede configurarse para lanzar otros cargadores mediante "chainload (cadena de carga)". Sin embargo, las herramientas de configuración del gestor de arranque de Microsoft no facilitan incluir entradas de este tipo. Nuevamente según este diseño, el "Código Maestro de Arranque" del MBR levanta mediante carga encadenada ("chain load") el código del VBR de la partición de Windows. Este código se encarga de buscar en la raíz de la partición de sistema el fichero BootMgr (C:\BootMgr).

El gestor de arranque BootMgr lee un fichero de configuración llamado bcd y llama al cargador de sistema de Windows (WinLoad.exe) que reside en la partición del sistema dentro del directorio de sistema (\windows\system32\winload.exe).

- Acronis OS Selector

Gestor de arranque potente, fiable y fácil de usar. Permite instalar 100 o más sistemas operativos en el equipo. Resiste arrancar un SO de cualquier partición en un disco duro o tener diversos sistemas operativos en la misma partición. Soporta diversidad de sistemas operativos, incluyendo distribuciones de Linux. Permite editar archivos de configuración sin necesidad de iniciar el sistema operativo, es compatible con sistemas operativos en particiones primarias y lógicas de cualquier disco duro. Posee además una interfaz intuitiva al estilo Windows XP con una ayuda online y la habilidad de

detectar un sistema operativo que se vuelve no arrancable debido a razones accidentales. Es necesario aclarar que no permite el arranque desde la red.**[2]**

- DualBootPRO

Posee una interfaz de fácil uso para todo tipo de usuarios pudiéndose hacer cambios en el BCD (Boot Configuration Data). Permite cambiar el orden del menú de arranque, renombrar el nombre de las entradas de los sistemas operativos. Es compatible con las versiones de Windows 7, Windows XP, además trabaja con Windows Server 2003 y Windows Server 2008.Limita la cantidad de memoria que se le asigna a un sistema operativo y es un software privativo. **[3]**

Para la familia de UNIX existe también diversidad de gestores de arranque entre los que se encuentran:

- LILO (Linux Loader)

Este no depende de ningún sistema de archivos en específico y puede arrancar imágenes de Linux desde un disco flexible externo o desde el disco duro. Puede instalarse en el máster boot record (MBR). Ofrece versatilidad y permite cargar más de un sistema operativo. Llegó a ser el gestor de arranque en las primeras versiones de Linux, pero en la actualidad es la segunda opción, dejando a GRUB como primera opción.**[4]**

- GNU/Grub (GRand Unifier Bootloader)

Se está en presencia de un gestor de arranque múltiple, desarrollado por el proyecto GNU, derivado del GRand Unified Bootloader (GRUB en español: Gran Gestor de Arranque Unificado). Es usado comúnmente para iniciar uno de dos o más sistemas operativos instalados en un mismo equipo. Es un gestor de arranque que brinda mayor limpieza, seguridad, robustez y poder a la hora de iniciar un sistema operativo.

GRUB es capaz de no necesitar una instalación de una partición o un núcleo nuevo, logrando cambiar todos los parámetros en el arranque mediante el uso de órdenes de consola de GRUB. Puede examinar un sistema de archivos y actualmente soporta gran variedad de sistemas de archivos como por ejemplo; ext2/ext3/ext4(GRUB 2). Estos archivos son usado por los sistemas Unix y su variante libre GNU/Linux, ReiserFs, XFS de SGI (aunque puede provocar problemas), UFS, VFAT como FAT16

y FAT32 usados por Windows 9.x, NTFS usado por los sistemas Windows NT (a partir de Windows NT v.3.51) entre otros.

- Proyecto Syslinux **[5]**

El Proyecto SYSLINUX abarca un conjunto de gestores de arranque ligeros, para arrancar ordenadores en el sistema operativo Linux. Ellos son Isolinix, ExtLinux y PXELinux.

- Syslinux e Isolinux

Syslinux es normalmente usado para arrancar desde sistemas de archivos FAT. No se usa habitualmente para arrancar instalaciones de Linux completas ya que Linux no suele instalarse en sistemas de archivos FAT. En cambio, se utiliza con frecuencia para discos flexibles de arranque o de rescate, LiveUSBs (memoria USB que contiene un sistema operativo completo sobre el cual arranca el ordenador), u otros sistemas de arranque ligeros, e ISOLINUX que se usa para arrancar sistemas de archivos ISO 9660 CD-ROM. Además de crear LiveCDs de Linux o CDs de arranque inestables.

- ExtLinux

Este se usa para arrancar desde sistemas de ficheros de linux etx2, ext3, ext4 o btrfs. El soporte para btrfs y ext4 fue agregado en la versión 4 de EXTLINUX.

- PXELinux

Usado para arrancar desde un servidor de red con el sistema Preboot Execution Enviroment (PXE).
PXE (Entorno de ejecución de pre arranque), es un entorno para arrancar e instalar el sistema operativo en ordenadores a través de una red, de manera independiente de los dispositivos de almacenamiento de datos disponibles (como discos duros) o de los sistemas operativos instalados. PXE utiliza varios protocolos de red como DHCP y TFTP, de hecho PXE es una combinación de los protocolos DHCP y TFTP con algunas modificaciones en ambos. Es por ello que se usa DHCP para localizar el servidor de arranque apropiado y FTP para descargar el programa inicial bootstrap (cargador de inicialización para describir el arranque) y archivos adicionales. Este protocolo fue creado para soportar varias arquitecturas.

El entorno PXE utiliza DHCP o BOOTP para habilitar una conexión TCP/IP básica, y luego descarga un programa de arranque por TFTP. Este programa de arranque carga y configura el kernel de acuerdo a directivas que son obtenidas también desde el servidor TFTP. Típicamente, PXELINUX se usa para instalaciones de Linux desde un servidor de red central o para arrancar estaciones de trabajo sin disco.

GPXE (antes Etherboot)

Es una implementación de código abierto (bajo la licencia GPL) del Preboot Execution Enviroment (PXE) y cargador de arranque. Puede ser usado para habilitar a los computadores que no tienen soporte para el PXE para que puedan cargar desde la red, o para extender con soporte para protocolos adicionales a una implementación de PXE existente. Mientras que los clientes tradicionales de PXE usan TFTP para transferir datos, el gPXE añade la capacidad de recuperar datos a través de otros protocolos como HTTP, iSCSI y ATA sobre Ethernet (AoE). Puede además trabajar con WiFi en vez de requerir una conexión alámbrica.

Es capaz de leer archivos desde múltiples protocolos de red, tales como TFTP, NFS, HTTP2 o el FTP, y puede levantar imágenes PXE, Linux y Windows. Posee además la ventaja de ser scriptable (posee archivos de órdenes o de procesamiento por lotes) y puede cargar extensiones COMBOOT y COM32 de SYSLINUX, permitiendo construir un menú gráfico para arranque desde la red.

Ventajas que brinda gPXE:

- Incluye variedad de drivers para tarjetas de red, incluyendo la mayoría de uso común.
- Alta compatibilidad con redes inalámbricas (wireless) 802.11
- Encadenador de arranque compatible con drivers PXE.
- Soporte DNS para uso de nombres de host (hostnames).
- Variados protocolos de descarga soportados: TFTP, HTTP, HTTPS, FTP, NFS.
- Diversidad de formatos de imagen de Sistemas Operativos: ELF, COM32, PXE, Linux y otros.
- Posee una línea de comandos con amplio soporte de scripts.
- Si el driver de una tarjeta de red no soporta PXE para arrancar, se puede “quemar” una imagen del cargador de arranque en red gPXE:
  - En un chip dentro de una tarjeta de expansión

- En el mismo BIOS.
- En un disquete.
- En un CD/DVD.
- En una memoria USB.

Pasos del proceso de arranque con gPXE:

- El BIOS del computador cliente pregunta por su configuración IP y por la información de arranque en red vía DHCP.
- El servidor DHCP le provee al BIOS del cliente los parámetros de arranque necesarios tales como la dirección IP que le asigna, la dirección del servidor TFTP o HTTP dependiendo del protocolo que se haya elegido para el arranque.
- El cliente recupera la imagen ejecutable del cargador de arranque desde el servidor utilizando TFTP, HTTP u otro protocolo.
- El cliente ejecuta la imagen extraída.
- Dependiendo de la imagen de arranque y del archivo de configuración o script gPXE, el cliente empieza a solicitar la imagen del kernel y la del sistema de archivos raíz inicial. **[6]**

1.3 XML-RPC como protocolo de llamada a procedimiento remoto.

Durante la gestión del arranque de los clientes ligeros, se hace necesario establecer una comunicación cliente-servidor, esta comunicación es necesaria para el envío de datos entre el cliente y servidor en el momento del registro y una vez iniciado el cliente, para actualizar algunos cambios que se realicen a la información o configuración del cliente. Como respuesta a esta necesidad se utiliza el protocolo XML-RPC. XML-RPC es un protocolo de llamada a procedimiento remoto (RPC) que usa XML para codificar los datos y HTTP como protocolo de transmisión de mensajes.

Es un protocolo muy simple, define unos cuantos tipos de datos y comandos útiles, además de una descripción completa de corta extensión. La simplicidad del XML-RPC está en contraste con la mayoría de protocolos RPC que tiene una documentación extensa y requiere considerable soporte de software para su uso. Entre los tipos de datos disponibles se encuentran tipos de datos simples (enteros, booleanos, etc.) y tipos complejos (arrays, fechas, estructuras, datos binarios o el valor null). Una petición se compone del nombre del método llamado junto con los parámetros, y la respuesta son los datos de salida del método invocado.

1.3.1 Implementaciones.

De este protocolo existen diversas implementaciones para diversos ambientes y lenguajes para ahorrar labores como:

Configuraciones de parser, cuestiones de seguridad, integración a servidores de páginas y otros detalles secundarios.

- XMLRPC APACHE para el lenguaje "Java".

- FRONTIER: RPC para el lenguaje "Perl".

- XMLRPC-PHP para el lenguaje "PHP".

- XMLRPC-PYTHON para el lenguaje "Python".

- XMLRPC-C para los lenguajes "C" y "C++".

- XMLRPC-ASP para el lenguaje "COM/VBasic".

1.4 Arranque por la Red.

Hasta este momento se puede observar la diversidad de gestores de arranque para ambas plataformas, no solo es la diversidad de gestores de arranque lo que hay que valorar, sino también las formas en que se puede iniciar el ordenador, haciendo referencia en este punto a los diskettes, CD o DVD, discos duros, y uno de los más novedosos y más importantes para esta investigación, el arranque desde la red. El arranque por la red es un proceso en el cual se inicia un ordenador desde la red y no desde un disco local, y procede de la misma forma que utilizando gPXE.

1.4.1 Tecnologías de Arranque por la red.

Las tecnologías de arranque por la red han evolucionado con el paso de los años paralelamente al desarrollo de los ordenadores. A continuación se muestra un listado con los principales hitos en cuanto a la evolución de las tecnologías de arranque.

Evolución de las tecnologías de arranque en red:

- 1993 Network Boot Image (NBI): Transferencia del control al kernel y enlace de drivers.
- 1995 Etherboot: Compatible con imágenes NBI.
- 1997 Preboot eXecution Enviroment (PXE): Un servidor DHCP que especifica el nombre de archivo de un programa de arranque de red (NBP) descargado a través de un servidor TFTP, el cual a su vez continua descargando el kernel y demás imágenes del Sistema Operativo. PXE se convirtió en el estándar sacado a la luz por Intel en ese año.
- 1999 Continúa el desarrollo del proyecto Etherboot.
- 2000 Rom-O-Matic: Generador vía web de imágenes de arranque Ethernet en http://www.rom-o-matic.net/
- 2005 Marty Connor y Michael Brown reescriben el proyecto Etherboot dando lugar al nacimiento degPXE, una implementación FOSS de PXE.
- 2010 surge gPXE versión1.0. **[6]**

Por tanto durante el desarrollo de la investigación, y valorando las ventajas y dificultades que posee cada gestor de arranque en correspondencia al problema que se quiere solucionar, se hace una selección inicial de los sistemas Syslinux y GPXE. El segundo por su superioridad sobre el primero en cuanto a los protocolos soportados para la transferencia de datos con la incorporación de HTTP, iSCSI, AoE entre otros y la posibilidad de trabajar sobre redes WiFi, además de soportar las extensiones de Syslinux, vesamenú y menú para la construcción de un menú de selección, logrando realizar el arranque de manera más interactiva. Analizando lo anteriormente expuesto, se utiliza en la solución a PXElinux en todo lo referente al arranque hasta la construcción del menú de arranque buscando mayor velocidad , gPXE se utiliza por su gran variedad de protocolos soportados, para la descarga del sistema operativo y por último el protocolo XML-RPC para establecer la comunicación entre el cliente y el servidor tanto en el momento de registro del cliente como cuando sea necesario actualizar algún cambio en la configuración del cliente.

Capítulo 2: Definición y planificación del módulo de arranque para OSplugger

En el presente capítulo se describe el desarrollo del módulo de arranque de clientes ligeros para OSplugger. Se determinan los requerimientos funcionales y no funcionales del mismo. Se realiza el modelado del negocio y se definen las historias de usuarios, los actores y trabajadores que intervienen durante el desarrollo de la aplicación.

2.1 Requisitos para el arranque por la red

Durante la investigación se llega a la conclusión que un servidor PXE (junto con otros servidores) es el encargado en arrancar un equipo por la red, procedimiento muy útil para crear terminales ligeras.

El funcionamiento es el siguiente:

1. Se enciende el equipo.
2. La tarjeta de red trata de obtener una IP por DHCP e informa que es un cliente PXE.
3. El servidor DHCP le envía al equipo cliente su IP y el nombre de la imagen a usar para arrancar.
4. El equipo cliente obtiene la imagen a iniciar por TFTP.
5. El sistema operativo de la imagen se inicia. **[7]**

2.2 Herramientas, tecnologías y metodologías utilizadas para el desarrollo.

2.2.1. Lenguaje de programación para el desarrollo:

La Arquitectura del sistema OSplugger está diseñada para soportar en la implementación de sus módulos varios lenguajes de programación; pero el proyecto ha definido como estándar el uso del lenguaje C++.

2.2.2. Metodología de desarrollo de software.

Una metodología es una ciencia que analiza las vías de conocimiento, es la aplicación lógica de un método o un grupo de acciones efectuadas con el objetivo de alcanzar un resultado definido, es decir, representa la forma de integrar el proceso de la investigación, de registrar sus efectos y de demostrar viables resultados a un problema que conlleva la toma de decisiones.

SXP:

La metodología de desarrollo de software SXP está integrada por las metodologías SCRUM y XP ofreciendo una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento del desempeño productivo fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo.

SCRUM es una vía de gestionar en los proyectos de software un grupo para que trabaje con calidad y tenga siempre controlado los progresos. XP es una metodología encaminada para el desarrollo; donde se lleva a cabo una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto.

Fases de SXP:

Planificación-Definición: en esta fase se establece la visión, se declaran todas las expectativas y se crean los aseguramientos necesarios para el financiamiento del proyecto.

Desarrollo: es la fase donde se va a implementar todo el sistema hasta que se encuentre listo para ser entregado a su destinatario.

Entrega: se pone en marcha el sistema desarrollado.

Mantenimiento: es donde se realiza el soporte para el cliente.

En cada fase se realizan varias actividades como levantamiento de requisitos, definición de las Historias de Usuarios, diseño, implementación, pruebas entre otras que facilitan la generación de artefactos para la documentación de todo el proceso. Se hacen frecuentes entregas permitiendo mejorar el diseño cada vez que surja una nueva funcionalidad.

SXP es la metodología usada en el trabajo de diploma puesto que: *"esta 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"* **[8].**

2.2.3. Lenguaje unificado de modelado (UML).

Es uno de los lenguajes más reconocidos y utilizados en la actualidad para la modelación de software. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Brinda un estándar para representar un "plano" del sistema, agregando aspectos conceptuales tales como: procesos de negocios, funciones del sistema, expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. Se puede aplicar de varias formas para sobrellevar una metodología de desarrollo, aunque no define cual usar o que proceso emplear. UML tiene diversos tipos de diagramas, los cuales expresan disímiles aspectos de las entidades personificadas:

Diagramas de clases para personalizar la estructura estática de las clases en el sistema.

1. Diagramas de objetos para simbolizar la estructura estática de los objetos en el negocio.
2. Diagramas de casos de uso para representar los procesos del negocio.
3. Diagramas de actividad para modelar el comportamiento de los casos de uso, objetos u operaciones.
4. Diagramas de secuencia para escenificar el paso de mensajes entre

objetos.

5. Diagramas de colaboración para modelar interacciones entre objetos.
6. Diagramas de estado para simbolizar el comportamiento de los objetos en el sistema.
7. Diagramas de componentes para modelar componentes.
8. Diagramas de implementación para personificar la distribución del sistema **[9]**.

2.2.4. Herramientas CASE.

Constituyen un cúmulo de programas y ayudas, que suministran apoyo a los ingenieros de software y desarrolladores, durante el período de desarrollo. *"El objetivo primordial de las herramientas de esta categoría consiste en representar objetos de datos de negocios, sus relaciones, y ayuda a comprender mejor la forma en que fluyen estos objetos de datos entre distintas zonas de negocio en el seno de la compañía. Estas herramientas proporcionan una ayuda importante cuando se diseñan nuevas estrategias para los sistemas de información y cuando los métodos y sistemas no satisfacen las necesidades de la organización"* **[10]**.

Ventajas de las herramientas CASE:

- Aumento de la velocidad en la construcción de sistemas.
- Ayudan a los analistas a beneficiarse de más tiempo para el análisis y diseño, además de recortar el tiempo para codificar y probar.
- Automatiza el esbozo de diagramas.
- Auxilian en la documentación del sistema.
- Aprueba generar estructuras de código.

En la actualidad estas herramientas acompañadas por una metodología y algún lenguaje de modelado, son utilizadas como punto clave en el desarrollo de cualquier software, pues admiten la creación y modificación de diagramas con gran facilidad y sencillez, aumentando en gran escala la calidad de los diseños de software.

Visual Paradigm:

"Es una herramienta profesional, es decir, un software de modelado que utiliza UML como lenguaje de modelado. Soporta el ciclo completo de vida del software (análisis y diseño orientados a objetos, implementación, pruebas y despliegue); permitiendo en

cada una de sus etapas generar los diagramas necesarios sin ningún tipo de problema" **[11]**. Está orientado para posibilitar tanto ingeniería directa como inversa, pues posee varios lenguajes de programación que aprueban la generación de código.

Esta herramienta CASE soporta la importación y exportación de varias versiones de XML. Facilita la modelación de diversos tipos de diagramas, transformando códigos de estos modelos, concibiendo de esta manera, los códigos fuentes de los diagramas.

Algunas características de Visual Paradigm:

- Posee generación de código para Java y la exportación de todos los diagramas a formato html y jpg.
- Ostenta de un medio de creación de diagramas para UML 2.0.
- Posibilita la integración a los principales IDE.
- Cuenta con un diseño enmarcado en casos de uso y dirigido al negocio.
- Contiene facilidades para representar especificaciones de casos de uso del sistema.

Visual Paradigm se empleará como herramienta para el modelado de diagramas debido a que es multiplataforma. Posee además una distribución automática de diagramas, ya que cuenta con una reorganización de las figuras y conectores de los diagramas UML. Además facilita la conversión de diagramas de colaboración a secuencia y viceversa.

2.3 IDES

Para la implementación del módulo a desarrollar, se hace uso de herramientas de desarrollo en dependencia de las necesidades, o de los recursos con los que se cuente en cada momento. Como IDE de programación y teniendo en cuenta lo antes mencionado, se usara Code:Block y Netbeans 6.9, así como el potente y versátil editor de texto Geany. Code:Block es usado por su flexibilidad, basado en su buen auto completado de código y el uso de abreviaturas, además como no se realizara ninguna aplicación visual pues la solución es completamente en consola donde gana mucho en velocidad por ser más ligero y a la vez potente. El Netbeans es usado principalmente por su gran completamiento de código, además la corrección de errores sin necesidad de compilar el programa, ganando así en tiempo, además de su potente debugguer. Por otra parte el editor de texto Geany es usado por su rapidez y

versatilidad, fundamentalmente cuando sea imposible contar con ninguno de los antes mencionados por razones externas al proyecto.

2.4 Propuesta del módulo.

Proceso de configuración en el arranque de los clientes ligeros en OSplugger

Actualmente el proceso de arranque por la red en OSplugger es muy complejo por el hecho de que todo el proceso de configuración del arranque hay que hacerlo de forma manual. Es por ello que para realizar esta tarea se necesita personal capacitado y con más que conocimientos básicos sobre GNU/Linux.

2.4.1. Módulo de arranque para OSplugger.

Es este trabajo de diploma se propone desarrollar un módulo que permita un arranque de forma automatizado y dinámico de los clientes ligeros denominado BootManager. Es un módulo que tiene interfaz gráfica, trabaja en los servidores mostrando un código estable. El mismo es capaz de preguntarle al usuario que sistema operativo (SO) desea para iniciar, añadir o eliminar de su cliente, en caso que el usuario no pueda conectarse al servidor le da la información necesario del por qué no hay conexión. Este módulo logra el arranque por la red de varias arquitecturas de hardware así como de protocolos de conexión, comportándose como un sistema robusto y fiable en el proceso de arranque por la red.

2.4.2. Arquitectura del módulo de arranque para OSplugger.

La arquitectura que se definió para la implementación del módulo de arranque fue una arquitectura por capas siendo su objetivo primordial la separación de sus tres niveles, permitiendo distribuir el trabajo de la aplicación que se esté desarrollando. En esta arquitectura se utilizará el patrón de diseño observador puesto que el mismo funcionará de la siguiente manera: el cliente ligero que este siendo atendido por OSplugger será observado por cada uno de sus módulos logrando que cada módulo actualice la información referente al cliente cada vez que se realice un cambio en su configuración. BootManager como parte del sistema

OSplugger el cual está compuesto por módulos se integra a su programación modular manteniendo la estructura basada en módulo.

En la capa presentación es donde se desarrolla lo que el usuario observa, o sea, se construye una interfaz con Syslinux ya que este permite construir un menú de arranque en corto período de tiempo convirtiéndose en el proyecto que más facilidades brinda en el desarrollo de un menú de arranque. Ahora en la capa lógica del negocio aparecen las clases implementadas para la creación de objetos y la manipulación de los mismos, conformando el núcleo de la aplicación. Estas clases se encargarán de configurar todo el proceso de arranque mediante la red, pues tendrán en cuenta la arquitectura que presenta el cliente que quiere iniciar y el protocolo de red por el cual desea ser atendido, llevando a cabo el almacenamiento de los datos necesarios para llevar un control estricto sobre los clientes que estén en ejecución; ejemplo MAC, arquitectura de la pc, protocolo de red.

La capa de datos es la que porta los ficheros y archivos que serán configurados para el buen funcionamiento del módulo de arranque, quedando demostrado que en la arquitectura tres capas cada capa proporciona servicios a la capa inmediatamente superior, y se sirve de las prestaciones que le brinda la inmediata inferior.

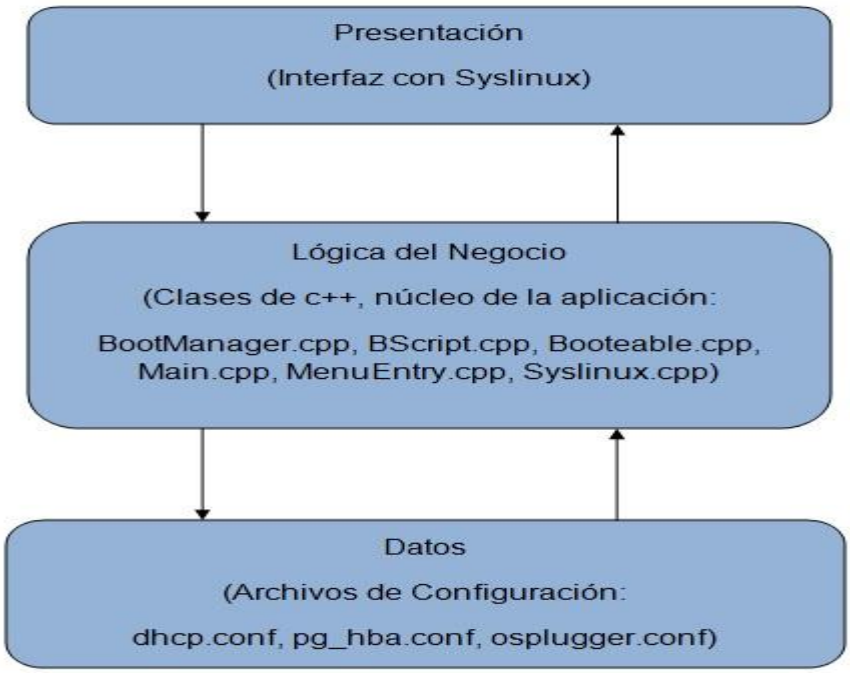


Fig # 1: Arquitectura tres capas del módulo de arranque.

2.5 Fase Planificación-Definición del Módulo de Arranque para OSplugger.

La fase Planificación-Definición es la primera de las cuatro fases con que cuenta la metodología SXP. En ella es donde se generan todos los documentos que se relacionan con la concepción inicial del sistema. Se incluyen otros documentos que se encuentran en la primera etapa de la Ingeniería de Software, estos son los relacionados con el Negocio, Requisitos del Sistema y Diseño.

A continuación se pasa a explicar todo el diseño del Módulo de Arranque en forma de historias de usuarios, prototipos de interfaz de usuario y algunos modelos auxiliares.

2.5.1. Plantilla de Concepción del Sistema

Es un documento que se describe después de sostener una entrevista con el cliente. En el mismo aparece una visión general del producto a implementar así como los roles que estarán presentes en su desarrollo y sus respectivas responsabilidades en este proceso.

Se podrá encontrar además a que proyecto pertenece, la especificación del polo productivo y su clasificación. En esta plantilla se recogen las herramientas que se emplearán para desarrollar el módulo, el alcance que tendrá, una descripción de los involucrados en el negocio, los motivos por los cuales se lleva a cabo el desarrollo del módulo y la propuesta de solución.

Es una plantilla de gran importancia para la documentación del proyecto porque sienta la base para generar los demás artefactos durante todo el ciclo de desarrollo del software.

Alcance del Módulo de Arranque: Se quiere poner en práctica primeramente en la Universidad de las Ciencias Informáticas, específicamente en el proyecto OSplugger perteneciente a la Facultad 1 y al centro de desarrollo GEITEL. Siendo utilizado posteriormente en todas las instituciones u organismos que necesiten clientes ligeros.

Herramientas Utilizadas: Para realizar este módulo se utilizarán las herramientas como: Visual Paradigm para los diagramas de la metodología;

C++, como lenguaje de programación y Netbeans 6.9 como IDE de programación.

Solución Propuesta: Se desea desarrollar un módulo de arranque que sea capaz de configurar el inicio de los clientes ligeros de forma automatizada y dinámica, y que su ejecución sea en diversos ámbitos. (Ver anexo # 1 - Plantilla de Concepción del Sistema).

2.5.2. Captura de Requisitos

En la etapa de captura de requisitos la plantilla de Lista de Reserva del Producto es el primer artefacto generado, estando conformada por una lista priorizada que define el trabajo que se va a realizar en el proyecto. Dicha plantilla permite la organización de los requisitos funcionales como no funcionales, en dependencia de la prioridad que tenga para el desarrollo del sistema.

Requisitos Funcionales				
Asignado a	Ítem	Descripción	Estimación	Estimado por
Prioridad		Muy Alta		
Sergio	1	Gestionar los SO disponibles del menú de selección para el cliente.	4.3 semanas	Sergio y Leiser
Prioridad		Alta		
Leiser y Sergio	2	Cambiar el estado del cliente.	4 semanas	Sergio y Leiser
Prioridad		Alta		
Leiser y Sergio	3	Borrar cliente.	0.6 semanas	Sergio y Leiser
Prioridad		Muy Alta		
Leiser	4	El sistema debe permitir al administrador de Osplugger	2.1 semanas	Sergio y Leiser

		seleccionar el protocolo de arranque.		
Prioridad		Muy Alta		
Sergio	5	El sistema debe permitir al administrador de Osplugger seleccionar la arquitectura de hardware por la cual se va a registrar los clientes.	0.3 semanas	Sergio y Leiser

Tabla # 3: Requisitos Funcionales con prioridad Muy Alta, alta.

Requisitos No funcionales				
Asignado a	Ítem	Descripción	Estimación	Estimado por
Prioridad		Media		
Sergio	1	En caso de que cese el fluido eléctrico, la carpeta que guarda la configuración de arranque debe.	1 semana	Sergio y Leiser
Prioridad		Alta		
Leiser	2	Al iniciar un cliente ligero sino está registrado, registrarlo automáticamente	1 semana	Sergio y Leiser

		en el sistema Osplugger.		
Prioridad		Media		
	3	Se necesita un servidor: • Apache • FTP • TFTP • DHCP		
Prioridad		Media		
	4	El servidor donde se alojará la aplicación debe tener: • Memoria: 1GB de RAM o superior. • Procesador: Pentium 4 o superior		
Prioridad		Alta		
Leiser	5	Lograr minimizar el tiempo de registro de un cliente en el servidor	1 semana	Sergio y Leiser

Tabla # 4: Requisitos No Funcionales con prioridad alta y media.

2.6 Definir las Historias de Usuarios.

En ellas el cliente describe brevemente las características que el sistema debe poseer, ya sean los requisitos funcionales o no funcionales. Trabajar con historias de usuarios es muy dinámico y flexible, dependiendo principalmente de la habilidad que tenga el analista y los desarrolladores para definirlas.

2.6.1. Plantilla de Historias de Usuarios.

Es una plantilla fácil de comprender puesto que está escrita en lenguaje del cliente, cada uno de los requisitos del sistema son especificados en ella sin necesidad de documentación extensa. La misma es la guía para el proceso de implementación del sistema reflejando todas sus características.

Historia de Usuario	
Número: HU-01	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Sergio y Leiser	**Iteración Asignada:** 1
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 4.3 semanas
Riesgo en Desarrollo: Alta	**Puntos Reales:** 4.3 semanas
Descripción: *Permite gestionar el trabajo con los sistemas operativos disponibles para el cliente en el menú de selección.*	
Observaciones:	
Prototipo de interfase:	

Tabla # 5: Historia de Usuario HU1.

Historia de Usuario	
Número: HU-02	**Nombre Historia de Usuario:** Cambiar el estado del cliente.
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Sergio y Leiser	**Iteración Asignada:** 2
Prioridad en Negocio: Alta	**Puntos Estimados:** 4 semanas
Riesgo en Desarrollo: Media	**Puntos Reales:** 4 semanas
Descripción: *Permite cambiar el estado de conexión del cliente.*	
Observaciones:	
Prototipo de interfase:	

Tabla #6: Historia de Usuario HU2.

Historia de Usuario	
Número: HU-03	**Nombre Historia de Usuario:** Borrar cliente.
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Sergio y Leiser	**Iteración Asignada:** 3
Prioridad en Negocio: Alta	**Puntos Estimados:** 0.6 semanas
Riesgo en Desarrollo: Media	**Puntos Reales:** 0.6 semanas
Descripción: *Eliminar un todos los datos de arranque de un cliente en el servidor.*	
Observaciones:	
Prototipo de interfase:	

Tabla #7: Historia de Usuario HU3.

Historia de Usuario	
Número: *H*U-04	**Nombre Historia de Usuario:** El sistema debe permitir al administrador de Osplugger seleccionar el protocolo de arranque.
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Sergio y Leiser	**Iteración Asignada:** 4
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 2.1 semanas
Riesgo en Desarrollo: Alta	**Puntos Reales:** 2.1 semanas
Descripción: *Seleccionar el protocolo mediante el cual el cliente realizará el arranque por la red.*	
Observaciones:	
Prototipo de interfase:	

Tabla #8: Historia de Usuario HU4.

Historia de Usuario	
Número: *H*U-05	**Nombre Historia de Usuario:** El sistema debe permitir al administrador de Osplugger seleccionar la arquitectura de hardware por la cual se va a registrar los clientes.
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Sergio y Leiser	**Iteración Asignada:** 5
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 0.3 semana
Riesgo en Desarrollo: Alta	**Puntos Reales:** 0.3 semana

Descripción: *Seleccionar la arquitectura de los clientes que están registrándose.*
Observaciones:
Prototipo de interfase:

Tabla #9: Historia de Usuario HU5.

2.6.2. Planificación de las Historias de Usuarios.

En esta etapa se planifica severamente el proyecto según su prioridad, riesgo y esfuerzos. Esta planificación es investigando experiencias pasadas en soluciones de este tipo. (Ver Anexo # 2-Planificación de Historias de Usuarios).

2.6.3. Lista de Riesgos.

En todo momento en el ciclo de desarrollo de un software existen un conjunto de riesgos que de plasmarse podrían poner en peligro el sistema, es por ello que para neutralizarlo es de suma importancia la gestión de los mismos.

La metodología SXP plantea la creación de la Plantilla Lista de Riesgos, en la que queda reflejado los posibles riesgos que procederán sobre el proceso de desarrollo del software, así como las estrategias trazadas para mitigarlos. Posee una gran importancia, pues a pesar de que es imposible definir desde un inicio todos los riesgos por los que pueda atravesar un proyecto, si se tendrán en cuenta la mayoría de estos. Esta plantilla propicia algunas ventajas, tales como: **[12]**

- Se definen los posibles riesgos, así como la forma de mitigarlos, lo que disminuye el efecto de los mismos, si ocurren.

- Se lleva un control de todos los problemas que han azotado al proyecto, así como de la manera que fueron enfrentados y el impacto que tuvieron en el proceso de desarrollo.

Los posibles riesgos durante el proceso de desarrollo del Módulo de Arranque para OSplugger se pueden encontrar en la Plantilla Lista de Riesgo (Ver Anexo # 3-Plantilla Lista de Riesgo).

2.7 Diseño con las Metáforas.

¿Qué es una metáfora?

Es una vía por donde todo el mundo puede contar acerca de cómo funciona todo el sistema. Su objetivo es proporcionarle al equipo una misma visión del fin del sistema y de su arquitectura, facilitando con esto que todos los desarrolladores hablen un mismo idioma y que los nuevos lo adquieran rápido y puedan integrarse en el proyecto sin ninguna dificultad.

2.7.1. Metáfora del módulo de arranque para OSplugger.

El módulo de arranque para OSplugger es una herramienta que permite que el proceso de arranque y configuración de los clientes ligeros se realice de una manera automatizada y dinámica para lograr un correcto funcionamiento dentro de OSplugger.

Plantilla de Modelo de Diseño.

Es una plantilla que permite confeccionar un diseño inicial y sencillo del sistema siendo la base para la definición de una futura arquitectura. En ella se realiza un diagrama de componentes el cual se encarga de describir los elementos físicos del sistema y sus relaciones. Los componentes representan todos los tipos de elementos de software que entran en la construcción de aplicaciones informáticas. Pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, etc.

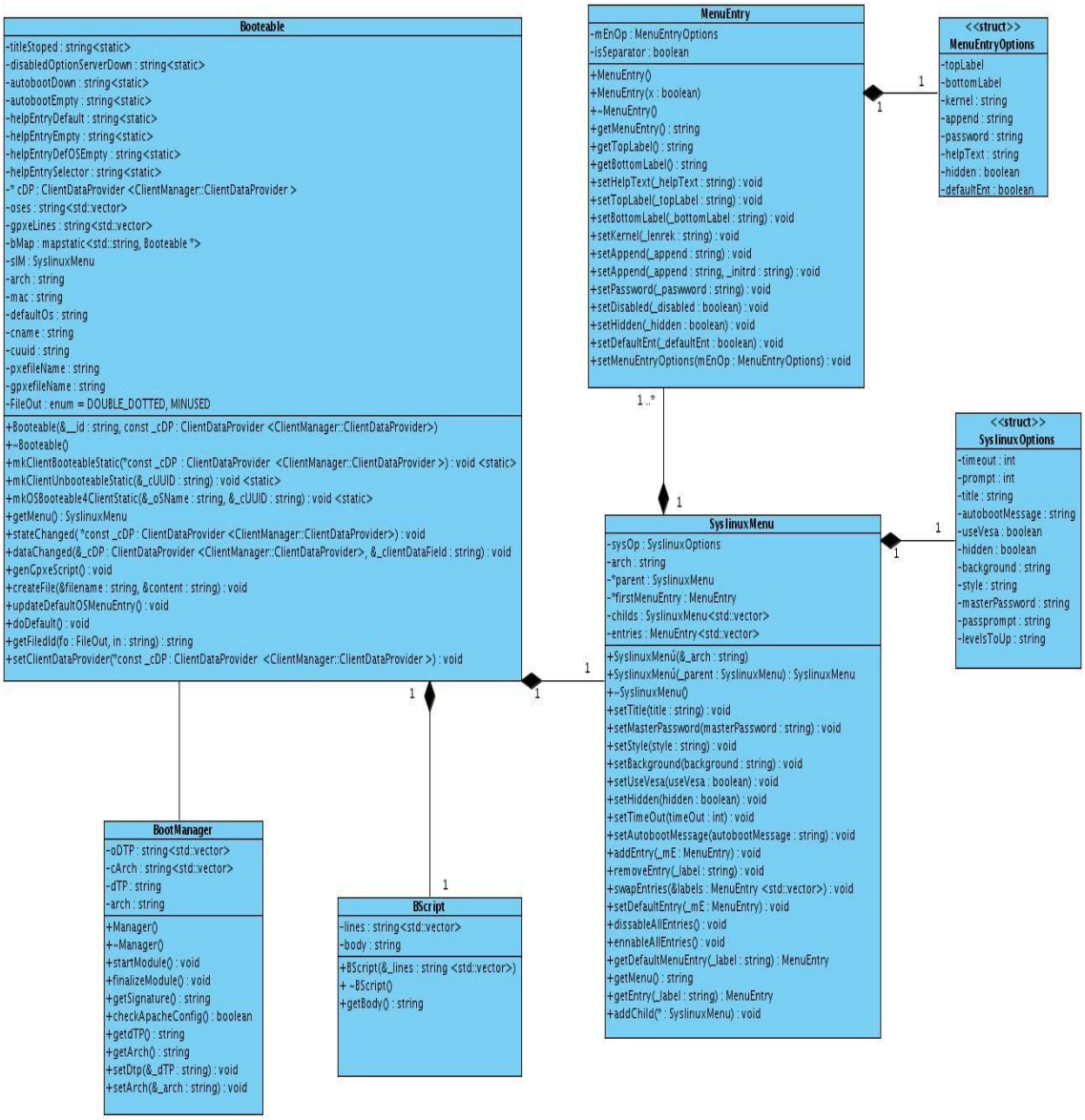


Fig # 2-Diagrama de Clases.

A todo lo largo de este capítulo se realizó la planificación del módulo de arranque para OSplugger llegando a un acuerdo entre el cliente y los desarrolladores, se definieron las historias de usuarios, se hizo un levantamiento de requisitos con el fin de determinar las necesidades del cliente. Se confeccionó el diseño de metáforas, donde aparece el diagrama de componentes del módulo a desarrollar el cual permitió dar una mejor visión del resultado final que se quiere alcanzar.

Capítulo 3: Implementación y pruebas del módulo de arranque para OSplugger

En el presente capítulo se encontraran todas las tareas de la ingeniería a desarrollar por los programadores del módulo de arranque a implementar dándole cumplimiento a todas las historias de usuario definidas en el capítulo anterior. Se construye un *plan de releases* donde queda creado el tiempo en que se pretende liberar el producto completo, mencionándose los estándares de codificación utilizado por el equipo programador de la aplicación. Se documentan todas las pruebas de funcionalidad realizadas al modulo de arranque.

3.1 Fase "Desarrollo" del Módulo de Arranque

La metodología SXP cuenta con una segunda fase llamada "Desarrollo". En ella se generan todos los documentos relacionados con la proyección de las iteraciones, se obtienen las principales definiciones que se manejan en la dicha metodología y los términos que sean de difícil entendimiento para los clientes, siendo estas las tareas a desarrollar durante la implementación. Da paso al código fuente en la etapa de implementación y los documentos relacionados con las pruebas.

3.2 Tareas de Ingeniería del Módulo de Arranque

Las tareas de la ingeniería definen cada una de las actividades que están asociadas a las historias de usuario. En ella se da a conocer el programador asignado a cada tarea, así como el tiempo necesario para su realización facilitando el tiempo que comprenderá cada historia de usuario en implementarse.

Tareas de Ingeniería	
Número Tarea: TI1	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.
Nombre de la Tarea: Estudiar el trabajo con los comboot de Syslinux menú.c32 y vesamenu.c32.	
Tipo de Tarea: Estudio	**Puntos Estimados:** 0.4 semana

Fecha de Inicio: 1/03/2011	**Fecha Fin:** 4/03/2011
Programador Responsable: Leiser Pérez, Sergio Velázquez.	
Descripción: Se estudia el trabajo y funcionamiento con los comboot de Syslinux menú.c32 y vesamenu.c32 para la construcción y muestra del menú de selección.	
Tareas de Ingeniería	
Número Tarea:TI2	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.
Nombre de la Tarea: Estudio de la utilización de gPXE y PXElinux para el arranque por la red, y los parámetros necesarios para iniciar los SO.	
Tipo de Tarea: Estudio	**Puntos Estimados:** 1.6 semana
Fecha de Inicio: 6/03/2011	**Fecha Fin:** 19/03/2011
Programador Responsable: Leiser Pérez, Sergio Velázquez.	
Descripción: Se realizará el estudio de los cargadores de arranque por la red gPXE y PXElinux, y de los parámetros necesarios para iniciar los sistemas operativos.	

Tabla # 10: TI1 y TI2.

Tareas de Ingeniería	
Número Tarea:TI3	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita adicionar un SO al menú de selección del cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 20/03/2011	**Fecha Fin:** 23/03/2011
Programador Responsable: Leiser Pérez, Sergio Velázquez.	
Descripción: Implementar la funcionalidad que permita adicionar un SO al menú de selección del cliente.	

Tabla # 11: TI3.

Tareas de Ingeniería	
Número Tarea:TI4	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita eliminar un SO al menú de selección del cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 24/03/2011	**Fecha Fin:** 27/03/2011
Programador Responsable: Leiser Pérez, Sergio Velázquez.	
Descripción: Implementar la funcionalidad que permita eliminar un SO al menú de selección del cliente.	

Tabla # 12: TI4.

Tareas de Ingeniería	
Número Tarea:TI5	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita seleccionar el SO por defecto a iniciar.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 28/03/2011	**Fecha Fin:** 31/03/2011
Programador Responsable: Leiser Pérez, Sergio Velázquez.	
Descripción: Implementar la funcionalidad que permita escoger el SO por el cual el cliente desea iniciar automáticamente.	

Tabla # 13: TI5.

Tareas de Ingeniería	
Número Tarea:TI6	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita la construcción de el archivo de configuración del menú de arranque donde muestre los SO disponibles.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 1/04/2011	**Fecha Fin:** 4/04/2011
Programador Responsable: Leiser Pérez, Sergio Velázquez.	
Descripción: Se implementa la funcionalidad que permite la elaboración del archivo de configuración del menú de arranque que mostrará los SO del cliente.	

Tabla # 14: TI6.

Tareas de Ingeniería	
Número Tarea:TI7	**Nombre Historia de Usuario:** Cambiar el estado del cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita cambiar el estado de un cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 05/04/2011	**Fecha Fin:** 08/04/2011
Programador Responsable: Sergio Velázquez.	
Descripción: Se implementará una funcionalidad que permitirá cambiar el menú de selección cuando un cliente está en estado empty.	

Tabla # 15: TI7.

Tareas de Ingeniería	
Número Tarea:TI8	**Nombre Historia de Usuario:** Cambiar el estado del cliente.

Nombre de la Tarea: Implementar la funcionalidad que permita cambiar el estado de un cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 09/04/2011	**Fecha Fin:** 12/04/2011
Programador Responsable: Sergio Velázquez.	
Descripción: Se implementará una funcionalidad que permitirá cambiar el menú de selección cuando un cliente está en estado started.	

Tabla # 16: TI8.

Tareas de Ingeniería	
Número Tarea:TI9	**Nombre Historia de Usuario:** Cambiar el estado del cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita cambiar el estado de un cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 13/04/2011	**Fecha Fin:** 16/04/2011
Programador Responsable: Sergio Velázquez.	
Descripción: Se implementará una funcionalidad que permitirá cambiar el menú de selección cuando un cliente está en estado stopped.	

Tabla # 17: TI9.

Tareas de Ingeniería	
Número Tarea:TI10	**Nombre Historia de Usuario:** Cambiar el estado del cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita cambiar el estado de un cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 17/04/2011	**Fecha Fin:** 20/04/2011

Programador Responsable: Sergio Velázquez.
Descripción: Se implementará una funcionalidad que permitirá cambiar el menú de selección cuando un cliente está en estado general_problem.

Tabla # 18: TI10.

Tareas de Ingeniería	
Número Tarea:TI11	**Nombre Historia de Usuario:** Cambiar el estado del cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita cambiar el estado de un cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 21/04/2011	**Fecha Fin:** 24/04/2011
Programador Responsable: Sergio Velázquez.	
Descripción: Se implementará una funcionalidad que permitirá cambiar el menú de selección cuando un cliente está en estado pluggos_problem.	

Tabla # 19: TI11.

Tareas de Ingeniería	
Número Tarea:TI12	**Nombre Historia de Usuario:** Cambiar el estado del cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita cambiar el estado de un cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 25/04/2011	**Fecha Fin:** 28/04/2011
Programador Responsable: Sergio Velázquez.	
Descripción: Se implementará una funcionalidad que permitirá cambiar el menú de selección cuando un cliente está en estado leaving.	

Tabla # 20: TI12.

Tareas de Ingeniería	
Número Tarea:TI13	**Nombre Historia de Usuario:** Cambiar el estado del cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita cambiar el estado de un cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.4 semana
Fecha de Inicio: 29/04/2011	**Fecha Fin:** 02/05/2011
Programador Responsable: Sergio Velázquez.	
Descripción: Se implementará una funcionalidad que permitirá cambiar el menú de selección cuando un cliente está en estado removed.	

Tabla # 21: TI13.

Tareas de Ingeniería	
Número Tarea:TI14	**Nombre Historia de Usuario:** Borrar cliente.
Nombre de la Tarea: Implementar la funcionalidad que permita eliminar un cliente de la lista de clientes.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.6 semana
Fecha de Inicio: 06/04/2011	**Fecha Fin:** 11/04/2011
Programador Responsable: Leiser Pérez.	
Descripción: Se implementará una funcionalidad que permita eliminar los datos de arranque de un cliente en el servidor, cuando se decida borrar el mismo.	

Tabla # 22: TI14.

Tareas de Ingeniería	
Número Tarea:TI15	**Nombre Historia de Usuario:** El sistema debe permitir al administrador de Osplugger seleccionar el protocolo de arranque.

Nombre de la Tarea: Estudiar el funcionamiento de protocolos para el arranque por la red.	
Tipo de Tarea: Estudio	**Puntos Estimados:** 1.2 semana
Fecha de Inicio: 12/04/2011	**Fecha Fin:** 28/04/2011
Programador Responsable: Leiser Pérez	
Descripción: Se realizará un estudio de los protocolos de red soportados por el cargador de arranque gPXE.	

Tabla # 23: TI15.

Tareas de Ingeniería	
Número Tarea:TI16	**Nombre Historia de Usuario:** El sistema debe permitir al administrador de Osplugger seleccionar el protocolo de arranque.
Nombre de la Tarea: Implementar la funcionalidad de la selección del protocolo de red a utilizar en el arranque.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.6 semana
Fecha de Inicio: 29/04/2011	**Fecha Fin:** 4/05/2011
Programador Responsable: Leiser Pérez	
Descripción: La funcionalidad permitirá la selección del protocolo de red por el cual se iniciaran los clientes.	

Tabla # 24: TI16.

Tareas de Ingeniería	
Número Tarea:TI17	**Nombre Historia de Usuario:** El sistema debe permitir al administrador de Osplugger seleccionar la arquitectura de hardware por la cual se va a registrar los clientes.
Nombre de la Tarea: Implementar la funcionalidad para seleccionar la arquitectura con la que se va a registrar el cliente.	
Tipo de Tarea: Desarrollo	**Puntos Estimados:** 0.3 semana

Fecha de Inicio: 03/05/2011	**Fecha Fin:** 06/05/2011
Programador Responsable: Sergio Alejandro Velázquez Leyva	
Descripción: La funcionalidad permitirá seleccionar la arquitectura con la que se va a registrar el cliente.	

Tabla # 25: TI17.

3.3 Plan de Release o Plan de Liberación del Módulo de Arranque

En el plan de release, el cliente decide que historias de usuarios deben ser incluidas en un lanzamiento. Las historias de usuarios de mayor riesgo y mayor prioridad se incluyen en las primeras iteraciones. Plan de release es un artefacto que proporciona las siguientes ventajas:

- Determina las historias de usuarios más importantes y las ubica en las iteraciones según su prioridad.
- El proceso de desarrollo de software lo divide en iteraciones, ingeniando el trabajo a realizar en cada una de ellas.
- Define las entregas intermedias y final del producto a desarrollar.

A continuación se muestra el Plan de Release que guiará el desarrollo del software:

Release	Descripción	Orden de Implementación	Duración
1	En esta iteración se implementaran las Historias de Usuarios con prioridad Muy Alta	HU-1 HU-4 HU-5	6.4 semanas
2	En esta iteración se implementaran las Historias de Usuarios con prioridad Alta.	HU-2 HU-3	4.5 semanas

Tabla # 26: Plan de Release.

3.4 Implementación del Módulo de Arranque

Estándares de Codificación

En el desarrollo del módulo de arranque se utilizó el estándar de codificación ANSI, es el estándar de codificación del lenguaje de programación C++, utilizado en el proyecto OSplugger. Las representaciones proporcionadas en éste mejoran considerablemente la legibilidad y el entendimiento del código.

3.5 Fase "Pruebas" del Módulo de Arranque

Todo producto de software depende de un plan de pruebas para alcanzar su máxima calidad, estas pruebas serán llevadas a cabo desde el principio, dándole seguimiento a los cambios y que el proceso de desarrollo del producto sea de forma iterativa. A continuación se muestran los casos de pruebas o test de aceptación a los que fue sometido el módulo de arranque en cada una de sus iteraciones, cumpliendo cada uno de estos casos de pruebas fue que se pudo lograr la especificidad de la aplicación.

3.5.1 Al iniciar el módulo de arranque BootManager.

Cuando se hicieron las primeras pruebas para inicializar el módulo se encontró el problema que al usar el Syslinux 3.63 que es el traído por defecto la distribución GNU/Linux Ubuntu el mismo no cargaba el menú de arranque en el cliente, puesto que no soportaba los ficheros menu.c32 y el versamenu.c32 de la distribución Syslinux antes mencionada acarreando la siguiente dificultad "*Al menos SuSE, Mandriva y Ubuntu usan la versión Syslinux modificada con el parche "gfxboot". Esto es una modificación que no soporta Syslinux. Por favor evite esta versión si es posible.*" **[13]** Es por ello que se utiliza la versión más actualizada de Syslinux, la 4.03 solucionando el problema anterior y facilitando una mejoría a la hora de cargar el menú de arranque para los clientes ligeros. (Ver anexo # 4-Posibles problemas al iniciar el módulo con S.O Ubuntu en el servidos).

3.5.2 Casos de Prueba de Aceptación del Módulo de Arranque.

Caso de Prueba de Aceptación	
Código de caso de prueba: HU-01	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.

Nombre de la persona que realiza la prueba: Mijaíl Hurtado Fedorovich
Descripción de la prueba: Se agrega uno o varios SO a un cliente.
Condiciones de ejecución: El servidor OSplugger debe estar funcionando y tener clientes registrados para asignarle los sistemas operativos.
Entrada / Pasos ejecución: Desde el servidor de OSplugger se le asignan nuevos sistemas operativos a un cliente. Se inicia el cliente y se verifica que aparezcan correctamente el o los nuevos sistemas a su disposición.
Resultado esperado: Al ejecutar la prueba el sistema debe: • Agregar el nuevo SO a la lista de sistemas disponibles para el cliente. • Añadir el SO al menú de selección al iniciarse el cliente. • Iniciar correctamente el SO añadido.
Evaluación de la prueba: Satisfactoria

Tabla # 27: Prueba HU-01 Agregar uno o varios SO.

Caso de Prueba de Aceptación	
Código de caso de prueba: HU-01	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.
Nombre de la persona que realiza la prueba: Mijaíl Hurtado Fedorovich	
Descripción de la prueba: Se eliminara uno de los sistemas operativos que el cliente tiene disponibles y se verificara que se realice correctamente.	
Condiciones de ejecución: El servidor OSplugger debe estar funcionando y tener clientes disponibles para eliminar de su lista de sistemas disponibles los sistemas operativos.	
Entrada / Pasos ejecución:	

Desde el servidor de OSplugger se le elimina un sistema operativo a un cliente. Se inicia el cliente y se verifica que no aparezca en el menú de selección el sistema operativo que se le eliminó.
Resultado esperado: Al ejecutar la prueba el sistema debe: • Eliminar el SO de la lista de sistemas disponibles para el cliente. • No mostrar el SO eliminado en el menú de selección al iniciar el cliente.
Evaluación de la prueba: Satisfactoria

Tabla # 28: Prueba HU-01 Eliminar un SO.

Caso de Prueba de Aceptación	
Código de caso de prueba: HU-01	**Nombre Historia de Usuario:** Gestionar los SO disponibles del menú de selección para el cliente.
Nombre de la persona que realiza la prueba: Mijaíl Hurtado Fedorovich	
Descripción de la prueba: Se cambiará el SO por defecto del cliente.	
Condiciones de ejecución: El servidor OSplugger debe estar funcionando y tener clientes disponibles para modificar el SO por defecto del cliente seleccionado.	
Entrada / Pasos ejecución: Desde el servidor de OSplugger se modifica el SO por defecto del cliente. Se inicia el cliente y se verifica que aparezca como SO por defecto el sistema seleccionado.	
Resultado esperado: Al ejecutar la prueba el sistema debe: • Modificar el SO a arrancar por defecto en el cliente. • Iniciar el cliente y los SO se muestren en el menú de selección conforme a los cambios realizados.	
Evaluación de la prueba: Satisfactoria	

Tabla # 29: Prueba HU-01 Modificar orden de los SO.

Caso de Prueba de Aceptación	
Código de caso de prueba: HU-02	**Nombre Historia de Usuario:** Cambiar el estado del cliente.
Nombre de la persona que realiza la prueba: Mijaíl Hurtado Fedorovich	
Descripción de la prueba: Se le cambia el estado a un cliente (iniciado, detenido, iniciándose, etc).	
Condiciones de ejecución: Que se encuentre corriendo el servidor de OSplugger. Que existan clientes disponibles para cambiar su estado.	
Entrada / Pasos ejecución: Se le cambiará el estado a un cliente.	
Resultado esperado: Al ejecutar la prueba el sistema debe: • Iniciar el cliente y mostrar los cambios de su estado en el menú de selección.	
Evaluación de la prueba: Satisfactoria	

Tabla # 30: Prueba HU-02 Cambiar estado del cliente.

Caso de Prueba de Aceptación	
Código de caso de prueba: HU-03	**Nombre Historia de Usuario:** Borrar un cliente.
Nombre de la persona que realiza la prueba: Mijaíl Hurtado Fedorovich	
Descripción de la prueba: Se eliminara todos los archivos y directorios pertenecientes al arranque de un cliente determinado.	
Condiciones de ejecución: Que el cliente este registrado en el servidor.	
Entrada / Pasos ejecución: Se elimina el cliente desde el servidor OSplugger.	
Resultado esperado:	

Al ejecutar la prueba el sistema debe: • Eliminar todos los archivos y directorios del cliente correctamente.
Evaluación de la prueba: Satisfactoria

Tabla # 31: Prueba HU-03 Borrar un cliente.

Caso de Prueba de Aceptación	
Código de caso de prueba: HU-04	**Nombre Historia de Usuario:** El sistema debe permitir al administrador de Osplugger seleccionar el protocolo de arranque.
Nombre de la persona que realiza la prueba: Mijaíl Hurtado Fedorovich	
Descripción de la prueba: Se selecciona el protocolo de arranque en el servidor Osplugger.	
Condiciones de ejecución: El servidor de OSplugger debe estar iniciado. El servidor Apache, TFTP o FTP según el protocolo seleccionado debe estar iniciado.	
Entrada / Pasos ejecución: Se inicia el cliente y su conexión al servidor OSplugger. Se verifica que se inicie el SO seleccionado correctamente.	
Resultado esperado: Al ejecutar la prueba el sistema debe: • Iniciar el SO seleccionado correctamente, descargando esta información utilizando el protocolo seleccionado.	
Evaluación de la prueba: Satisfactoria	

Tabla # 32: Prueba HU-04 Seleccionar el protocolo de arranque.

Caso de Prueba de Aceptación	
Código de caso de prueba: HU-05	**Nombre Historia de Usuario:** El sistema debe permitir al administrador de Osplugger seleccionar la arquitectura de hardware por la cual se va

	a registrar los clientes.
Nombre de la persona que realiza la prueba: Mijaíl Hurtado Fedorovich	
Descripción de la prueba: Se selecciona la arquitectura de hardware con la que se va a registrar el cliente en el servidor Osplugger.	
Condiciones de ejecución: El servidor de OSplugger debe estar iniciado.	
Entrada / Pasos ejecución: El servidor Osplugger selecciona la arquitectura por la cual va a registrar los clientes.	
Resultado esperado: Al ejecutar la prueba el sistema debe: • Registrar el cliente ligero correctamente, utilizando la arquitectura de hardware seleccionada.	
Evaluación de la prueba: Satisfactoria	

Tabla # 33: Prueba HU-05 Seleccionar la arquitectura de hardware por la cual se va a registrar los clientes.

Caso de Prueba de Aceptación	
Código de caso de prueba: HU-00	**Nombre Historia de Usuario:** Mostrar menú de selección a los clientes.
Nombre de la persona que realiza la prueba: Mijaíl Hurtado Fedorovich	
Descripción de la prueba: Se inician los clientes para verificar que se muestre correctamente su menú de selección con los sistemas operativos disponibles.	
Condiciones de ejecución: El servidor de OSplugger debe estar iniciado.	
Entrada / Pasos ejecución: Se inician los clientes.	
Resultado esperado: Al ejecutar la prueba el sistema debe:	

• Mostrar correctamente el menú de selección y los SO disponibles al cliente.
Evaluación de la prueba: Satisfactoria

Tabla # 34: Prueba HU-00 Mostrar menú de selección de los clientes.

Caso de Prueba de Aceptación	
Código de caso de prueba: HU-00	**Nombre Historia de Usuario:** Seleccionar e iniciar correctamente los sistemas operativos del menú de selección.
Nombre de la persona que realiza la prueba: Mijaíl Hurtado Fedorovich	
Descripción de la prueba: Se inician los clientes y se selecciona un sistema operativo de su menú de sistemas disponibles para iniciar.	
Condiciones de ejecución: El servidor de OSplugger debe estar iniciado.	
Entrada / Pasos ejecución: Se inicia un cliente y se selecciona uno de los sistemas operativos disponibles en su menú de selección y se inicia el mismo.	
Resultado esperado: Al ejecutar la prueba el sistema debe: • Permitir seleccionar los SO disponibles para el cliente, y que se inicie correctamente el que sea seleccionado finalmente.	
Evaluación de la prueba: Satisfactoria	

Tabla # 35: Prueba HU-00 Seleccionar e iniciar correctamente los SO del menú de selección.

3.6 Resultados Obtenidos

Con el desarrollo de este módulo de arranque OSplugger cuenta con una herramienta que agiliza todo el proceso de configuración y arranque de los clientes ligeros. Los usuarios que hagan uso de este módulo podrán realizar el proceso de inicio de un cliente ligero sin la necesidad de conocer cada uno de los comandos que hacen posible esto.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DEL MÓDULO DE ARRANQUE PARA OSPLUGGER

Como fin del presente trabajo de diploma la propuesta de solución está disponible en su versión 1.0, que está siendo usada actualmente en la facultad 1 de la Universidad de las Ciencias Informática específicamente en el proyecto OSplugger. Entre las principales funcionalidades que presenta el producto en su primera versión se encuentran:

- Adicionar, modificar, eliminar un sistema operativo determinado por el cliente.
- Cambiar el estado del cliente.
- Borrar cliente.
- Seleccionar el protocolo de arranque.
- Seleccionar la arquitectura de hardware por donde se registra el cliente ligero.

Es de importancia destacar que la calidad alcanzada en el producto logró el arranque de varios clientes ligeros dando muestra de eficiencia y rapidez en cuanto a la configuración de los mismos.

Conclusiones

Durante el desarrollo de esta investigación se demostró la importancia y la necesidad de la implementación del módulo de arranque BootManager en su versión 1.0 para el sistema OSplugger. Luego de terminado este trabajo se arriba a las siguientes conclusiones:

- La arquitectura diseñada para la implementación del módulo BootManager, soporta diferentes arquitecturas de hardware y protocolos de red.
- El uso del estándar de codificación ANSI en el lenguaje de programación C++, permitió obtener un código robusto y de fácil entendimiento.
- Se validó la calidad del módulo desarrollado, lo que permitió concluir que las funcionalidades alcanzadas se encuentran en correspondencias con las necesidades del cliente.

Recomendaciones

Los autores de la presente investigación recomiendan:

- Realizar pruebas de carga al módulo implementado, con el objetivo de asegurar su correcto funcionamiento.
- Cambiar para mayor seguridad la transferencia de datos a protocolos seguros.

Referencias Bibliográficas

[1] RIVEROS G, Héctor, ROSAS, Lucía. *El método científico aplicado a las ciencias experimentales*. México: 2000.

[2] ACRONIS, Inc. *Boot loader for Linux and Windows (2000, XP, Vista) OSes creates an emergency boot disk even in case of a boot hard disk failure*. Actualizada: 2000. [Fecha de consulta: 26 de enero del 2011]. Disponible en: http://www.acronis.es/enterprise/products/diskdirectorsuite/multibooting.html.

[3] DUALBOOTPRO. 2001. *Dual Boot PRO - Microsoft Windows Boot Editor - Windows BCDEdit Configuration*. Actualizada: 2001. [Fecha de consulta: 26 de enero del 2011] Disponible en: http://www.dualbootpro.org/

[4] GARNER, Keith, RAYE Geoff. *LILO*. Actualizada: 1996. [Fecha de consulta: 26 de enero del 2011] Disponible en**:** http://www.acm.uiuc.edu/workshops/linux_install/lilo.html

[5] PETER Anvin, H. *SYSLINUX*. Actualizada: 2010. [Fecha de consulta: 26 de enero del 2011] Disponible en: http://syslinux.zytor.com/wiki/index.php/SYSLINUX.

[6] ESPACIO, Linux. *Tuquito versión 4.0*. Tuxinfo número 29(2010) [Fecha de consulta: 17 de marzo del 2011] Disponible en: http://www.espaciolinux.com/2010/08/tuxinfo-29/

[7] FRIKILAND.COM. *Aumentando nuestra seguridad en Internet*. Actualizado: 2010 [Fecha de consulta: 17 de marzo del 2011] Disponible en: http://www.frikiland.com/?tag=pxe

[8] PEÑALVER Romero, Gladis Marsi. *MA-GMPR-UR2 Metodología ágil para proyectos de software libre*. Universidad de las Ciencias Informáticas. La Habana, 2008 [Fecha de consulta: 17de marzo del 2011]

[9] PRESSMAN, Roger. *Un enfoque práctico*. Madrid: 2010. [Fecha de consulta: 17 de marzo del 2011] Disponible en: http://eva.uci.cu/mod/resource/view.php?id=21621

[10] TARINGA. 2008. *Principales herramientas CASE del mercado y su uso*. Actualizado: 2008 [Fecha de consulta: 17 de marzo del 2011] Disponible en: http://www.taringa.net/posts/info/1461505/Principales-herramientas-CASE-del-mercado-y-su-uso.html

[11] LTD, Visual Paradigm International. *Paradigma visual para UML (Plataforma Java) (Visual Paradigm for UML [Java Platform]). 2007* [Fecha de consulta: 17 de marzo del 2011]

[12] CÉSPEDES R, F, GARCÍA S, P. *Propuesta de un expediente, para los proyectos productivos del Polo de Software Libre*. Universidad de las Ciencias Informáticas. La Habana, 2008. [Fecha de consulta: 20 de marzo del 2011]

[13] PETER Anvin, H. *Download from Kernel.org*. Actualizado: 2010 [Fecha de consulta: 19 de mayo del 2011] Disponible en: http://syslinux.zytor.com/wiki/index.php/Download

Bibliografía

RIVEROS G, Héctor, ROSAS, Lucía. *El método científico aplicado a las ciencias experimentales*. México: 2000.

ACRONIS, Inc. *Boot loader for Linux and Windows (2000, XP, Vista) OSes creates an emergency boot disk even in case of a boot hard disk failure*. Actualizada: 2000. [Fecha de consulta: 26 de enero del 2011]. Disponible en: http://www.acronis.es/enterprise/products/diskdirectorsuite/multibooting.html.

DUALBOOTPRO. 2001. *Dual Boot PRO - Microsoft Windows Boot Editor - Windows BCDEdit Configuration*. Actualizada: 2001. [Fecha de consulta: 26 de enero del 2011] Disponible en: http://www.dualbootpro.org/

GARNER, Keith, RAYE Geoff. *LILO*. Actualizada: 1996. [Fecha de consulta: 26 de enero del 2011] Disponible en**:** http://www.acm.uiuc.edu/workshops/linux_install/lilo.html

PETER Anvin, H. *SYSLINUX*. Actualizada: 2010. [Fecha de consulta: 26 de enero del 2011] Disponible en: http://syslinux.zytor.com/wiki/index.php/SYSLINUX.

ESPACIO, Linux. *Tuquito versión 4.0*. Tuxinfo número 29(2010) [Fecha de consulta: 17 de marzo del 2011] Disponible en: http://www.espaciolinux.com/2010/08/tuxinfo-29/

LINUXFOCUS. 2003. *Desarrollos de Software*. Actualizado: 2003 [Fecha de consulta: 17 de marzo del 2011]Disponible en: http://es.tldp.org/LinuxFocus/pub/mirror/LinuxFocus/Castellano/

FRIKILAND.COM. *Aumentando nuestra seguridad en Internet*. Actualizado: 2010 [Fecha de consulta: 17 de marzo del 2011] Disponible en: http://www.frikiland.com/?tag=pxe

PEÑALVER Romero, Gladis Marsi. *MA-GMPR-UR2 Metodología ágil para proyectos de software libre*. Universidad de las Ciencias Informáticas. La Habana, 2008 [Fecha de consulta: 17de marzo del 2011]

PRESSMAN, Roger. *Un enfoque práctico*. Madrid: 2010. [Fecha de consulta: 17 de marzo del 2011] Disponible en: http://eva.uci.cu/mod/resource/view.php?id=21621

TARINGA. 2008*. Principales herramientas CASE del mercado y su uso*. Actualizado: 2008 [Fecha de consulta: 17 de marzo del 2011] Disponible en:

http://www.taringa.net/posts/info/1461505/Principales-herramientas-CASE-del-mercado-y-su-uso.html

LTD, Visual Paradigm International. *Paradigma visual para UML (Plataforma Java) (Visual Paradigm for UML [Java Platform]). 2007* [Fecha de consulta: 17 de marzo del 2011]

CÉSPEDES R, F, GARCÍA S, P. *Propuesta de un expediente, para los proyectos productivos del Polo de Software Libre*. Universidad de las Ciencias Informáticas. La Habana, 2008. [Fecha de consulta: 20 de marzo del 2011]

PETER Anvin, H. *Download from Kernel.org*. Actualizado: 2010 [Fecha de consulta: 19 de mayo del 2011] Disponible en: http://syslinux.zytor.com/wiki/index.php/Download

CHAPLEAU, Frederick. *weBlog on IT*. Actualizado: 21 de septiembre del 2010. [Fecha de consulta: 17 de marzo del 2011]. Disponible en: http://www.chapleau.info/cs/blogs/fchapleau/default.aspx

REBOOT. GPXE Boot WinPE 3.0 Over HTTP. Actualizado: 12 de marzo del 2010. [Fecha de consulta: 22 de febrero del 2011]. Disponible en: ***http://reboot.pro/9802/***

DALI. Etherboot Project. Actualizado: 14 de octubre del 2010. [Fecha de consulta: 18 de febrero del 2011]. Disponible en: http://www.etherboot.org/wiki/httpboot

MARIN E, Lucian. *scnr.net.* Actualizado: 13 de noviembre del 2010. [Fecha de consulta: 8 de marzo del 2011]. Disponible en: http://scnr.net/blog/index.php/archives/177

Peter Anvin, H. *ISOLINUX.* Actualizado: 5 de noviembre del 2010. [Fecha de consulta: 15 de noviembre del 2011]. Disponible en: http://syslinux.zytor.com/wiki/index.php/ISOLINUX

LIMA. *linuxcentro.net*. Actualizado: 22 de febrero del 2011. [Fecha de consulta: 5 de febrero del 2011]. Disponible en: http://www.linuxcentro.net/linux/staticpages/index.php?page=GestorArranqueGrub.

SCRIBD. *Entrevista a Paco Revilla*. Linvix, número 6, febrero del 2010. [Fecha de consulta: 21 de enero del 2011]. Disponible en: http://es.scribd.com/doc/29854117/linvix-6

SUAREZ , Sebastián. *Infodocs*. Actualizado: marzo del 2010. [Fecha de consulta: 21 de noviembre del 2010]. Disponible en: http://infodocs.net/infodocs_old/documentation/ubuntu/livecd-ubuntu.

PETER Anvin, H. *Menu.c32.* Actualizado: 30 de abril del 2010. [Fecha de consulta: 25 de noviembre del 2010]. Disponible en: http://syslinux.zytor.com/wiki/index.php/Menu.c32.

PETER Anvin, H.*PXELINUX*. Actualizado: 20 de abril del 2011. [Fecha de consulta: 25 de abril del 2011]. Disponible en: http://syslinux.zytor.com/wiki/index.php/PXELINUX

BIOMETRIC EU, SIGNTECH. *Publicación de aplicaciones y escritorios*. Actualizado: 2010. [Fecha de consulta: 8 de diciembre del 2010]. Disponible en: http://www.signtechbiometric.com/2x.htm

HERRERO, Héctor. *2X ThinClient Server.* Actualizado: 2008. [Fecha de consulta: 21 de enero del 2011]. Disponible en: http://www.bujarra.com/?p=544

LINVIX. *Clientes Ligeros con TCOS y Ubuntu*. Actualizado: 2009. [Fecha de consulta: 21 de enero del 2011]. Disponible en: http://linvix.wordpress.com/2009/01/06/clientes-ligeros-con-tcos-y-ubuntu/

PETER Anvin, H. *Comboot/chain.c32*. Actualizado: 2010. [Fecha de consulta: 26 de noviembre del 2010]. Disponible en: http://syslinux.zytor.com/wiki/index.php/Comboot/chain.c32

HAJNOCZI, Stefan. *Etherboot command-line*. Actualizado: 2009. [Fecha de consulta: 9 de noviembre del 2010]. Disponible en: http://etherboot.org/wiki/commandline

SYSTEM, Zator. *5.5.2 Directorios y ficheros en C/C++*. Actualizado: 1990. [Fecha de consulta: 17 de febrero del 2011]. Disponible en: http://www.zator.com/Cpp/E5_5_2.htm#TOP

HAJNOCZI, Stefan. *Etherboot-scriptig*. Actualizado: 2009. [Fecha de consulta: 9 de noviembre del 2010]. Disponible en: http://etherboot.org/wiki/scripting

ROMERO, Gladys. MA-GMPR-UR2 Metodología ágil para proyectos de software libre. PhD thesis, Facultad 10, Universidad de las Ciencias Informáticas, Ciudad de La Habana, Cuba, Junio 2008.

SYSTEM, Dual Boot Pro. Microsoft Windows Boot Editor - Windows BCDEdit Configuration.Actualizado:2010. [Fecha de consulta 26 de noviembre del 2010]. Disponible en: http://www.dualbootpro.org

Anexos

Anexo 1: Plantilla de Concepción del Sistema

Nombre del Proyecto: Nova.

Nombre del Producto: Módulo de Arranque: Asistente para el arranque y configuración automatizada de los clientes ligeros.

Versión: 1.0

Polo Productivo: Software Libre.

Clasificación del proyecto: Desarrollo de un módulo.

Tipo de Proyecto: Nacional.

Resumen: Este producto pretende facilitar el trabajo de arranque y configuración automatizada de los clientes ligeros con los sistemas operativos Gentoo, Ubuntu, Nova y Windows XP. El mismo podrá soportar los protocolos de red TFTP, DHCP, HTTP y diferentes arquitecturas de maquinas como por ejemplo la arquitectura X86.

Palabras claves: Nova.

Surgimiento: Se decide desarrollar un módulo de arranque como este a partir de que al poner en práctica el arranque de los clientes ligeros se pone de manifiesto la demora y falta de personal capacitado para su configuración. Por lo anteriormente planteado el módulo de arranque puede permitir que estas tareas se realicen de forma automatizada acarreando dinamismo a la hora de arrancar un sistema operativo por la red.

Qué es: Es un módulo de arranque que una vez integrado a Osplugger puede permitir una configuración automatizada y dinámica necesaria para que un cliente ligero arranque adecuadamente por la red.

Metodología a utilizar: Se utilizará la metodología ágil SXP siendo esta una compilación de las mejores prácticas de dos metodologías ágiles muy bien conocidas como son Scrum y XP. Esta metodología permitirá a los autores de esta investigación implementar en menor tiempo la aplicación con un equipo reducido de desarrolladores, en este caso solo dos personas.

Involucrados: Ingeniero Mijaíl Hurtado Fedorovich, experto en servidores de terminales ligeras. Sergio Alejandro Velázquez Leyva y Leiser Pérez Matos, estudiantes universitarios de la Universidad de las Ciencias Informáticas.

Roles:

Rol	Responsabilidad	Nombre
Gerente	Su función es tomar decisiones acerca de estándares y decisiones durante el proyecto. Juega un papel importante en la definición de objetivos y requerimientos.	Mijaíl Hurtado Fedorovich
Cliente	Posibilita la definición de Historias de Usuario y los casos de prueba de funcionalidad, para validar su implementación. Es el encargado de asignar las prioridades de las Historias de Usuario y en las iteraciones que se van a realizar.	Proyecto Nova
Programador	Crea las tareas de ingeniería e implementa el código del sistema. Selecciona el o los estándares de programación a utilizar.	Sergio Alejandro Velázquez Leyva Leiser Pérez Matos
Analista	Escribe las historias de usuario y la concepción del sistema. Crea el modelo de Historias de Usuario del Negocio y la Lista De Reserva del Producto. Trabaja en estrecha relación con el cliente.	Sergio Alejandro Velázquez Leyva

Diseñador	Encargado del diseño del sistema y los prototipos de interface, realiza el diseño de las metáforas y supervisa el proceso de construcción.	Leiser Pérez Matos
Encargado de las Pruebas	Ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente. Responsable de soporte de pruebas.	Sergio Alejandro Velázquez Leyva Leiser Pérez Matos

Misión: El módulo de arranque para OSplugger tiene como misión facilitar el trabajo de los administradores de servidores de terminales ligeras a la hora de configurar todo el proceso de arranque de los mismos, pues este módulo posibilitará que todo el arranque sea de forma automatizada y dinámica.

Visión: Los autores del producto propuesto tienen como aspiración que esta aplicación ayude en el desarrollo de la tecnología de clientes ligeros en distintas áreas de la economía y de la sociedad cubana, reportando beneficios para el pueblo.

Alcance: Se quiere poner en práctica primeramente en la Universidad de las Ciencias Informáticas, específicamente en el proyecto OSplugger perteneciente a la Facultad 1 y al centro de desarrollo GEITEL. Siento utilizado posteriormente en todas las instituciones u organismos que necesiten clientes ligeros.

Herramientas Utilizadas: Para realizar este módulo se utilizarán las herramientas como: Visual Paradigm para los diagramas de la metodología; C++, como lenguaje de programación y Netbeans 6.9 como IDE de programación.

Solución Propuesta: Se desea desarrollar un módulo de arranque que sea capáz de configurar el arranque de los clientes ligeros de forma automatizada y dinámica, y que su ejecución sea en diversos ámbitos.

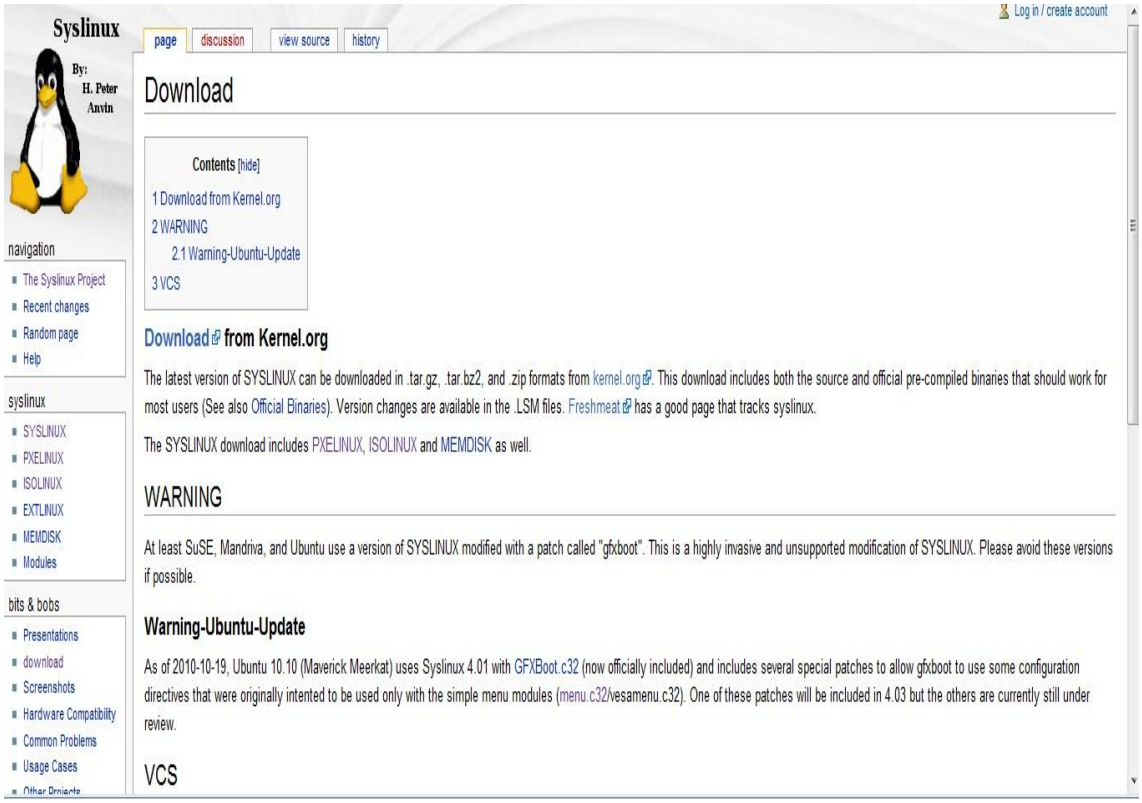
Anexo 2: Planificación de Historias de Usuarios.

No	Nombre HU	Prioridad	Riesgo	Esfuerzo (Semanas)	Iteración
1	Gestionar los SO disponibles del menú de selección para el cliente.	Muy Alta	Alto	4.3	1
2	Cambiar el estado del cliente.	Alta	Media	4	2
3	Borrar cliente.	Alta	Medio	0.5	3
4	El sistema debe permitir al administrador de Osplugger seleccionar el protocolo de arranque.	Muy Alta	Alta	2.1	4
5	El sistema debe permitir al administrador de Osplugger seleccionar la arquitectura de hardware por la cual se va a registrar los clientes.	Muy Alta	Alta	0.3	5

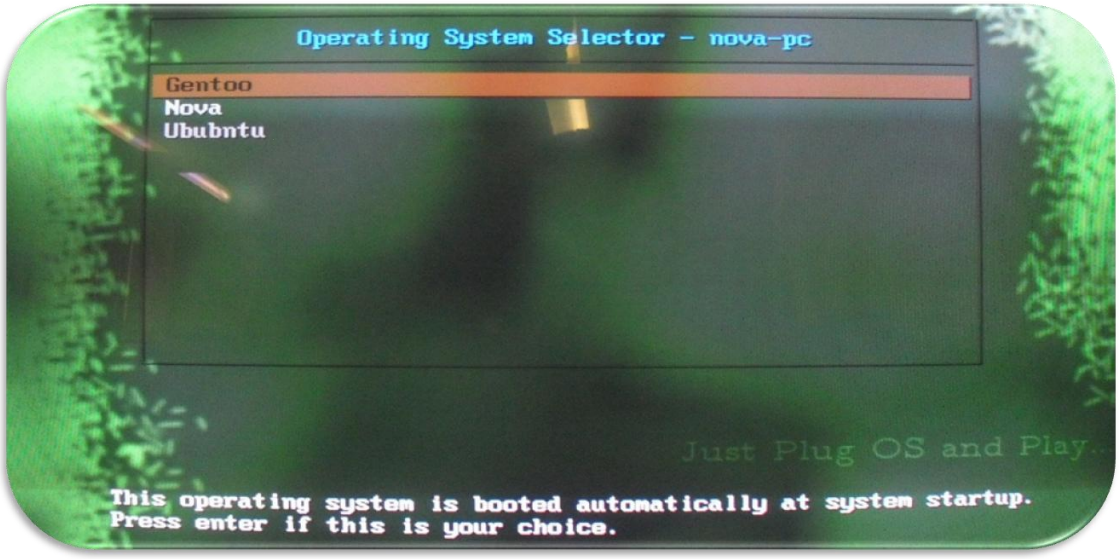
Anexo 3: Plantilla Lista de Riesgo

Riesgo	Tipos de Riesgos	Impacto	Descripción	Probabilidad	Efectos	Mitigación de Riesgo
1	Cálculo	Desarrollo	Complejidad en las tecnologías a usar.	Alta	Tolerables	Obtener conocimientos específicos del tema.
2	Tecnológico	Demora en el trabajo del sistema	Rotura de una pc donde se encuentre información importante.	Alta	Serios	Realizar salvas periódicas de la información.
3	Personal	Atraso en las actividades asignadas	Enfermedad de algún integrante del grupo de desarrollo.	Alta	Tolerable	Capacitar a varias personas como apoyo al grupo de desarrollo.
4	Tecnológico	Demora en el trabajo del sistema	Afectación del fluido eléctrico.	Baja	Tolerable	Tener horarios alternativos para el desarrollo del sistema.
5	Tecnológico	Desarrollo	Desconexión de la red a menudo.	Alta	Serio	Informar al jefe del proyecto.

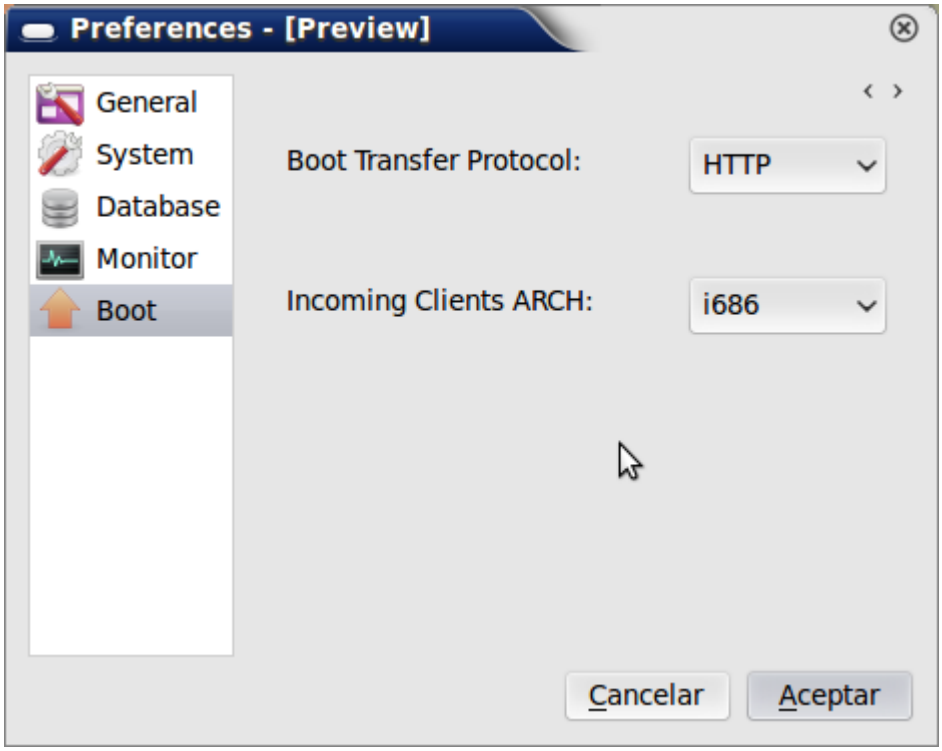
Anexo 4: Posibles problemas al iniciar el módulo con SO Ubuntu en el servidor.



Anexo 5: Módulo de arranque de estaciones ligeras para Osplugger.



Img # 1: Menú para seleccionar el SO por el cual va a iniciar el cliente ligero.



Img # 2: Interfaz gráfica en la cual se selecciona el protocolo de red y la arquitectura de hardware con la que se descargará el SO a iniciar y registrará el cliente ligero respectivamente.

Glosario de términos

GNU/Linux: es el término para referirse al sistema operativo similar a Unix que utiliza como base las herramientas del sistema GNU y el núcleo Linux. Es un software libre distribuido bajo la licencia GPL (Licencia Pública General de GNU) y otras. Las variantes de este sistema se denominan distribuciones y su objetivo es ofrecer un sistema ajustado a las necesidades de determinado grupo de usuarios.

PXE: siglas en inglés de Preboot eXecution Environment (Entorno de Ejecución de Prearranque). Es un entorno para arrancar e instalar el sistema operativo en computadoras desde una red. Está relacionado con varios protocolos de red como IP, DHCP, TFTP y UDP.

DHCP: siglas en inglés de Dynamic Host Configuration Protocol (Protocolo de Configuración Dinámica de Anfitrión). Es un protocolo que permite a los nodos de una red obtener los parámetro de red dinámicamente (Máscara de red, puerta de enlace broadcast y otros). Es un protocolo de tipo cliente servidor en el que generalmente el servidor posee una lista de direcciones IP y las va asignando a los clientes conforme que van estando disponibles, manteniendo siempre un registro de quien ha tenido esa IP, cuánto tiempo y a quién se le ha asignado después.

GPL: La Licencia Pública General de GNU es una licencia creada por la Free Software Foundation (Fundación del Software Libre) que está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

TFTP: siglas en inglés de Trivial File Transfer Protocol (Protocolo de Transferencia de Archivos Trivial). Es un protocolo de transferencia de archivos bastante sencillo, algo así como una versión básica de FTP. A menudo se utiliza para transferir pequeños archivos entre computadoras en una red, como cuando un terminal X Window o cualquier otro cliente ligero arrancan desde un servidor de red. Su característica distintiva es que no necesita de autenticación para establecer la transferencia y tampoco tiene mecanismos de cifrado.

X Window: el sistema de ventanas X (X Window System) fue desarrollado a mediados de los años '80 en el MIT (instituto Tecnológico de Massachusetts) para dotar de una interfaz gráfica a los sistemas Unix. Este protocolo permite la interacción gráfica en red entre un usuario y una o más computadoras haciendo transparente la red para éste.

Generalmente se refiere a la versión 11 de este protocolo, X11, el que está en uso actualmente.

RAM: de las siglas en inglés Random Acces Memory o Memoria de Acceso Aleatorio. Es la memoria desde donde el procesador recibe las instrucciones y guarda los resultados. Es el área de trabajo para la mayor parte del software de un computador.

HDD: de las siglas en inglés Hard Disc Drive o Unidad de Disco Duro. Es un dispositivo de almacenamiento no volátil que conserva la información aún con la perdida de energía empleando un sistema de grabación magnética digital.

Demonio: es un programa(s) informático(s) que se ejecuta(n) en segundo plano en Unix/Linux para brindar algún servicio como puede ser correo, servicio de impresión y otros. Estos programas se mantienen en ejecución mientras el sistema esté activo o hasta que el usuario no lo detenga. Un servicio puede estar soportado por más de un demonio.

ROM: es un medio de almacenamiento utilizado en ordenadores y dispositivos electrónicos, que permite sólo la lectura de la información y no su escritura, independientemente de la presencia o no de una fuente de energía.

BIOS (Sistema básico de entrada y salida): es un código de software que localiza y reconoce todos los dispositivos necesarios para cargar el sistema operativo en la memoria RAM; es un software muy básico instalado en la placa base que permite que ésta cumpla su cometido. Proporciona la comunicación de bajo nivel, el funcionamiento y configuración del hardware del sistema que, como mínimo, maneja el teclado y proporciona una salida básica.

MBR: es el primer sector ("sector cero") de un dispositivo de almacenamiento de datos, como un disco duro. Es usado para almacenar una tabla de particiones y, en ocasiones, se usa sólo para identificar un dispositivo de disco individual, aunque en algunas máquinas esto último no se usa y es ignorado.

UNIX: es un sistema operativo portable, multitarea y multiusuario.

NTFS: es un sistema de archivos de Windows NT incluido en las versiones de Windows 2000, Windows XP, Windows Server 2003, Windows Server 2008, Windows Vista y Windows 7. Es un sistema adecuado para las particiones de gran tamaño requeridas en estaciones de trabajo de alto rendimiento y servidores.