Universidad de las Ciencias Informáticas

Facultad 10

"Sistema de Clonación y Distribución de Imágenes de Sistemas Operativos"

Trabajo de Diploma para optar por el título de Ingeniero Informático

Autor:
Alejandro Valdés Villarrubia
Tutor:
Ing. Joe Fernández Vanega
Co-tutor:
Ing. Abel Meneses Abad
Consultor:
MSc. Ing. Fernando E. Valdés Pérez

Ciudad de la Habana

Curso 2006 – 2007

DECLARACIÓN DE AUTORÍA

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

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

Alejandro Valdés Villarrubia	Ing. Joe Fernández Vanega

Agradecimientos

A mis padres y familiares por el apoyo y la confianza brindada durante todos estos años.

A mis amigos por la ayuda prestada.

A mis profesores por la enseñanza.

Dedicatoria

A mi familia y amigos.

Resumen

Debido al alto número de laboratorios docentes con los que cuenta la Universidad del las Ciencias Informáticas (UCI) y al gran número de computadoras de diversos tipos que componen cada uno, se hace necesario una elevada cantidad de personal para poder mantener estos laboratorios en óptimas condiciones para su uso. Teniendo en cuenta esto y con el objetivo de reducir el tiempo y la cantidad de personal necesaria para realizar esta tarea, es que se desarrolla el presente trabajo.

En este trabajo se hace un estudio de los sistemas de clonación y distribución de software existentes en el mundo, así como se hace un análisis del funcionamiento y características fundamentales de cada uno de estos sistemas. Además se hace una propuesta de un sistema capaz de clonar, particionar e instalar imágenes de sistemas operativos en las computadoras de una subred de forma simultánea. Un sistema que para su funcionamiento utiliza la tecnología de Clientes Ligeros, repartiendo una meta-distribución del sistema operativo GNU/Linux a todas las computadoras de la subred y convirtiéndolas en clientes y tomando el control total del hardware de cada una de las computadoras.

Índice

Introducción

Las tecnologías de la información y las comunicaciones (TIC) se hacen imprescindibles para el desarrollo de todo el mundo. Cuba con el objetivo de alcanzar un mayor desarrollo, se ha introducido aun más en la esfera de la producción de software, con la creación de la más joven de las universidades cubanas, la Universidad de las Ciencias Informáticas (UCI). Esta universidad está estructurada en diferentes facultades, donde cada facultad se orienta a un perfil determinado, pero que tienen como principal objetivo la formación de especialistas en informática, teniendo como principio del proceso docente educativo "La formación desde la producción".

De acuerdo con este principio, la formación docente está vinculada a la producción, donde el estudiante adquiere las habilidades prácticas para el desarrollo y producción de software, el cual se desarrolla en los laboratorios docentes.

Estos laboratorios docentes se encuentran equipados con diferentes tipos de computadoras, que son atendidas por el grupo de técnicos de la universidad, quienes les dan el mantenimiento e instalan el software necesario de acuerdo con los requerimientos del proceso docente educativo. Teniendo en cuenta que para cada facultad las necesidades en cuanto a los software que utilizan son diferentes y que las imágenes a instalar en las computadoras tienen que ser personalizadas para cada uno de los tipos de computadoras y teniendo en cuanta el perfil de cada facultad, lo que provoca que se tengan que hacer varias imágenes personalizadas con las necesidades de cada una de las facultades.

El problema que motiva la realización de este trabajo, es que actualmente en la UCI el proceso de soporte de instalación de imágenes se realiza manualmente, lo cual acarrea muchas dificultades: Lentitud y complejidad del proceso de instalación. Dado que el proceso es realizado manualmente computadora a computadora se convierte en un proceso muy lento y tedioso debido al gran número de computadoras y de pasos necesarios para completar el proceso. Producto a esto surge la **situación problémica**, que consiste en que se tiene que pasar imagen a una gran cantidad de computadoras a la vez y no existe un

software que automatice este proceso. Por lo que el **problema científico** consiste en ¿Cómo diseñar un software que automatice el proceso de clonación y distribución de imágenes en la UCI?

Con este trabajo se pretende modelar un software que permita la instalación remota de imágenes de sistemas operativos en varias computadoras simultáneamente.

Por tanto el **objeto de investigación** de este trabajo son los sistemas de clonación y distribución de software existentes y su funcionamiento.

El **objetivo** de este trabajo es proponer el diseño de un software que automatice la clonación y distribución de imágenes para los laboratorios docentes de la Universidad de las Ciencias Informáticas.

Para cumplir con el objetivo propuesto se han definido las siguientes **tareas**:

- Investigar los diferentes sistemas de clonación y distribución software existentes en Cuba y el mundo.
- Comprender el funcionamiento de los sistemas de clonación y distribución de software.
- Analizar y diseñar un sistema cliente/servidor que permita clonar y distribuir imágenes de sistemas operativos de manera remota.

Este trabajo está estructurado en 3 capítulos y anexos, que incluye todo lo relacionado con el trabajo investigativo sobre los sistemas de clonación y distribución de software, así como el análisis y diseño del sistema a desarrollar. A continuación se muestra una breve descripción de cada uno de los capítulos.

Capítulo 1: Fundamentación teórica. En este capítulo se hace un análisis del tema a tratar y de las tecnologías actuales relacionadas con el tema. Además se hace una breve descripción de las técnicas y tecnologías utilizadas para el análisis, diseño e implementación del sistema.

Capítulo 2: Características del sistema. Se hace una descripción del objeto de estudio, así como el entorno de trabajo sobre el que funcionará el sistema. Se hará una descripción de una propuesta del sistema y de sus requerimientos funcionales y no funcionales así como la especificación de los casos de usos del sistema.

Capítulo 3: Análisis y diseño del sistema. En este capítulo se hace una descripción detallada del análisis y

del diseño del sistema desarrollado, y se muestran además los diferentes diagramas utilizados para el modelado del sistema.

Capítulo 1. Fundamentación Teórica

En el presente capítulo se pretende analizar los fundamentos de los sistemas de clonación y distribución de software existentes. También se mostrarán las diferentes técnicas y tecnologías existentes más utilizadas para el análisis, diseño e implementación de sistemas de clonación y distribución de software, así como los lenguajes a utilizar para el desarrollo del sistema.

1.1 Sistemas de clonación y distribución de software.

Uno de los problemas que afrontan los administradores de redes en la actualidad, es el del proceso de dar soporte e instalación de sistemas operativos a las diferentes computadoras de las redes de clientes. Algunos grupos de desarrolladores se han percatado de lo molesto y tedioso que se hace realizar un proceso como este en un número considerable de computadoras, por lo que se han dado a la tarea de realizar diferentes software que brinden rapidez y eficiencia a los administradores, a la hora de realizar el proceso de soporte e instalación de sistemas operativos en una red de computadoras.

Los sistemas de instalación remota se definen como los software que permiten que un administrador instale y configure nuevos equipos clientes de forma remota, sin necesidad de trabajar directamente en cada equipo cliente. Estos sistemas en la actualidad se han desarrollado en dos ramas fundamentales, la de los sistemas de instalación remota básica y la de los sistemas de clonación y distribución de software. Los sistemas de instalación remota básica se limitan solamente a la instalación de un sistema operativo base, en los equipos clientes, haciendo uso de la técnica de los llamados archivos de respuestas. Esta técnica consiste en crear un archivo donde se especifican las respuestas a las preguntas que hace el instalador por defecto de los sistemas operativos, estos archivos son creados por el administrador del sistema y una vez concluida su configuración son distribuidos en los diferentes equipos clientes con el objetivo de automatizar el proceso de instalación de los diferentes equipos.

A diferencia de los sistemas de instalación remota básica, los sistemas de clonación y distribución de software son capaces de realizar funciones como particionamiento de discos duros, clonación de sistemas operativos y distribución de imágenes de sistemas operativos. Para poder realizar estas funciones, hacen uso de diferentes tecnologías, así como de utilidades como son: (LTSP) Linux Terminal Server Project y Partimage.

1.1.1 Funcionamiento de los Sistemas de clonación y distribución de Software.

Los sistemas de Clonación y Distribución de Software se basan en un clásico modelo cliente/servidor, donde los clientes son los equipos que se van a instalar, y el servidor es la computadora que le proveerá a los clientes las imágenes que se les instalará. Por lo tanto en estos sistemas se tienen dos tipos de aplicaciones diferentes: una para el cliente y otra para el servidor. La aplicación servidor utiliza diferentes servicios (DHCP, TFTP, NFS, SSH) y tecnologías (Cluster) para poder comunicarse con los diferentes clientes y realizar el procesos de clonación y distribución.

Este proceso de clonación y distribución de software se divide en dos procesos fundamentales: El procesos de arranque y conexión del cliente con el servidor y el proceso de transferencia de ficheros y ejecución de comandos.

Procesos de arranque y conexión del cliente con el servidor.

En este proceso el servidor comienza una comunicación con el cliente, suministrándole una configuración de la interfaz TCP/IP de la tarjeta de red, para establecer la conexión por la que realizará la transferencia de ficheros. Este proceso inicia una vez que se enciende el terminal cliente.

● Cuando se enciende el terminal, este irá a través de su 'Power On Self Test' (POST) y el bootcode de la EPROM en la placa de red comenzará a ejecutarse.

● El bootcode intentará detectar la placa de red. Una vez que la detecte la inicializará.

● El bootcode hará un broadcast con un pedido dhcp en la red local. El pedido incluirá la dirección MAC de la placa de red.

● El servidor DHCP verá el broadcast y responderá el pedido leyendo su archivo de configuración y localizando la entrada que coincide con la dirección MAC que fue enviada. Entre la información que envía

el servidor podemos encontrar:

- dirección IP asignada al terminal
- máscara de red configurada para la red local
- directorio del cual extraer el kernel
- nombre del kernel a descargar

● El bootcode recibirá la respuesta del servidor DHCP y configurará la interfaz TCP/IP de la placa de red con los parámetros que fueron suministrados.

Una vez concluido este paso automáticamente comienza el proceso de transferencia de ficheros y ejecución de comandos.

Proceso de transferencia de ficheros y ejecución de comandos.

Al concluir el proceso de configuración de la interfaz TCP/IP de la tarjeta de red se comienza la configuración de los diferentes servicios de transferencia de ficheros.

● El bootcode, enviará un pedido TFTP al servidor para comenzar a descargar el kernel.

● Una vez que el kernel este completamente descargado en el terminal, el bootcode hará un salto hacia el código de inicio del mismo.

● El kernel comenzará a ejecutarse, inicializando todo el sistema y todos los periféricos.

● Luego hará otro pedido al servidor DHCP, el cual responderá enviando la información que necesita para continuar, entre ellas:

- ● Dirección IP asignada al terminal
- ● Máscara de red configurada para la red local
- ● El directorio raíz a ser montado por NFS
- ● El gateway

- El servidor DNS
- El hostname del terminal

● Una vez que estos parámetros lleguen al terminal, la nueva interfaz de red será configurada.

● El sistema de archivo raíz se montará vía NFS. Este se montará como solo lectura como medida de seguridad debido a que será montado simultáneamente en cada uno de los clientes a instalar.

Una vez que se monta el sistema de archivos en el cliente, el control lo toma el kernel pasando a hacer un reconocimiento del hardware del cliente. Después de terminado este reconocimiento el kernel pasa a ejecutar los diferentes script de configuración y ejecución de comandos determinado por el administrador del sistema para realizar la instalación, clonación o particionamiento del cliente.

1.2 Sistemas de clonación y distribución de software de uso actual.

A nivel mundial se han desarrollado diferentes sistemas para la instalación y administración de grupos de equipos en una red. Algunos de estos sistemas se han orientado específicamente a la clonación y distribución de sistemas. Debido a que son desarrollados para utilizarlos sobre sistemas operativos específicos, trae consigo que presenten diferentes características y que utilicen tecnologías y servicios diferentes para realizar sus funciones.

1.2.1 FAI (Instalación completamente automatizada).

Fully Automated Installation (FAI) es un sistema para instalar de forma automática y no iterativa una computadora o un grupo de computadoras. Está diseñado para trabajar sólo en una distribución de Debian(GNU/Linux) lo cual es un punto negativo debido a la variedad de sistemas operativos existentes. FAI puede automatizar el proceso de instalación completamente, tiene funcionalidades como:

- Proceso de instalación completamente automatizada y muy rápida.
- Los equipos a instalar se pueden arrancar desde disquete o por red. Se proporcionan los programas para crear discos de arranque.
- Soporta BOOTP y DHCP para la configuración de la red.

- No se necesita un disco RAM inicial, con 8Mb de memoria basta.
- Funciona en procesadores 386 y superiores (también se ha probado en SPARC).
- El kernel de instalación soporta módulos.
- Durante el proceso de instalación se puede entrar en los sistemas mediante ssh para realizar ajustes, o mediante dos terminales virtuales en consola.
- Se guarda una bitácora de todo el proceso en el servidor.
- Se pueden añadir scripts de configuración personalizados en shell script, perl, expect y cfengine.
- Se puede acceder al repositorio con los paquetes a instalar por NFS, FTP o HTTP.
- Se puede utilizar lilo o grub para configurar el arranque de disco duro.
- Puede ser utilizado como sistema de "rescate" ante fallos.

Características y funcionamiento.

La computadora a instalar se arranca, bien desde un disquete de arranque o por red mediante PXE, y obtiene su configuración de red (IP, servidores de DNS, gateway...) por DHCP, BOOTP o la lee del disco. Arranca un kernel de Linux y monta su sistema de ficheros raíz por NFS desde el servidor de instalación. Una vez que el sistema está listo, se comienzan a ejecutar una serie de scripts que obtienen del servidor de instalación todos los pasos a realizar: particionamiento del disco, software a instalar, configuración local. Finalmente, se reinicia el equipo para que arranque el sistema recién instalado.

La configuración sobre el particionamiento de discos, software a instalar, etc., se almacena en ficheros en el servidor. Estos ficheros de configuración pueden ser compartidos entre distintos equipos a instalar, e incluso se pueden crear clases con herencia, de forma que un mismo equipo puede tomar su configuración de varios perfiles distintos. De esta forma, se flexibiliza el proceso de configuración, con lo que se puede escalar la instalación de un número potencialmente muy grande de equipos. [SCDS]

FAI también puede ser utilizado como método de rescate ante fallos: es posible arrancar, montar el

sistema raíz desde NFS y hacer un login en ese sistema en lugar de continuar con la instalación. De esta forma se entra en el equipo sin utilizar los discos locales, que se comprueban por si tienen errores.

FAI es un sistema de instalación remota bastante potente y cómodo, a pesar de no contar con una interfaz gráfica. Sin embargo presenta el problema que está diseñado para funcionar únicamente con una distribución determinada de Linux, lo que limita al administrador tanto a la hora de elegir la distribución a instalar, como a la hora de instalar software que no tengan empaquetado en el formato propio de esa distribución.

1.2.2 LIU (Utilidad de GNU/Linux para la instalación del clusters).

LIU es una aplicación open-source desarrollada por el departamento de Linux de IBM. LIU al igual que FAI es una herramienta para la instalación remota sobre una red ethernet de equipos GNU/Linux. Esta aplicación presenta una interfaz grafica que brinda mayor facilidad al administrador en la selección de los recursos que LIU permite instalar y configurar. LIU permite instalar y configurar recursos como:

- Tabla de particiones del disco
- Kernel de Linux
- System.map del kernel
- Lista de RPMs a instalar
- Sistemas de ficheros (locales y remotos)
- Scripts de configuración/salida de usuarios
- Discos RAM
- Ficheros “fuente” (ficheros a copiar directamente del servidor al cliente)

Este sistema a pesar de tener un proceso de instalación relativamente fácil y de brindar diversas funcionalidades muy útiles para los administradores de grupos de computadoras, presenta dos importantes problemas:

- Sólo funciona en distribuciones RedHat y derivadas, ya que el software a instalar se elige en base a paquetes RPM. Por ello, hará difícil la tarea de mantener software que se hayan instalado a parte de la distribución.

- Sólo permite la instalación a través de redes ethernet. No se pueden realizar instalaciones o actualizaciones remotas a través de redes WAN TCP/IP.

1.2.3 VA SystemInstaller.

VA SystemInstaller es un Sistema para la instalación y actualización remota de sistemas basados en cualquier distribución de GNU/Linux, desarrollado por la empresa VA Linux, una empresa dedicada a la venta de hardware (servidores) equipado con sistemas Linux. VA SystemInstaller utiliza dos aplicaciones: System Imager y System Configurator. [SCDS]

System Imager: Proporciona las imágenes desde el servidor a los clientes, realiza actualizaciones, extrae imágenes desde los clientes hacia el servidor y dispone de los siguientes servicios:

- dhcpd (Servidor de DHCP y PXE).

- rsyncd (Servidor de rsync, para gestionar y extraer imágenes de los clientes).

- sshd (Servidor opcional de transporte).

System Configurator: Se inicia después de la instalación de (System Imager) y tiene las siguientes funciones:

- Detectar y configurar las interfaces de red.

- Configurar y ejecutar el gestor de arranque.

- Crear un ramdisk en caso de que sea necesario.

- Recordar y reportar todos los ficheros que se modifican.

- Configurar teclado y variantes.

- Configurar el horario.

Características principales.

- Todos los equipos se instalan en paralelo, incluso si se dispone de diferentes tipos de clientes con diferentes configuraciones de software.
- Los clientes a instalar se pueden arrancar (con el software cliente apropiado) desde el disco duro, disquetes, CDROM o incluso por red (si el hardware lo soporta).
- El software se encarga de particionar y formatear los discos duros, si fuera necesario, sin que sea necesario que los discos de cada equipo tengan el mismo tamaño.
- Las imágenes que se instalan son réplicas de un equipo que ha sido debidamente configurado.
- Las imágenes se basan en ficheros independientes, no en sistemas de paquetes (.deb o .rpm) exclusivos de alguna distribución.
- Se pueden mantener varias imágenes en el servidor, y “reinstalar” los equipos de una a otra al instante.
- A la hora de actualizar una de las instalaciones, sólo se envía por la red aquellas partes de la instalación que hayan sido modificadas haciendo uso de rsync, con la consiguiente reducción del tiempo de instalación.
- La instalación de un cliente se puede iniciar de forma remota por la red (con el software apropiado), haciendo posible la realización de instalaciones en zonas geográficamente dispersas sin necesidad de enviar técnicos a realizarla.

1.3 Herramientas, lenguajes y tecnologías a utilizar.

En el desarrollo de todo sistema informático es de vital importancia la selección de las herramientas a utilizar, paso que garantizará, de realizarse correctamente, un óptimo desempeño del sistema. Para el desarrollo del sistema al cual se refiere este documento, la selección se realizó teniendo en cuenta la infraestructura tecnológica de la UCI y valorando que el sistema a desarrollar, estará orientada a funcionar sobre el sistema operativo GNU/Linux.

1.3.1 ANJUTA.

Anjuta es un entorno integrado de desarrollo (IDE) desarrollado por Naba Kumbar en 1999. Fue desarrollado para programar en C y C++ en sistemas GNU/Linux aunque actualmente admite lenguajes como C#, Perl y Python. Su principal objetivo es trabajar con GTK y en el escritorio GNOME, además ofrece un gran número de características avanzadas de programación. Anjuta es un software libre, liberado bajo la licencia GPL.

Anjuta incluye:

- Administrador de proyectos.
- Asistentes.
- Plantillas.
- Depurador interactivo.
- Editor de texto; que verifica y resalta las sintaxis escritas.
- Extensión para Subversion (control de versiones).
- Interprete de comandos propio.

A pesar de incluir abundantes utilidades y tener una amplia variedad de funcionalidades es un sistema bastante estable y muy ligero en el consumo de recursos. [ANJ]

1.3.2 FDISK.

FDISK es una aplicación que suministra información sobre las particiones, con él es posible crear o eliminar particiones y unidades lógicas y definir la partición activa, si es que no lo está. Fdisk tiene versiones para diferentes sistemas operativos incluyendo GNU/Linux, para el cual se han desarrollado dos versiones; una con interfaz gráfica y otra para uso con comandos. La versión para uso por comandos, a pesar de no ser de tan fácil uso como la versión que tiene interfaz grafica, presenta la ventaja de que se puede automatizar el proceso, creando script y pasándole todos los parámetros necesarios para realizar el procesos de una forma simple y rápida. [FDK]

1.3.3 PARTIMAGE.

Partimage es una aplicación desarrollada para el trabajar sobre el sistema operativo GNU/LINUX. Esta aplicación fue desarrollada para guardar particiones de cualquier sistema de ficheros en un archivo de imagen. La imagen puede ser comprimida en formato GZIP/BZIP2, con tal de ocupar el mínimo espacio en el disco duro, también brinda la posibilidad de dividir este archivo imagen en múltiples ficheros para poder ser copiados en diferentes discos de almacenamiento. Partimage también tiene la funcionalidad de realizar el proceso contrario; teniendo un archivo imagen, instalarlo en una partición vacía.

Partimage soporta los siguientes sistemas de ficheros:

- ReiserFS: Un sistema de archivos utilizado por Linux y Unix.
- ext2fs/ext3fs: El sistema de archivos estándar de Linux.
- FAT16/32: El sistema de archivos de DOS & Windows 9x.
- NTFS: El sistema de archivos de Windows NT, 2000 y XP.
- HPSF: El sistema de archivos de IMB OS/2.
- JFS: Creado por IBM y usado en Aix (soporte beta).

1.3.4 LTSP.

Linux Terminal Server Project o **LTSP** son un conjunto de aplicaciones servidores que proporcionan la capacidad de ejecutar Linux en computadoras remotas. El sistema de funcionamiento del LTSP consiste en repartir por medio de la red el núcleo de Linux que es ejecutado por los clientes y que posteriormente ejecutaran secuencias de scripts típicos de una mini distribución. Los clientes podrán acceder a las aplicaciones por medio de una consola textual o por un servidor gráfico que se comparte utilizando el protocolo XDMCP. [LTSP00]

1.3.5 NCURSES.

Ncurses es una librería de programación que provee una API que permite al programador escribir interfaces basadas en texto, posibilitando la construcción de un Interfaz para el usuario, para aplicaciones

ejecutadas en un terminal. [NCS00]

1.3.6 BASH.

Bash es un Intérprete de comandos de tipo Unix (shell) escrito para el proyecto GNU. Su nombre es un acrónimo de Bourne-Again Shell. Es el shell por defecto en la mayoría de las distribuciones de GNU/Linux de hoy en día. Se encarga de interpretar las órdenes que le demos para su proceso por el kernel. Es el encargado de leer los caracteres tecleados por los usuarios, los interpreta y los ejecuta. Al escribir un comando, es el Shell y no el S.O., quien se encarga de interpretarlo y ordenar que se ejecute. [BSH00]

1.3.7 Lenguaje C.

C es un lenguaje de programación creado en 1969 por Ken Thompson y Dennis M. Ritchie en los Laboratorios Bell como evolución del anterior lenguaje B, a su vez basado en BCPL. Al igual que B, es un lenguaje orientado a la implementación de Sistemas Operativos, concretamente Unix. C es apreciado por la eficiencia del código que produce y es el lenguaje de programación más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.

Se trata de un lenguaje débilmente tipado de medio nivel pero con muchas características de bajo nivel. Dispone de las estructuras típicas de los lenguajes de alto nivel pero, a su vez, dispone de construcciones del lenguaje que permiten un control a muy bajo nivel. Los compiladores suelen ofrecer extensiones al lenguaje que posibilitan mezclar código en ensamblador con código C o acceder directamente a memoria o dispositivos periféricos.

C es un lenguaje muy eficiente puesto que es posible utilizar sus características de bajo nivel para realizar implementaciones óptimas. A pesar de su bajo nivel es el lenguaje más portado en existencia, habiendo compiladores para casi todos los sistemas conocidos. Proporciona facilidades para realizar programas modulares y/o utilizar código o bibliotecas existentes. [C00]

Entre las características de C las más importantes son:

- Un núcleo del lenguaje simple, con funcionalidades añadidas importantes, como funciones matemáticas y de manejo de ficheros, proporcionadas por bibliotecas.
- Es un lenguaje muy flexible que permite programar con múltiples estilos. Uno de los más

empleados es el estructurado no llevado al extremo (permitiendo ciertas licencias rupturistas).

- Un sistema de tipos que impide operaciones sin sentido.
- Usa un lenguaje de preprocesado, el preprocesador de C, para tareas como definir macros e incluir múltiples ficheros de código fuente.
- Acceso a memoria de bajo nivel mediante el uso de punteros.
- Un conjunto reducido de palabras clave.
- Los parámetros se pasan por valor. El paso por referencia se puede simular pasando explícitamente el valor de los punteros.
- Punteros a funciones y variables estáticas, que permiten una forma rudimentaria de encapsulado y polimorfismo.
- Tipos de datos agregados (struct) que permiten que datos relacionados se combinen y se manipulen como un todo.

1.3.8 Lenguaje C++.

El **C++** es un lenguaje de programación, diseñado a mediados de los años 1980, por Bjarne Stroustrup, como extensión del lenguaje de programación C. Las principales características del C++ son el soporte para programación orientada a objetos y el soporte de plantillas o programación genérica (*templates*). Se puede decir que C++ es un lenguaje que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos. Además posee una serie de propiedades difíciles de encontrar en otros lenguajes de alto nivel:

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

C++ está considerado por muchos como el lenguaje más potente, debido a que permite trabajar tanto a alto como a bajo nivel, sin embargo es a su vez uno de los que menos automatismos trae (obliga a hacerlo casi todo manualmente al igual que C) lo que "dificulta" mucho su aprendizaje. [CPLUS]

1.3.9 PERL.

Perl (Lenguaje Práctico para la Extracción e Informe) es un lenguaje de programación diseñado por Larry Wall y creado en 1987. Perl fue originalmente desarrollado para la manipulación de texto y en la actualidad es ampliamente utilizado en la administración de sistemas y en el desarrollo Web. Perl toma características del C y del lenguaje interpretado bash. La estructura completa de perl deriva de lenguaje C, perl es un lenguaje imperativo, con variables, expresiones, asignaciones y bloques de códigos delimitados por llaves. Perl tiene una gran potencia en la manipulación de textos debido a que incluye expresiones regulares que facilitan el trabajo con textos. [PRL00]

1.3.10 UML.

La Tecnología de Orientación a Objetos constituye la base de la reutilización de código por medio de componentes. UML (Lenguaje de Modelación Unificado o Unified Modeling Language, en inglés) es el lenguaje estándar adoptado por el OMG (Object Management Group) y mundialmente aceptado para la descripción de los "planos" de software. UML es un lenguaje gráfico para visualizar, especificar, construir y documentar cada una de las partes que comprende el desarrollo de software. UML posee formas de modelar conceptos como lo son procesos de negocio y funciones de sistema, además de aspectos concretos como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables. [UML00]

1.4 Conclusiones.

En este capitulo se a cumplido su objetivo principal, que ha sido mostrar al lector de forma general los aspectos teóricos a tener en cuenta para el desarrollo de un sistema de clonación y distribución de imágenes de sistemas operativos. Con la intención de ubicarlo y sumergirlo en el mundo de los servicios de soporte e instalación remota, así como mostrarle los diferentes sistemas existentes actualmente y sus características y funcionamiento.

A la vez se ha hecho un estudio valorativo de las herramientas, tecnologías y lenguajes propuestos a utilizar para el futuro desarrollo del sistema a proponer en este trabajo.

Capítulo 2. Características del Sistema

En este capítulo se hace una descripción del objeto de estudio del trabajo, se describe el entorno de trabajo sobre el que se desarrolla la aplicación. Además, se hace una propuesta del sistema y se analizan sus los requerimientos funcionales y no funcionales. Se realiza el modelado del negocio y se definen los casos de usos y los actores que intervienen en cada caso de uso. También se muestran los diagramas realizados para cada caso de uso.

2.1 Objeto de estudio.

En todo centro que opte por un desarrollo y expansión tecnológica, resulta de vital importancia el uso de los sistemas informáticos. Dada las facilidades que brindan en cuanto a la búsqueda y administración de la información, los sistemas informáticos se ha convertido a nivel mundial, en herramientas de desarrollo de las empresas.

Con el objetivo de formar especialistas en informática, y teniendo en cuenta el binomio docencia-producción, la Universidad de las Ciencias Informáticas (UCI) ha construido una serie de laboratorios docentes, en los cuales los estudiantes ponen en práctica los conocimientos teóricos adquiridos en las aulas. Estos laboratorios cuentan con más de 20 computadoras personales cada uno, las cuales tienen instalados diferentes sistemas operativos, personalizados de acuerdo con las necesidades del perfil de cada facultad.

2.1.1 Problema y situación problémica.

Actualmente, el proceso de instalación y personalización de los sistemas operativos en las computadoras de los laboratorios de la UCI se realiza con los sistemas UDPcast y Northon-Ghost. Estas aplicaciones brindan la posibilidad de instalar imágenes personalizadas de sistemas operativos en las computadoras. El problema es que para realizar el proceso de instalación con estas aplicaciones en cada computadora es necesario ejecutar una serie de comandos o acciones, lo que convierte este proceso en un proceso manual. Como es conocido, el número de laboratorios construidos en la UCI y que actualmente cuentan

con más de 20 computadoras, es considerablemente grande sin contar los que aun están por construir. Por tal motivo, cada vez se hace más necesario una mayor rapidez y eficiencia en el proceso de instalación de imágenes de sistemas operativos en los laboratorios docentes.

El proceso actual establecido en la UCI para el soporte y mantenimiento: particionamiento, instalación de sistemas operativos, instalación de imágenes; de las máquinas en los laboratorios docentes se lleva a cabo manualmente por un grupo de técnicos de la universidad. Teniendo en cuenta que a medida que la universidad crece y con ella el número de laboratorios, se hace cada vez más difícil mantener todas las computadoras en condiciones óptimas, para ser utilizadas por los estudiantes.

Pudiera pensarse que una solución a este problema seria la contratación de más personal calificado para la atención y mantenimiento de los laboratorios docentes, pero esto traería como consecuencia que para realizar de instalación de imágenes personalizadas en un tiempo relativamente corto, se necesitaría una cantidad de personal elevada. Además, hay que tener en cuenta la posible introducción de errores por parte del personal en el manejo de los comandos de las aplicaciones utilizadas para realizar los diferentes pasos que componen el proceso. Por tal motivo esta no se considera una solución eficiente para resolver los problemas mencionados.

2.1.2 Objeto de automatización.

Teniendo en cuenta los problemas planteados anteriormente, se hace necesario desarrollar un sistema capaz de automatizar el proceso de instalación de imágenes personalizadas en los laboratorios docentes. Un sistema que sea capaz de automatizar cada paso de este proceso como son el particionamiento y la gestión de las imágenes a instalar.

2.1.3 Información que se maneja.

La información que se manipula serán algunos datos del hardware de las computadoras a las que se le va a realizar el proceso. Estos datos pueden ser: el tamaño y tipo del disco duro y el tipo de tarjeta de red. Además se manipularán datos referentes a las configuraciones a aplicar, que pueden ser: cantidad de particiones a realizar, tamaño de cada una de ellas y la dirección de la imagen a instalar.

2.1.4 Propuesta del sistema.

En el presente trabajo se propone un sistema que brinde la posibilidad de realizar un particionamiento

simultaneo a todas las computadoras de una subred, teniendo en cuenta la posible variedad de hardware entre las computadoras, así como la distribución e instalación simultanea de las imágenes personalizadas para grupos de máquinas en específico o para máquinas individuales. Al usuario le será posible realizar una nueva configuración a la tabla de particiones de un computadora cliente o un Grupo de computadora clientes seleccionado por el, posibilitándole también escoger el sistema de fichero que tendrá cada partición creada, así como la selección de la imagen a instalar en una determinada partición.

2.2 Modelo de dominio.

Debido al bajo nivel de estructuración que presenta el negocio que se esta estudiando, ya que se centra en su mayoría en herramientas y tecnologías informáticas, se propone un modelo de dominio que nos permita visualizar los principales conceptos que se manejan en el sistema a desarrollar. Trayendo consigo un mejor entendimiento del contexto en que se emplaza el sistema por parte de los usuarios, desarrolladores e interesados.

El modelo del dominio cuenta con un diagrama de clases UML donde se especifican las principales clases conceptuales que pueden intervenir en el sistema a desarrollar, como muestra la figura 2.1.

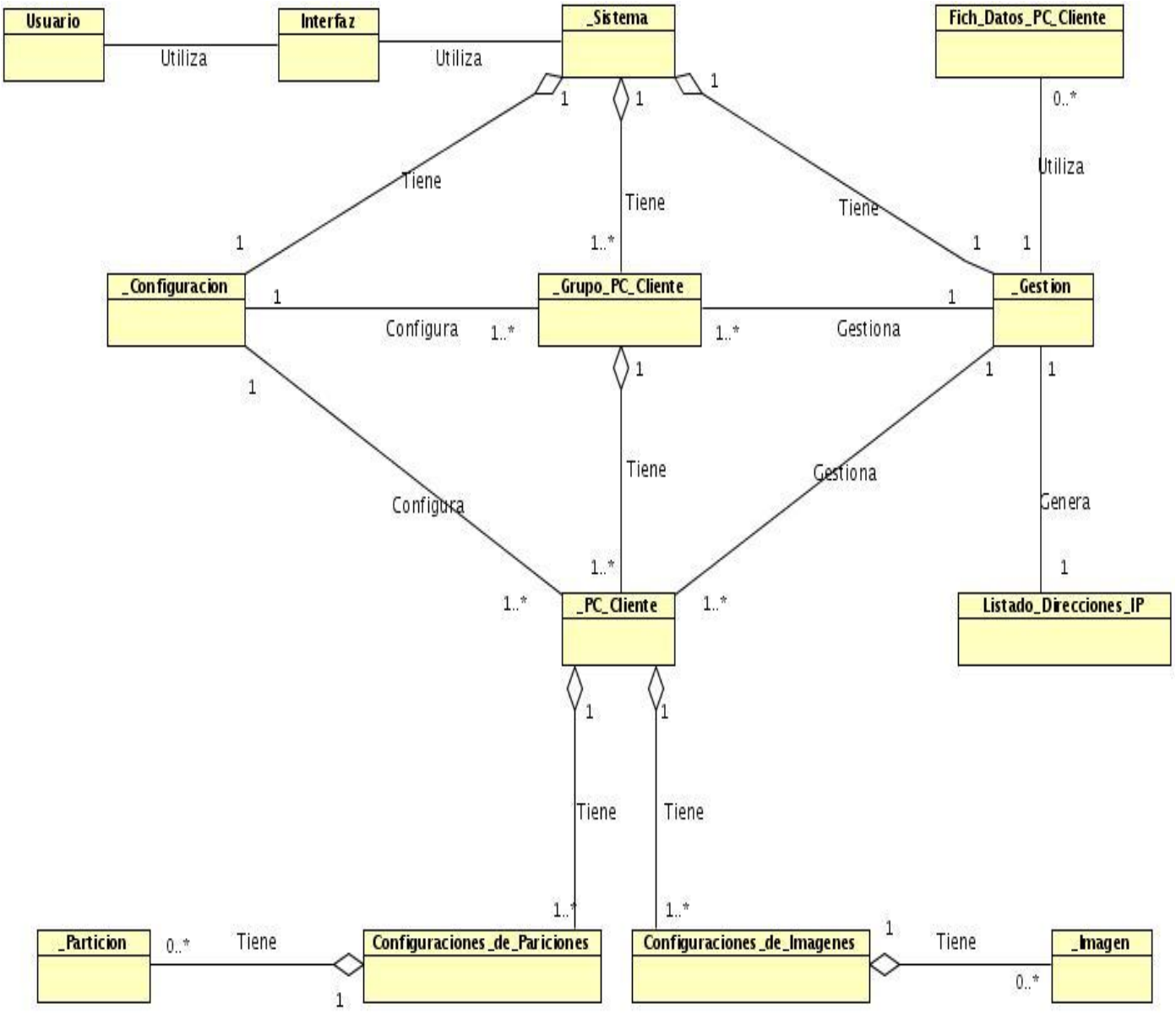

Figura 2.1 Diagrama del modelo de dominio.

2.3 Especificación de los requisitos del software.

2.3.1 Dependencias y relaciones con otro software.

Para lograr un óptimo funcionamiento del sistema SCDI, es necesario el uso de una serie de software, de los cuales SCDI se vale para realizar las funciones para las que fue concebido. Entre los software que se utilizaron se encuentran:

- LTSP (Linux Terminal Server Project): para la creación de un cluster entre las PC Clientes.

- Fdisk: para realizar el particionamiento

- Partimage: para realizar la copia de las imágenes en cada cliente

2.3.2 Requerimientos funcionales.

Los requerimientos funcionales definen las funciones que el sistema será capaz de realizar.

1. Gestionar listado de IP de los clientes.

   1.1. Obtener direcciones IP. El sistema debe obtener la dirección IP de cada cliente conectado y adicionarlo a una lista.

   1.2. Organizar listado de direcciones IP. El sistema debe poder organizar en grupos las direcciones IP de las PC de acuerdo con su hardware.

   1.3. Mostrar el listado de IP de los clientes. El sistema debe poderle mostrar al técnico el listado de las direcciones IP de los clientes que están conectados.

2. Realizar particionamiento de discos. El sistema debe permitir al técnico realizar el ¿? particionamiento de cada PC.

   2.1 Crear partición. El sistema debe permitir al técnico crear una partición.

      2.1.1 Definir sistema de archivo de la partición. El sistema debe permitir al técnico elegir el tipo de sistema de archivo que tendrá la partición a crear.

      2.1.2 Mostrar espacio disponible. El sistema debe mostrar el espacio disponible para

particionar.

2.1.3 Definir tamaño de la partición. El sistema debe permitir al técnico escoger el tamaño que tendrá la partición a crear.

2.1.4 Definir tipo de partición. El sistema debe permitirle al usuario definir el tipo de partición a crear; Extendida, Lógica, Primaria.

2.2 Eliminar partición. El sistema debe permitir al técnico eliminar una partición que fue creada por el previamente.

3. Gestionar imagen a instalar. El sistema debe permitir al técnico gestionar la imagen a instalar en una partición.

3.1 Entrar dirección de la imagen a instalar. El sistema debe permitir al técnico entrar la dirección de la imagen a instalar.

3.2 Escoger destino de la imagen a instalar. El sistema debe permitirle al usuario escoger la partición en donde se instalará la imagen.

4. Actualizar el listado de IP de los clientes. El sistema debe mantener actualizado el listado de direcciones IP de los clientes que están conectados.

5. Gestionar información.

5.1 Tamaño del disco duro. El sistema debe ser capaz gestionar el tamaño del disco duro.

5.2 Tipo de disco duro. El sistema debe ser capaz gestionar el tipo del disco duro.

5.3 Número de particiones existentes. El sistema debe ser capaz gestionar el número de particiones existentes.

5.4 Dirección IP. El sistema debe ser capaz gestionar la dirección IP de la PC cliente.

6. Borrar tabla de particiones. El sistema debe borrar previamente la tabla de particiones del cliente antes de efectuar operaciones.

7. Efectuar operaciones.

   7.1 Particionar.

   7.2 Crear partición. El sistema debe ser capaz de crear una partición.

   7.3 Formatear partición. El sistema debe ser capaz de dar formato con un tipo de sistema de archivo a una partición determinada.

   7.4 Instalar Imagen. El sistema debe ser capaz de instalar una imagen en una partición determinada.

8. Mostrar información del cliente. El sistema debe ser capaz de mostrar información de cualquier cliente conectado al sistema.

2.3.3 Requerimientos no funcionales.

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Son las características que hacen que el producto sea atractivo, usable, rápido o confiable.

Apariencia o Interfaz externa.

El sistema tendrá una interfaz de usuario sencilla y amigable.

Usabilidad.

El sistema permitirá al usuario realizar las operaciones con rapidez y sin complicaciones.

Soporte.

Extensibilidad: El sistema permitirá su extensibilidad, permitiendo agregar nuevas funcionalidades en un futuro.

Mantenimiento: El sistema debe ser bien documentado de forma tal que en caso de mantenimiento el tiempo que se requiera sea el mínimo.

Requerimientos de Portabilidad.

El sistema deberá funcionar sobre plataforma GNU/Linux.

El sistema podrá ser portado en un LiveCD.

2.4 Definiciones de casos de uso.

Los casos de uso constituyen una técnica narrativa para describir el comportamiento del sistema y sus funcionalidades. Cada caso de uso puede describir una o más funcionalidades requeridas para el sistema por parte del usuario.

2.4.1 Definición de los actores.

En la siguiente tabla se muestra una descripción del rol que desempeña cada uno de los actores del sistema.

Tabla 2.1 Definición y descripción de los actores del sistema.

Actores	Justificación
Usuario	Es la persona encargada de operar el sistema, quien se encarga de configurar y ejecutar las operaciones definidas por el sistema.
Sistema Operativo	Se refiere al temporizador del sistema operativo quien desencadena las acciones temporizadas en el sistema. Es un actor abstracto.

2.4.2 Listado de casos de uso del sistema.

Como se puede ver, en las siguientes tablas se hace una descripción de los casos de uso del sistema, definiéndose el actor que inicializa cada caso de uso y una breve reseña de las funcionalidades que cumple cada uno.

Tabla 2.2 CU - Gestionar listado de direcciones IP.

CU-1	Gestionar Listado de direcciones IP

Actor	Sistema Operativo
Descripción	Se realiza al iniciar el sistema y consiste en gestionar y crear un listado organizado por grupos, de las direcciones IP de los PC clientes que están conectados con el sistema, este listado se mantendrá en constante actualización.
Referencia	RF-1

Tabla 2.3 CU - Configurar instalación de imagen.

CU-2	Configurar instalación de imagen
Actor	Usuario
Descripción	El usuario realiza las configuraciones necesarias para gestionar la imagen a clonar en una partición determinada de un PC cliente.
Referencia	RF-3

Tabla 2.4 CU - Gestionar información del PC cliente.

CU-3	Gestionar información del PC cliente
Actor	Sistema Operativo
Descripción	Se realiza automáticamente una vez que se inicia el sistema y consiste en gestionar toda la información referente a cada PC cliente; tipo de disco duro, tamaño, dirección IP, cantidad de particiones existentes. Información que será necesaria para realizar las operaciones definidas por el usuario.
Referencia	RF-5

Tabla 2.5 CU - Aplicar configuraciones.

CU-4	Aplicar configuraciones
Actor	Usuario
Descripción	El usuario realiza la aplicación de las configuraciones previamente hechas para cada uno de los PC clientes.
Referencia	RF-7

Tabla 2.6 CU - Mostrar información del cliente.

CU-5	Mostrar información del cliente
Actor	Usuario
Descripción	Se realiza automáticamente una vez que el usuario selecciona un cliente en el listado de direcciones IP de los PC clientes conectados al sistema.
Referencia	RF-8

Tabla 2.7 CU - Configurar particionamiento.

CU-6	Configurar particionamiento
Actor	Usuario
Descripción	El usuario solicita realizar la configuración del particionamiento; agregar partición, eliminar partición, de un PC cliente.
Referencia	RF-2

2.4.3 Diagrama de casos de uso.

La figura 2.2 muestra una representación gráfica de los casos de uso del sistema así como su relación con los actores del sistema, los cuales son los encargados de inicializar en interactuar con cada caso de uso.

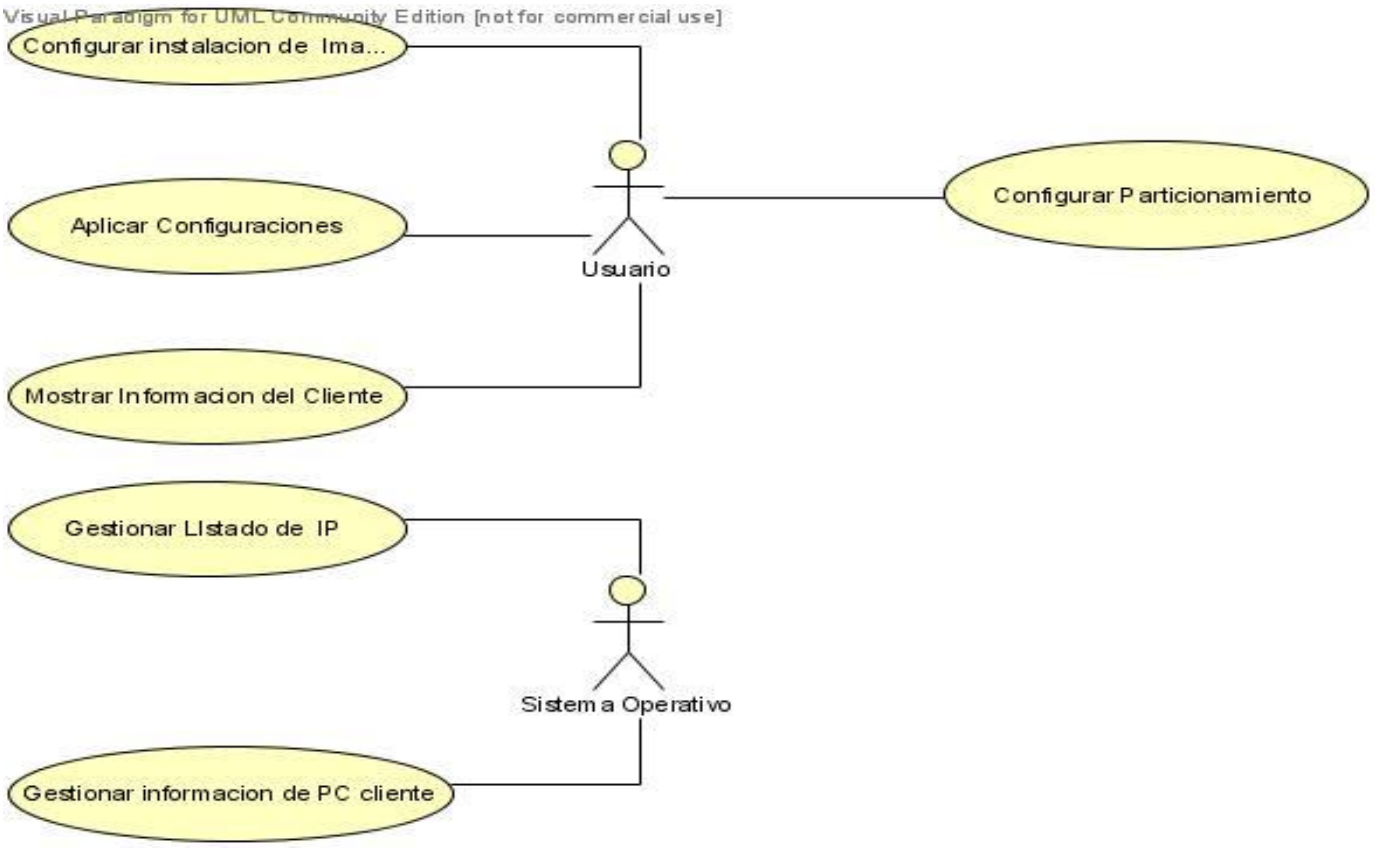

Figura 2.2 Diagrama de Casos de Uso del Sistema

2.4.4 Casos de Uso por Ciclo

Tabla 2.8 Descripción de casos de uso por ciclo.

Ciclo de desarrollo primario			
Cód	**Nombre del caso de uso**	**Paquete**	**Justificación**

CU-1	Gestionar Listado de IP		• Son los casos de uso que representan los procesos primarios imprescindibles para el funcionamiento básico del sistema e influyen en la arquitectura básica.
CU-2	Configuración de Instalación de Imagen		
CU-3	Gestionar Información del PC Cliente		
CU-4	Aplicar Configuraciones		
CU-6	Configurar Particionamiento		
Ciclo de desarrollo Secundario			
CU-5	Mostrar Información del PC Cliente		• Son casos de uso importantes pero no imprescindible para el funcionamiento básico del sistema.

2.4.5 Casos de uso expandidos.

En las siguientes tablas se hace una descripción más detallada de cada caso de uso del sistema. Cada tabla muestra en secciones una secuencia de las acciones (actores) y respuestas del sistema para cada una de las funcionalidades del caso de uso. Cada tabla puede tener una o más secciones en dependencia de la complejidad de cada caso de uso.

Tabla 2.9 Descripción del caso de uso "Gestionar listado de IP".

Caso de Uso	
CU-1	Gestionar Listado de IP
Propósito	Gestionar un listado con las direcciones IP de los PC clientes que se encuentran conectados con el sistema.

Actores	Sistema Operativo
Resumen: Una vez que se inicie el sistema, el sistema operativo comienza la gestión del listado con las direcciones IP, una vez concluida la gestión, el sistema le muestra el listado obtenido en la interfaz principal. El sistema realizará este proceso constantemente para actualizar el listado en caso de conexión o desconexión de un PC cliente.	
Referencias	RF-1

Sección 1 : Generar listado de direcciones IP	
Acción del actor	**Respuesta del sistema**
1- El sistema operativo verifica que se ha iniciado el sistema.	
	2- El sistema busca todos los clientes que se encuentran conectados al sistema y genera una lista con sus direcciones IP.
	3- El sistema muestra en una interfaz el listado generado.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Puntos de extensión	

Sección 2: Actualizar listado de direcciones IP	
Acción del actor	**Respuesta del sistema**

1- El sistema operativo verifica si ha transcurrido el tiempo indicado y activo la operación definida.	
	2- El sistema genera un nuevo listado acuatizado con las direcciones IP de los PC clientes conectados al sistema.
	3- El sistema actualiza el listado generado originalmente con los datos del listado nuevo generado.
	4- El sistema muestra en una interfaz el listado original actualizado.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Puntos de extensión	

Tabla 2.10 Descripción del caso de uso "Configuración de instalación de imagen".

Caso de Uso	
CU-2	Configuración de instalación de imagen
Propósito	Realizar las configuraciones necesarias para la gestión y clonación de la imagen en una PC cliente.
Actores	Usuario
Resumen: El usuario solicita realizar la configuración para la gestión de la imagen a clonar en una PC cliente determinada.	

Referencias	RF-3
Acción del actor	**Respuesta del sistema**
1- El caso de uso se inicia cuando el usuario solicita realizar la configuración de la imagen a clonar en una PC cliente determinada	
	2- El sistema muestra una interfaz, para la gestión y configuración de la imagen.
3- El usuario entra los datos pertinentes para la gestión y configuración.	
	4- El sistema retorna a la interfaz principal.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Puntos de extensión	

Tabla 2.11 Descripción del caso de uso "Gestionar información del PC cliente".

Caso de Uso	
CU-3	Gestionar información del PC cliente
Propósito	Gestionar la información del PC cliente, necesaria para que el usuario pueda realizar las operaciones definidas en el sistema.
Actores	Sistema Operativo
Resumen: Cuando el sistema operativo detecta que se ha iniciado el sistema, automáticamente se pone	

en marcha la gestión de la información necesaria para el correcto funcionamiento del sistema.	
Referencias	RF-5
Acción del actor	**Respuesta del sistema**
1- Se inicia cuando el sistema operativo detecta que se a iniciado el sistema	
	2- El sistema gestiona la información necesaria en el cliente y genera un archivo con dicha información.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Puntos de extensión	

Tabla 2.12 Descripción del caso de uso "Aplicar configuraciones".

Caso de Uso	
CU-4	Aplicar configuraciones
Propósito	Aplica en los PC clientes las configuraciones realizadas previamente por el usuario.
Actores	Usuario
Resumen: Una vez que se han terminado las configuraciones para cada cliente el usuario las pone en ejecución.	
Referencias	RF-7
Acción del actor	**Respuesta del sistema**

1- El usuario solicita aplicar las configuraciones y efectuar las operaciones definidas en dichas configuraciones.	
	2- El sistema efectúa la aplicación de las configuraciones y operaciones definidas por el usuario para cada PC cliente.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Puntos de extensión	

Tabla 2.13 Descripción del caso de uso "Mostrar información del cliente".

Caso de Uso	
CU-5	Mostrar información del cliente
Propósito	Mostrar informaciones y características de cada cliente
Actores	Usuario
Resumen: Le muestra al usuario características determinadas de un cliente que el determine	
Referencias	RF-8
Acción del actor	**Respuesta del sistema**

1- El usuario selecciona un cliente	
2- El usuario solicita información del cliente seleccionado.	.
	3- El sistema le muestra una interfaz con la información del cliente seleccionado.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Puntos de extensión	

Tabla 2.14 Descripción del caso de uso "Configurar particionamiento".

Caso de Uso	
CU-6	Configurar particionamiento.
Propósito	Configurar el sistema de particiones que tendrá un PC cliente.
Actores	Usuario
Resumen: Le permite al usuario configurar el sistema de particiones de un cliente o grupo de cliente determinado, efectuando las operaciones de adicionar partición, y eliminar partición, posibilitando que el usuario escoja el tamaño, tipo, y sistema de fichero que tendrá la partición a adicionar.	
Referencias	RF-2
Sección 1 : Adicionar Partición	
Acción del actor	**Respuesta del sistema**

1- El usuario solicita agregar una partición a la configuración.	
	2- El sistema le muestra una interfaz para entrar los datos de la nueva partición.
3- El usuario entra los datos necesarios para agregar la partición.	
4- El usuario agrega la partición.	
	5- El sistema retorna a la interfaz de configuración de particiones.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Puntos de extensión	
Sección 2: Eliminar Partición	
Acción del actor	**Respuesta del sistema**
1- El usuario solicita eliminar una partición	
	2- El sistema muestra una interfaz con un listado con las particiones creadas previamente.
3- El usuario selecciona la partición a eliminar	

4- El usuario elimina la partición.	
	5- El sistema retorna a la interfaz de configuración de particiones.
Flujo alternativo	
Acción del actor	**Respuesta del sistema**
Puntos de extensión	

2.5 Conclusiones.

En este capítulo se han planteado alguno de los problemas fundamentales presentes en el proceso de soporte e instalación de los sistemas operativos en las computadoras de los laboratorios docentes de la UCI y el por que se hace necesario proponer un sistema de clonación y distribución de imágenes. También se hace una propuesta de un sistema que pueda satisfacer las necesidades de la UCI para una realización rápida y eficiente del proceso de soporte de los laboratorios docentes. Se hace además un análisis de los requerimientos funcionales y no funcionales del sistema, facilitando la identificación de los casos de uso del sistema.

Capítulo 3. Análisis y Diseño del Sistema

En el presente capítulo se abordan cuestiones fundamentales sobre el análisis y diseño del sistema, utilizando para su modelado el Lenguaje Unificado de Modelación (UML). Se plasman los resultados de estas etapas, incluyéndose el modelo de clases de análisis, los diagramas de secuencia y los diagramas de clases del diseño, así como una descripción detallada de las clases empleadas en el diseño del sistema.

3.1 Análisis.

Esta es la fase en la que se analizan los requisitos definidos en la captura de requisitos, se refinan y estructuran. Con el objetivo de conseguir una comprensión más precisa de los requisitos y una descripción de los mismos que sea fácil de mantener y que nos ayude a estructurar el sistema, incluyendo su arquitectura, se formula además el modelo de dominio del problema permitiéndonos responder la pregunta ¿Que hacer? .

3.1.1 Diagrama de clases del análisis.

El Diagrama de Clase es el diagrama principal de análisis y diseño para un sistema. En él, la estructura de clases del sistema se especifica, con relaciones entre clases y estructuras de herencia. Durante el análisis del sistema, el diagrama se desarrolla buscando una solución ideal. Se realiza a través de un diagrama de clases de UML simplificado, en el cual se representan las clases preliminares, las asociaciones preliminares entre las clases, y los atributos de las clases. Esto se muestra en la **figura 3.1**.

Una clase de análisis representa la abstracción de una o varias clases y/o subsistemas del diseño del sistema. Posee un conjunto de características tales como: se centra en el tratamiento de los requisitos funcionales y pospone los no funcionales, su comportamiento se define mediante responsabilidades a un nivel más alto y menos formal que las operaciones y signaturas, define atributos y participa en relaciones, entre otras.

Con el objetivo de facilitar el análisis, las clases se clasifican en:

- **Clases de interfaz:**
  - Modelan la interacción entre el sistema y sus actores.
  - Reciben y presentan información y peticiones de y hacia los actores.
  - Reúnen los requisitos en los límites del sistema.
  - Suelen ser abstracciones de ventanas, formularios, interfaces de impresoras, censores, terminales y otros artefactos de intercambio de información del sistema.
  - Describen lo que se obtiene con las interacciones, no el proceso físico de cómo se ejecuta.
- **Clases de entidad:**
  - Modelan información que poseen una larga vida.
  - Modelan la información y el comportamiento asociado a algún fenómeno o concepto, como una persona, un objeto o un suceso del mundo real.
  - Derivan normalmente de una clase entidad del negocio.
  - Pueden tener comportamiento complejo
  - Aísla los cambios en la información que representa.
- **Clases de control:**
  - Representan coordinación, secuencia, transacciones, y control de otros objetos.
  - Se usan mucho para capsular el control de un caso de uso concreto.
  - Manejan y coordinan las acciones y los flujos de control principales, y delegan trabajo a otros objetos (de interfaz y de entidad).

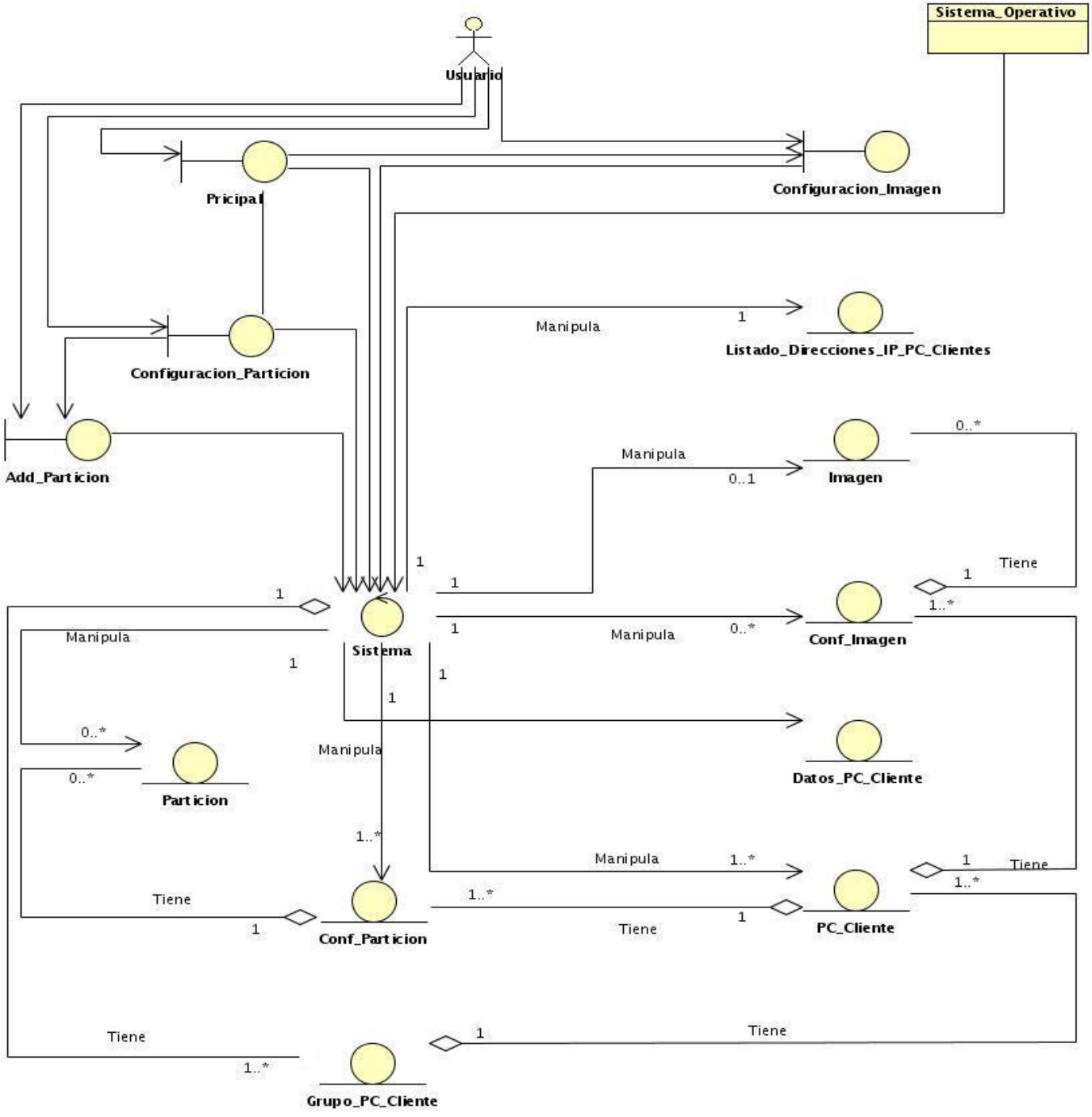

Figura 3.1 Diagrama de clases del análisis.

3.2 Diseño.

3.2.1 Diagramas de secuencia del sistema.

Con la idea de dar una visión gráfica de las interacciones de los actores con el sistema, se utilizan los diagramas de secuencia del sistema (DSS), los cuales muestran qué hace el sistema ante el medio, sin explicar el cómo. [DSS00]

Para cada uno de los casos de uso se define un diagrama de secuencia del sistema. Los diagramas de secuencias del sistema de cada caso de uso se muestran en el **anexo 1.**

3.2.2 Diagrama de clases del diseño.

El diagrama de clases del diseño que se muestra en la figura 3.2, no es más que una representación gráfica de las clases que serán implementadas en el sistema. Este diagrama muestra las especificaciones y detalles más concretos de cada clase del sistema así como su relación de asociación o composición agregación existentes entre ellas.

Estos diagramas se muestran en el **anexo 2.**

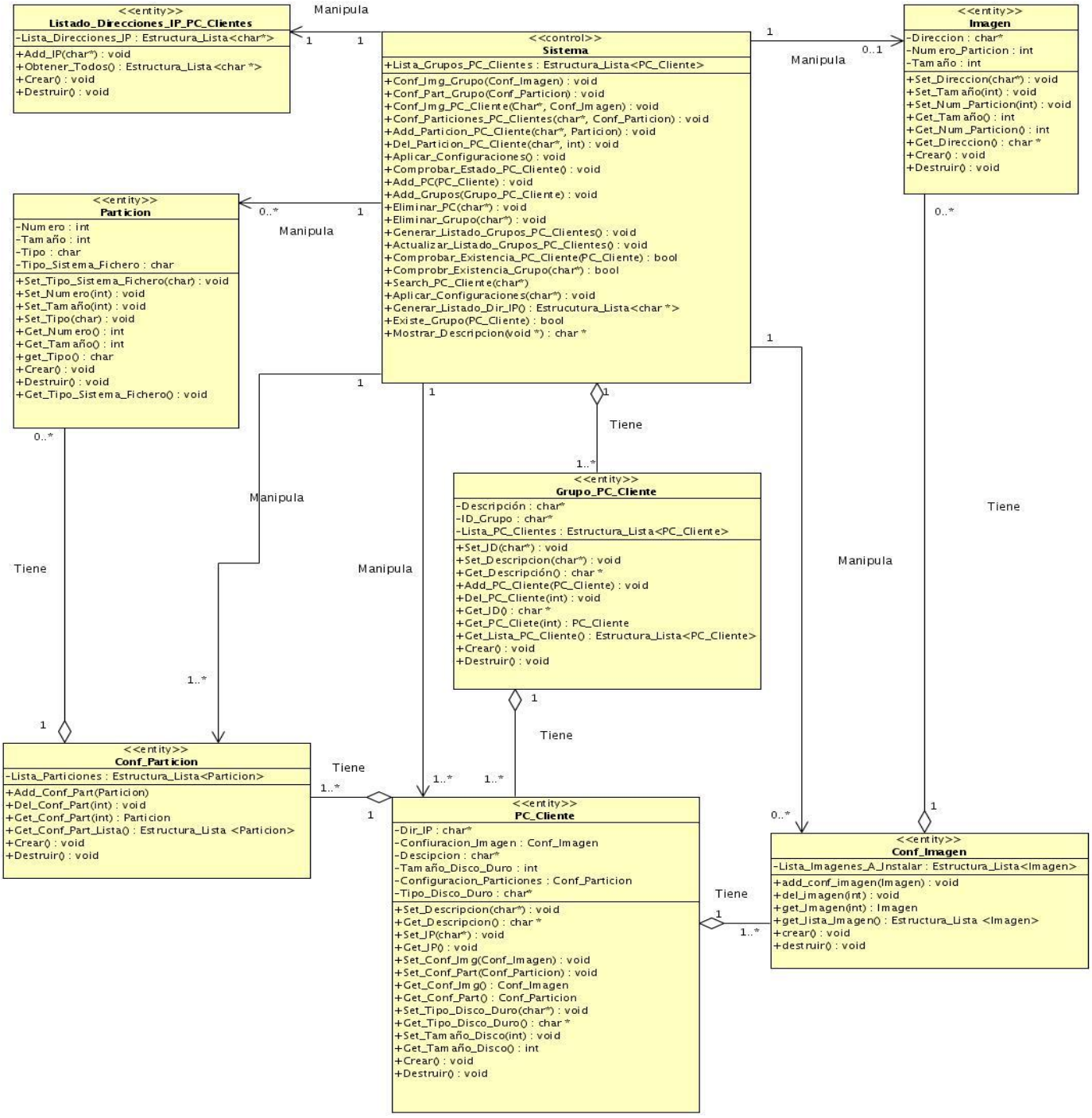

Figura 3.2. Diagrama de clases del diseño.

3.3 Descripción de las clases del diseño.

Con el objetivo de lograr un mayor entendimiento de cada una de las funcionalidades de las clases que componen el diagrama de clases del diseño del sistema, se han conformado una serie de tablas donde se describe detalladamente cada una de estas clases, mostrando cada uno de los atributos y funciones que componen cada clase, explicando a su vez el funcionamiento de cada función, los parámetros que reciben y el valor de retorno.

Tabla 3.1 Descripción de la clase Sistema.

Nombre: Sistema	
Tipo: Controladora	
Atributo	**Tipo**
Lista_Grupos_PC_Clientes	Estructura_Lista <PC_Cliente>
Para cada responsabilidad:	
Nombre:	Conf_Img_Crupo(char*,Conf_Imagen): void
Descripción:	Realiza las configuraciones de instalación de imagen a todas las PC clientes pertenecientes a un grupo. Se le entran como parámetros el nombre del grupo y la configuración de instalación de imagen a establecer.
Nombre:	Conf_Part_Crupo(char*,Conf_Particion): void
Descripción:	Realiza las configuraciones de particiones a todas las PC clientes pertenecientes a un grupo. Se le entra como parámetros el nombre del Grupo_Cliente y la configuración de particiones a establecer.
Nombre:	Conf_Img_PC_Cliente(char*, Conf_Imagen): void
Descripción:	Realiza las configuraciones de instalación de imagen a un PC cliente dado. Se le

	entra como parámetros la dirección IP del PC_Cliente y la configuración de instalación de imagen a establecer.
Nombre:	Conf_Particiones_PC_Cliente(char*, Conf_Particion): void
Descripción:	Realiza las configuraciones de particiones a un PC_Cliente dado. Se le entra como parámetros la dirección IP del PC_Cliente y la configuración de particiones a establecer.
Nombre:	Add_Particion_PC_Cliente(char*, Particion): void
Descripción:	Adiciona una partición a la configuración de particiones de un PC_Cliente dado. Se le entra como parámetros la dirección IP del PC_Cliente y la configuración de particiones a adicionar.
Nombre:	Del_Particion_PC_Cliente(char*, int) : void
Descripción:	Borra una partición de las configuraciones de particiones de un PC_Cliente dado. Se le entra como parámetros la dirección IP del PC_Cliente y el número de la partición a borrar.
Nombre:	Aplicar_Configuraciones(): void
Descripción:	Aplica las configuraciones definidas para cada PC_Cliente.
Nombre:	Comprobar_Estado_PC_Cliente(): void
Descripción:	Comprueba si los PC clientes continúan conectados con el sistema.
Nombre:	Add_PC(PC_Cliente): void
Descripción:	Adiciona un PC_Cliente. Se le entra como parámetros el PC_Cliente a adicionar.
Nombre:	Add_Grupo(Grupo_Cliente): void
Descripción:	Adiciona un Grupo a la lista de grupos de PC clientes. Se le entra como parámetros

	el Grupo_Cliente a adicionar.
Nombre:	Eliminar_PC(char*): void
Descripción:	Elimina un PC_Cliente de la lista de grupos de PC clientes. Se le entra como parámetros la dirección IP del PC_Cliente a eliminar.
Nombre:	Eliminar_Grupo(char*) : void
Descripción:	Elimina un Grupo_Cliente de la lista de grupos de PC_Cliente. Se le entra como parámetros el nombre del Grupo_Cliente a eliminar.
Nombre:	Generar_Listado_Grupos_PC_Clientes() : void
Descripción:	Genera el listado de grupos de PC_Cliente. Es quien se encarga de gestionar los datos de cada uno de los PC_Cliente desde ficheros y adicionar y confeccionar los grupos de acuerdo con las características de los PC clientes.
Nombre:	Actualizar_ Listado_Grupos_PC_Clientes() : void
Descripción:	Actualiza el listado de grupos de PC clientes en caso de una nueva conexión de un nuevo PC cliente o en caso de desconexión.
Nombre:	Comprobar_Existencia_PC_Cliente(PC_Cliente) : bool
Descripción:	Comprueba la existencia de un PC_Cliente dado. Se le entra como parámetro un PC_Cliente.
Nombre:	Comprobar_Existencia_Grupo(char*) : bool
Descripción:	Comprueba la existencia de un Grupo_Cliente con las características dadas. Se le pasa como parámetro una descripción de un PC_Cliente.
Nombre:	Search_PC_Cliente(char*): PC_Cliente
Descripción:	Busca un PC_Cliente en el listado de grupos de PC clientes y lo retorna. Se le pasa

	como parámetro la dirección IP del PC_Cliente a buscar.
Nombre:	Aplicar_Configuraciones(char*) : void
Descripción:	Aplica las configuraciones a un Grupo_Cliente o un PC_Cliente. Se le pasa como parámetro el nombre de un Grupo_Cliente o la dirección IP de un PC_Cliente.
Nombre:	Generar_Listado_Dir_IP() : void
Descripción:	Genera el listado de direcciones IP de los PC_Cliente conectados al sistema.
Nombre:	Existe_Grupo(PC_Cliente) : bool
Descripción:	Comprueba si existe algún grupo con las características de un PC_Cliente dado. Se le pasa por parámetro el PC_Cliente a establecer búsqueda.
Nombre:	Mostrar_Descripcion(void*): char*
Descripción:	Muestra la descripción de un Grupo_Cliente o un PC_Cliente dado. Se le pasa por parámetro un PC_Cliente o un Grupo_Cliente.

Tabla 3.2 Descripción de la clase Grupo_PC_Cliente.

Nombre: Grupo_PC_Cliente	
Tipo: Entidad	
Atributo	**Tipo**
Descripción	char*
ID_Grupo	char*

Lista_PC_Clientes	Estructura_Lista<PC_Cliente>
Para cada responsabilidad:	
Nombre:	Set_ID(char*) : void
Descripción:	Entra el ID del grupo. Se le pasa por parámetro el ID
Nombre:	Set_Descripcion(char*) : void
Descripción:	Entra la descripción del grupo. Se le pasa por parámetro la descripción.
Nombre:	Get_Descripcion() : char*
Descripción:	Devuelve la descripción del grupo.
Nombre:	Add_PC_Clinete(PC_Cliente) :void
Descripción:	Adiciona un PC_Cliente a la lista de clientes del grupo. Se le pasa por parámetro el PC_Cliente a adicionar.
Nombre:	Del_PC_Cliente(int) : void
Descripción:	Elimina un PC_Cliente de la lista de PC_Cliente del grupo. Se le pasa por parámetro la dirección IP del PC_Cliente a eliminar.
Nombre:	Get_ID() : char*
Descripción:	Devuelve el ID del grupo.
Nombre:	Get_PC_Cliente(int): PC_Cliente
Descripción:	Busca un PC_Cliente con una dirección IP dada y lo devuelve. Se le pasa por parámetro la dirección IP del PC_Cliente a buscar.
Nombre:	Get_Lista_PC_Cliente() : Estructura_Lista<PC_Cliente>

Descripción:	Devuelve la lista de PC_Cliente del grupo.
Nombre:	Crear() : void
Descripción:	Crea un objeto Grupo_PC_Cliente.
Nombre:	Destruir() : void
Descripción:	Destruye el objeto Grupo_PC_Cliente.

Tabla 3.3 Descripción de la clase PC_Cliente.

Nombre: PC_Cliente	
Tipo: Entidad	
Atributo	**Tipo**
Dir_IP	char*
Configuracion_Imagen	Conf_Imagen
Descripcion	char*
Tamaño_Disco_Duro	Int
Configuracion_particiones	Conf_Particion
Tipo_Disco_Duro	char*
Para cada responsabilidad:	
Nombre:	Set_Descripcion(char*) : void

Descripción:	Entra la descripción del PC_Cliente. Se le pasa por parámetro la descripción.
Nombre:	Get_Descripcion() : char*
Descripción:	Devuelve la descripción del PC_Cliente.
Nombre:	Set_IP(char*) : void
Descripción:	Entra la dirección IP del PC_Cliente.
Nombre:	Get_IP() : char*
Descripción:	Devuelve la dirección IP del PC_Cliente.
Nombre:	Set_Conf_Img(Conf_Imagen) : void
Descripción:	Entra la configuración de las imágenes a instalar en el PC_Cliente
Nombre:	Get_Conf_Img() : Conf_Imagen
Descripción:	Devuelve las configuraciones de imágenes del PC_Cliente.
Nombre:	Set_Conf_Part(Conf_Particion) : void
Descripción:	Entra la configuración de particiones del PC_Cliente.
Nombre:	Get_Conf_Part() : Conf_Particion
Descripción:	Devuelve la configuración de particiones del PC_Cliente
Nombre:	Set_Tipo_Disco_Duro(char*) : void
Descripción:	Entra el tipo de disco duro del PC_Cliente.
Nombre:	Get_Tipo_Disco_Duro() : char*
Descripción:	Devuelve el tipo de disco duro del PC_Cliente.

Nombre:	Set_Tamaño_Disco(int) : void
Descripción:	Entra el tamaño del disco duro del PC_Cliente
Nombre:	Get_Tamaño_Disco() : int
Descripción:	Devuelve el tamaño del disco duro del PC_Cliente.
Nombre:	Crear() : void
Descripción:	Crea un objeto de PC_Cliente.
Nombre:	Destruir() : void
Descripción:	Destruye un objeto de PC_Cliente.

Tabla 3.4 Descripción de la clase Conf_Particiones.

Nombre: Conf_Particiones	
Tipo: Entidad	
Atributo	**Tipo**
Lista_Particiones	Estructura_Lista <Particion>
Para cada responsabilidad:	
Nombre:	Add_conf_Part(Particion) : void
Descripción:	Adiciona una partición a la lista de particiones. Se le pasa por parámetro la partición a adicionar.

Nombre:	Del_Conf_Part(int) : void
Descripción:	Elimina una partición de la lista de particiones. Se le pasa por parámetro el número de la partición a eliminar.
Nombre:	Get_Conf_Part(int) : Particion
Descripción:	Devuelve una partición. Se le pasa por parámetro el número de la partición a devolver.
Nombre:	Get_Conf_Part_Lista() : Estructura_Lista<Particion>
Descripción:	Devuelve la lista de particiones.
Nombre:	Crear() : void
Descripción:	Crea un objeto de Conf_Particion.
Nombre:	Destruir() : void
Descripción:	Destruye un objeto de Conf_Particion.

Tabla 3.5 Descripción de la clase Conf_Imagen.

Nombre: Conf_Imagen	
Tipo: Entidad	
Atributo	**Tipo**
Lista_Imagenes_A_Instalar	Estructura_Lista<Imagen>
Para cada responsabilidad:	

Nombre:	Add_Conf_Imagen(Imagen) : void
Descripción:	Adiciona una imagen a la lista de imágenes.
Nombre:	Del_Imagen(int) : void
Descripción:	Elimina una imagen de la lista de imágenes. Se le pasa por parámetro la posición de la imagen a eliminar en la lista de imágenes.
Nombre:	Get_Imagen(int) : Imagen
Descripción:	Devuelve una imagen de la lista de imágenes. Se le pasa por parámetro la posición en la lista de la imagen a devolver.
Nombre:	Get_Lista_Imagen() : Estructura_Lista<Imagen>
Descripción:	Devuelve la lista de imágenes.
Nombre:	Crear() : void
Descripción:	Crea un objeto de Conf_Imagen.
Nombre:	Destruir() : void
Descripción:	Destruye un objeto de Conf_Imagen.

Tabla 3.6 Descripción de la clase Imagen.

Nombre: Imagen	
Tipo: Entidad	
Atributo	**Tipo**

Direccion	char*
Numero_Particion	Int
Tamaño	int
Para cada responsabilidad:	
Nombre:	Set_Direccion(char*) : void
Descripción:	Entra la dirección fuente de la imagen. Se le entra por parámetro la dirección.
Nombre:	Set_Tamaño(int) : void
Descripción:	Entra el tamaño de la imagen. Se le pasa por parámetro el tamaño.
Nombre:	Set_Numero_Particion(int) : void
Descripción:	Entra el número de la partición donde se instalara la imagen. Se pasa por parámetro el número.
Nombre:	Get_tamaño() : int
Descripción:	Devuelve el tamaño de la imagen.
Nombre:	Get_Num_particion() : int
Descripción:	Devuelve el número de la partición donde se instalará la imagen.
Nombre:	Get_Direccion() : char*
Descripción:	Devuelve la dirección fuente de la Imagen a instalar
Nombre:	Crear() : void
Descripción:	Crea un objeto de Imagen.

Nombre:	Destruir() : void
Descripción:	Destruye un objeto de Imagen.

Tabla 3.7 Descripción de la clase Particion.

Nombre: Particion	
Tipo: Entidad	
Atributo	**Tipo**
Numero	int
Tamaño	Int
Tipo	char
Tipo_Sistema_Archivo	char
Para cada responsabilidad:	
Nombre:	Set_Numero(int) : void
Descripción:	Entra el número de la partición. Se le pasa por parámetro el número.
Nombre:	Set_Tamaño(int) : void
Descripción:	Entra el tamaño de la partición. Se le pasa por parámetro el tamaño.
Nombre:	Set_Tipo(char) : void
Descripción:	Entra el tipo de partición. Se le pasa por parámetro el tipo
Nombre:	Set_Tipo_Sistema_Fichero(char) : void

Descripción:	Entra el tipo de sistema de fichero. Se le pasa por parámetro el tipo (L, P, E).
Nombre:	Get_Numero() : int
Descripción:	Devuelve el número de la partición.
Nombre:	Get_Tamaño() : int
Descripción:	Devuelve el tamaño de la partición.
Nombre:	Get_Tipo() : char
Descripción:	Devuelve el tipo de partición.
Nombre:	Get_Tipo_Sistema_Archivo() : char
Descripción:	Devuelve el tipo de sistema de fichero que tendrá la partición.
Nombre:	Crear() : void
Descripción:	Crea un objeto de Particion.
Nombre:	Destruir() : void
Descripción:	Destruye un objeto de Particion.

Tabla 3.2 Descripción de la clase Lista_Direcciones_IP_PC_Clientes.

Nombre: Lista_Direcciones_IP_PC_Clientes	
Tipo: Entidad	
Atributo	**Tipo**
Lista_Direcciones_IP_Clientes	Estructura_Lista<char*>

Para cada responsabilidad:	
Nombre:	Add_IP(char*) : void
Descripción:	Adiciona una dirección IP a la lista.
Nombre:	Obtener_Todos() : Estructura_Lista<char*>
Descripción:	Devuelve la lista de direcciones IP.
Nombre:	Crear(): void
Descripción:	Crea un objeto de tipo Lista_Direcciones_IP_PC_Clientes
Nombre:	Destruir() : void
Descripción:	Destruye el objeto Lista_Direcciones_IP_PC_Clientes

3.4 Conclusiones.

En el presente capítulo se presentaron las etapas de análisis y diseño del sistema, habiéndose hecho el modelo de clases del análisis y del diseño así como la realización de los diagramas de clases del análisis y diseño del mismo y los diagramas de secuencia correspondientes para cada una de las operaciones del sistema propuesto. Se ha realizado una descripción detallada en forma de tablas de las clases que conforman el diagrama de clases del diseño del sistema propuesto, describiéndose cada uno de los atributos y funciones que conforman dichas clases. Teniéndose así una idea más precisa de los elementos constitutivos del sistema que se propone.

Conclusiones Generales

Con este trabajo se ha logrado el objetivo trazado, se ha realizado un estudio de los sistemas de clonación y distribución de software más usados a nivel mundial así como se ha analizado su funcionamiento y características fundamentales. Se ha completado la propuesta de un sistema capaz de satisfacer las necesidades existentes en la UCI en cuanto a la instalación de las imágenes de los sistemas operativos en las computadoras de los laboratorios docentes. Se ha alcanzado además la etapa de análisis y diseño de un sistema capaz de automatizar el proceso de clonación y distribución de imágenes existente en la UCI, reduciendo en gran medida el tiempo y la cantidad de personal requerido para la realización del proceso. Se han elaborado una serie de diagramas y tablas que facilitan al lector el entendimiento del sistema propuesto.

Recomendaciones

Teniendo en cuenta que el sistema que se propone en el presente trabajo se ha diseñado para ser usado en la UCI, se recomienda;

- Al grupo de informatización de la UCI desarrollar el sistema teniendo en cuenta los ciclos de desarrollo propuestos.
- Al grupo de administración y soporte de red de la UCI analizar otros usos y facilidades que puede brindar a la UCI el sistema propuesto.

Referencias Bibliográficas

[SCDS] - ROSELLÓ, V. J. A. Clustering de Alta Disponibilidad bajo GNU/Linux, 2001.

[ANJ] - Pfenning T. Anjuta, 2007: http://live.gnome.org/Anjuta .

[FDK] - Lissot A. Linux Partition, 2007: http://tldp.org/HOWTO/Partition/index.html

[LTSP00] - Morales A, Torres J, Quesada A. Linux Terminal Server Project (LTSP), 2006: http://wiki.ltsp.org/twiki/bin/view/Ltsp/Documentation .

[NCS00] - Padala P. NCURSES Programming, 2005: http://www.tldp.org/HOWTO/NCURSES-Programming-HOWTO/

[BSH00] - Rodríguez Alberich G. Programación en BASH, 2000: http://es.tldp.org/COMO-INSFLUG/COMOs/Bash-Prog-Intro-COMO/Bash-Prog-Intro-COMO.html

[C00] - Correa Casablanca A. Programación en C, 2006: http://ar.geocities.com/zonadelprogramador/lenguaje-c.htm .

[CPLUS] - Bjarne Stroustrup,El lenguaje de programación C++, Madrid, 1998: http://www.research.att.com/~bs/3rd.html .

[PRL00] - Merelo Guervós .J .J. Perl, 2007: http://www.perl.com/ .

[UML00] – Pilone D; Pitman N. UML 2.0 in a Nutshell, 2005:

[DSS00] - Álvarez, S; Hernández A. Metodología para el desarrollo de aplicaciones con tecnología Orientada a Objetos utilizando notación UML. La Habana, 2000.

Glosario de Términos

Cluster: Un cluster es un grupo de equipos independientes que ejecutan una serie de aplicaciones de forma conjunta y aparecen ante clientes y aplicaciones como un solo sistema.

Power_On_Self_Test: Un test automático después del encendido de la computadora que comprueba que todo esté conectado correctamente y que no haya ningún problema en los dispositivos.

EPROM: Es un tipo de chip de memoria ROM que retiene los datos cuando la fuente de energía se apaga.

BroadCast: Es un modo de transmisión de información por difusión donde la computadora emisora trasmite los paquetes simultáneamente para todos los receptores.

MAC: Es un identificador hexadecimal de 48 bits que se corresponde de forma única con una tarjeta o interfaz de red.

Kernel: Núcleo. Parte esencial de un sistema operativo que provee los servicios más básicos del sistema. Se encarga de gestionar los recursos como el acceso seguro al hardware de la computadora.
Se encarga también del multiplexado, determinando qué programa accederá a un determinado hardware si dos o más quieren usarlo al mismo tiempo.

Gateway: Dispositivo dedicado a intercomunicar sistemas con protocolos incompatibles. Puerta de enlace, acceso, pasarela. Nodo en una red informática que sirve de punto de acceso a otra red.

PXE: Pre-Boot Execution Enviornment es una herramienta que permite a una computadora bootear de un servidor en una red antes de bootear el sistema operativo del propio disco.

Bootp: Protocolo de Arranque-Asignación. Es el protocolo que asigna a un ordenador una dirección de IP, Gateway y Netmask al conectarse a una subred.

RAM: Memoria de acceso aleatorio. Tipo de memoria donde la computadora guarda información para que pueda ser procesada más rápidamente. En la memoria RAM se almacena toda información que está siendo usada en el momento.

Shell: Es una interfaz que provee acceso a los servicios del sistema operativo

Instalar: Incorporar a la computadora un programa o dispositivo para ser utilizado.

Particionar: Creación de divisiones en un disco duro para aplicarles un formato lógico (sistema de archivos).

Partición: División lógica en un disco duro.

API: Interfaz de Aplicación del Programa. Grupo de rutinas del sistema operativo o de una aplicación que definen cómo invocar cualquier servicio desde un programa.

Dirección IP: Dirección de un ordenador dentro de una red con protocolo TCP/IP.

Imagen: Grupo de configuraciones especificas que se extraen de una PC para distribuirlas en un en un grupo de PC.

Anexo 1 Diagramas de secuencia del sistema.

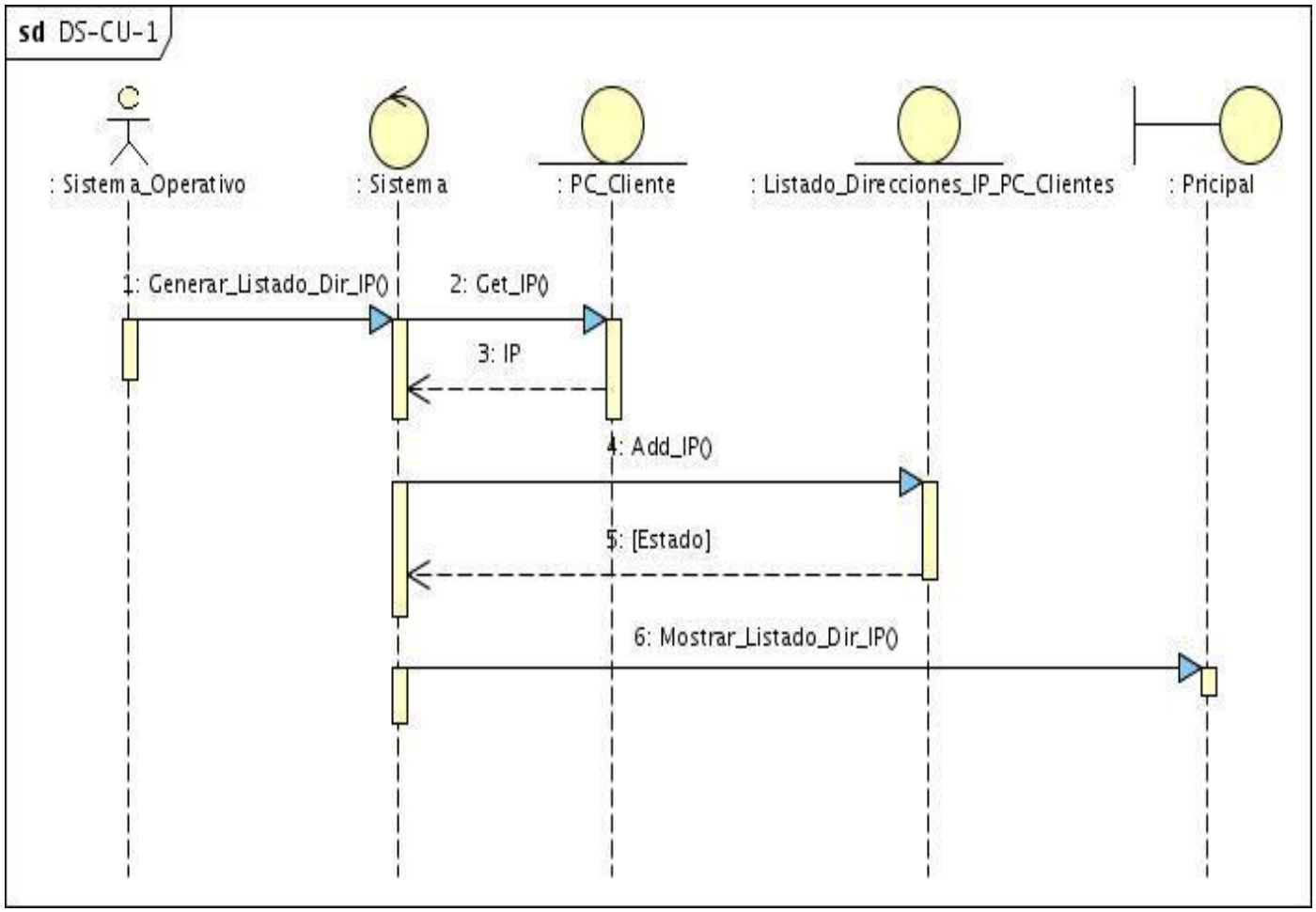

Figura A1.1 Diagrama de secuencia del CU - Generar listado de IP.

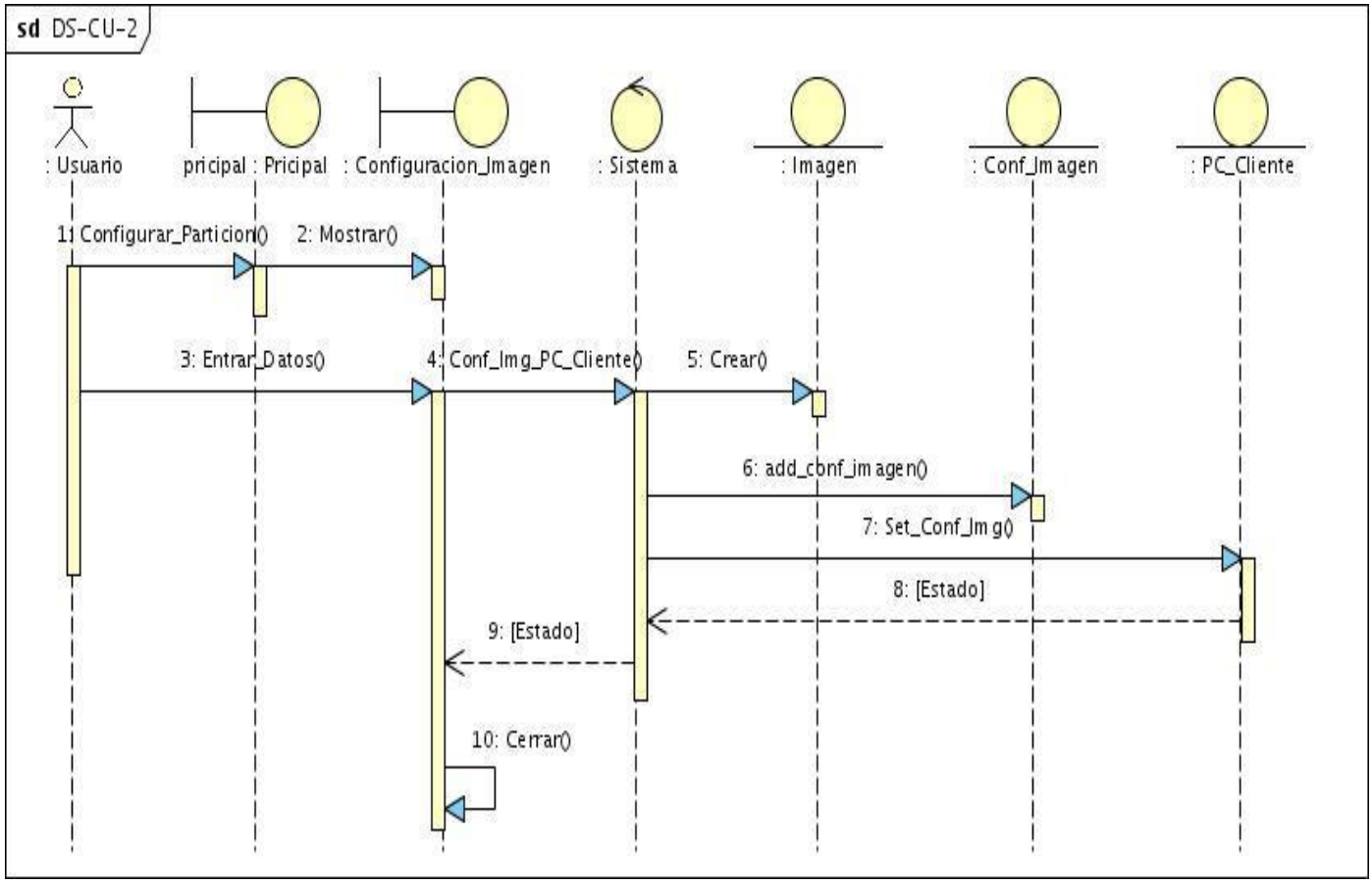

Figura A1.2 Diagrama de secuencia del CU - Configurar instalación de imagen.

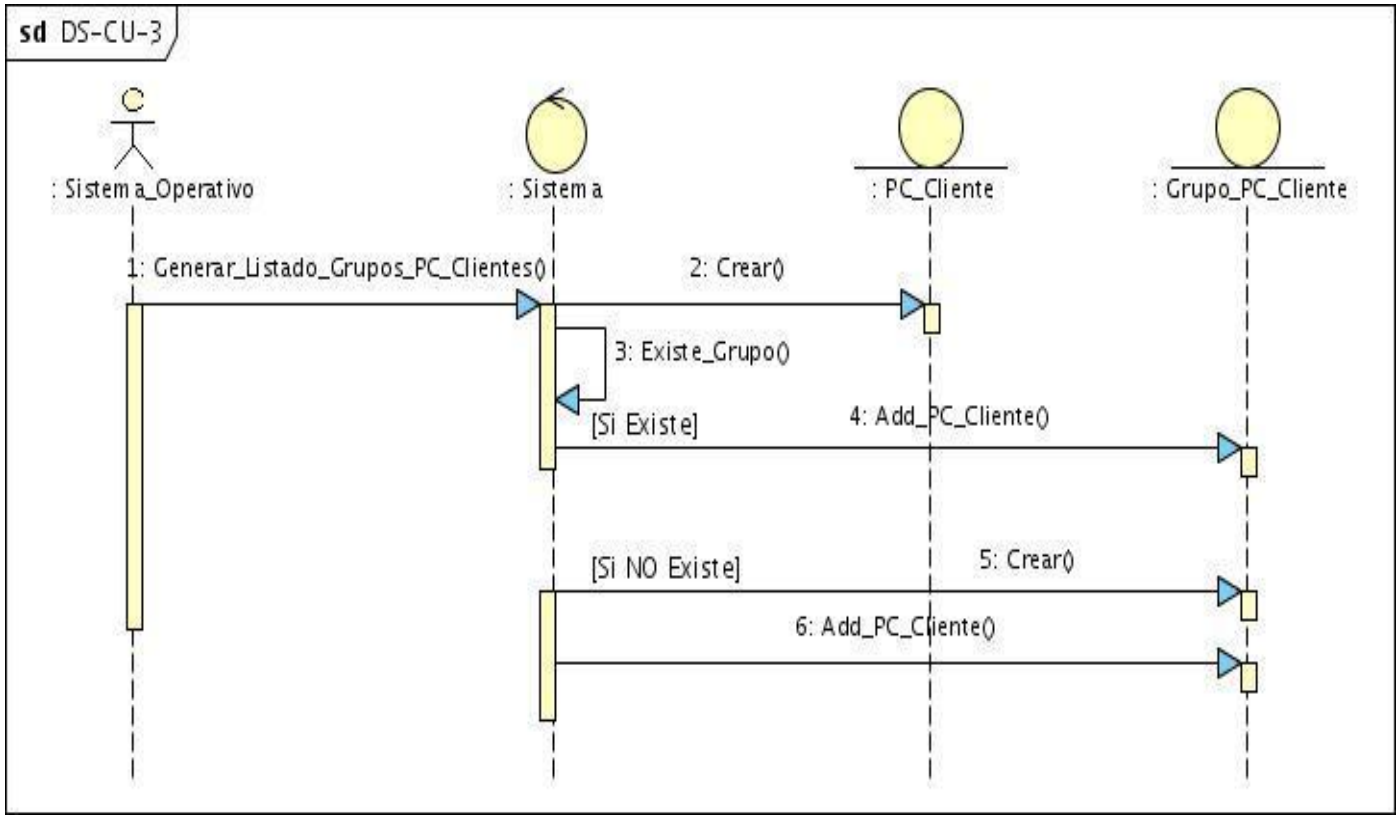

Figura A1.3 Diagrama de secuencia del CU - Gestionar información del PC_Cliente.

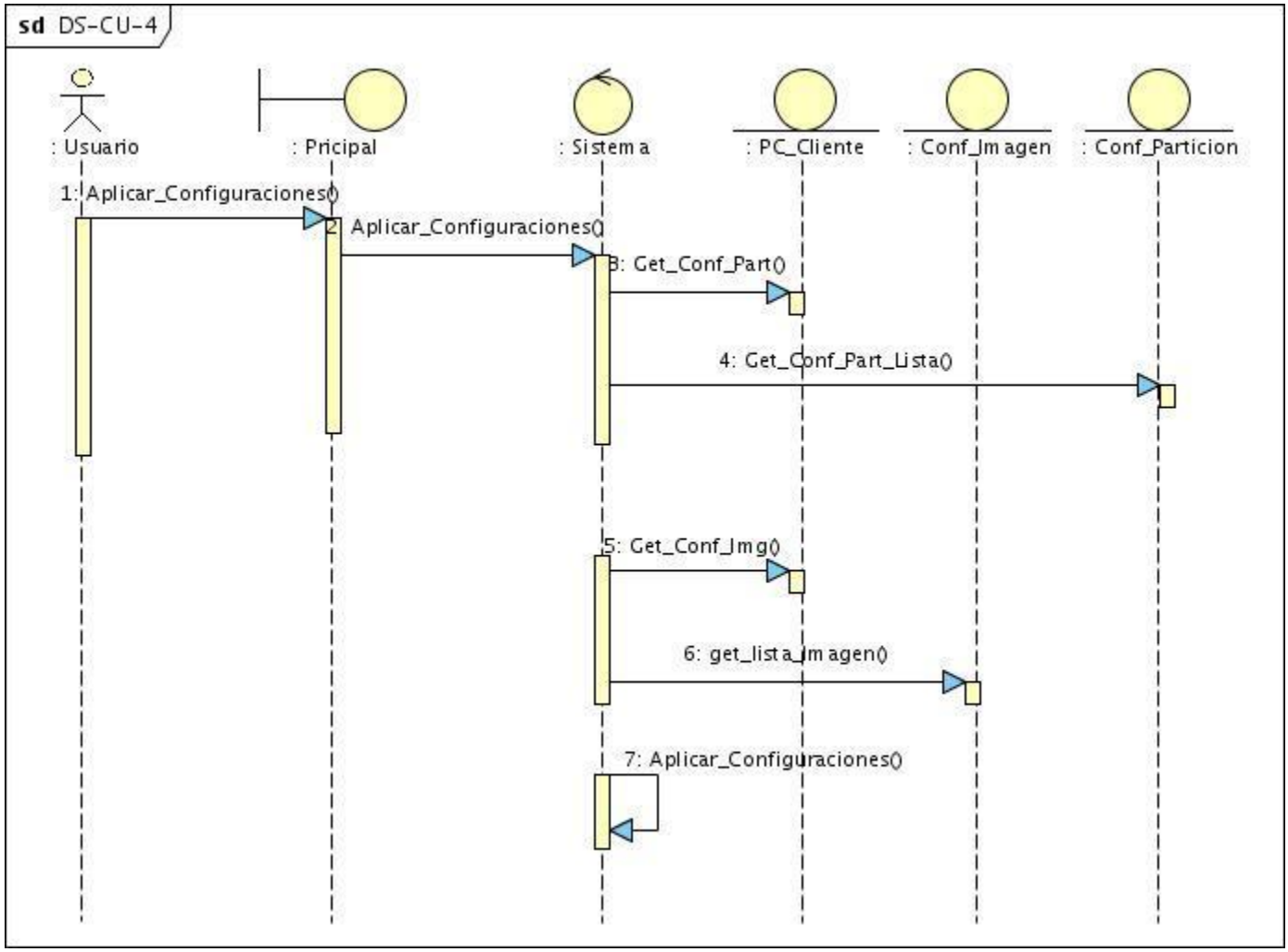

Figura A1.4 Diagrama de secuencia del CU - Aplicar Configuraciones.

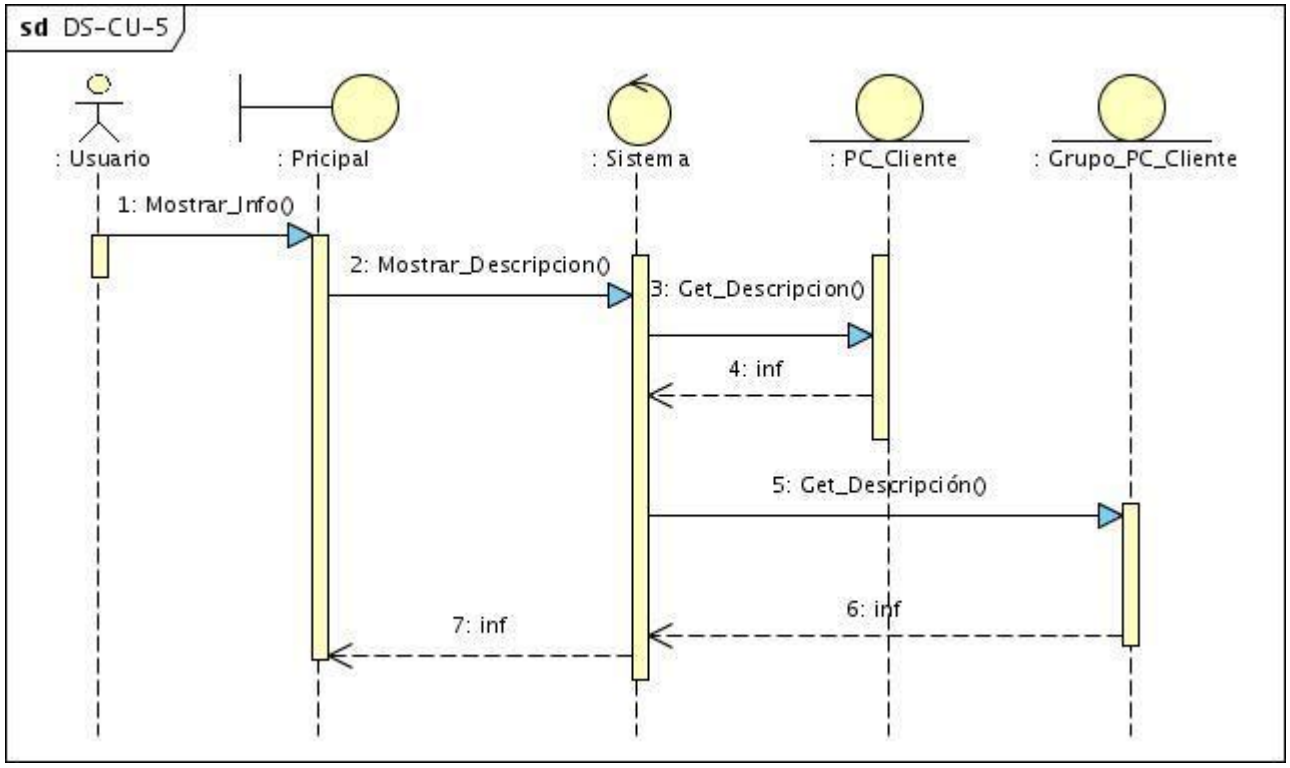

Figura A1.5 Diagrama de secuencia del CU - Mostrar información del PC_Cliente.

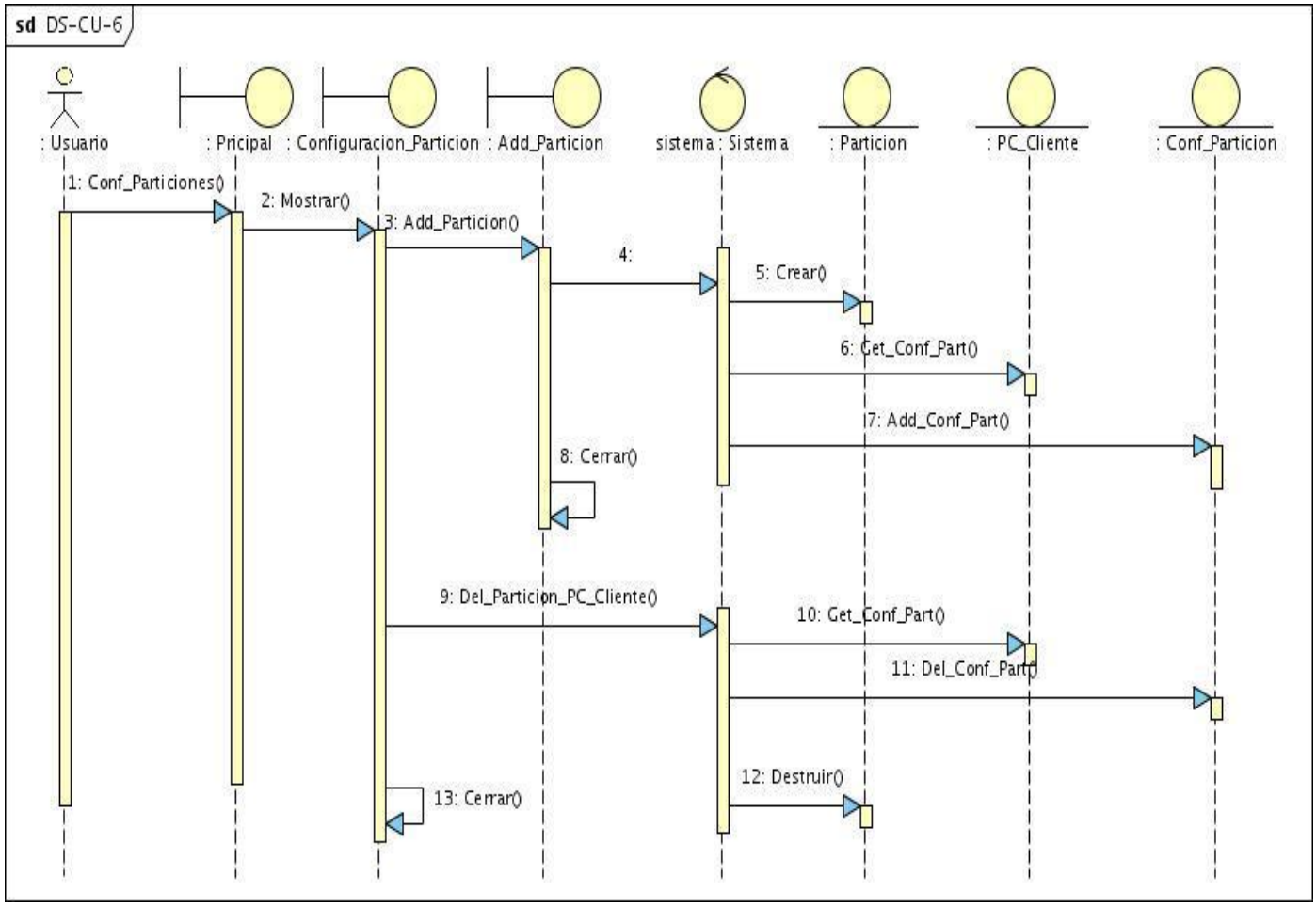

Figura A1.6 Diagrama de Secuencia del CU - Configurar Particionamiento.

Anexo 2 Diagramas de clases del diseño.

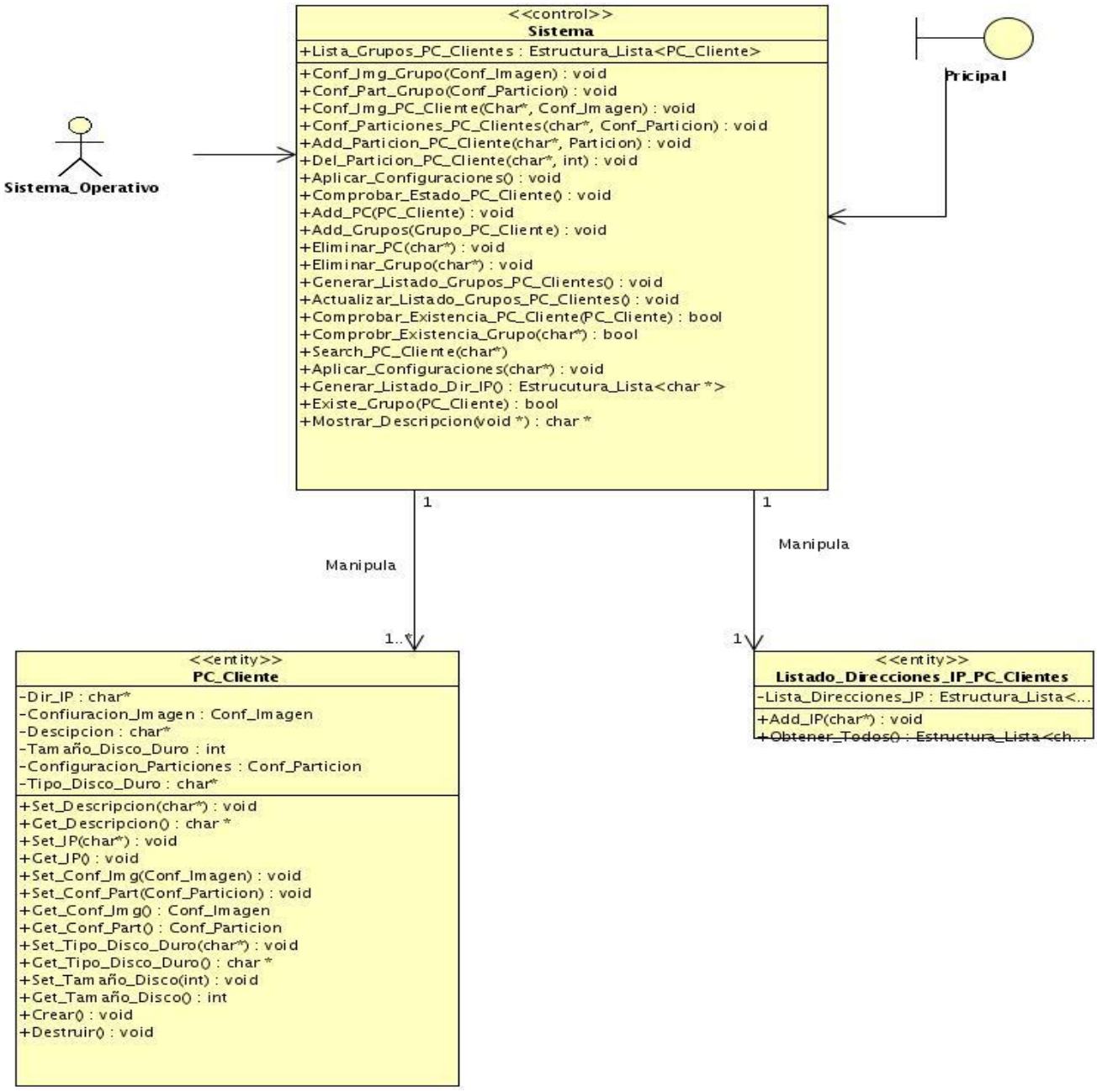

Figura A2.1 Diagrama de clases del CU - Generar listado de IP.

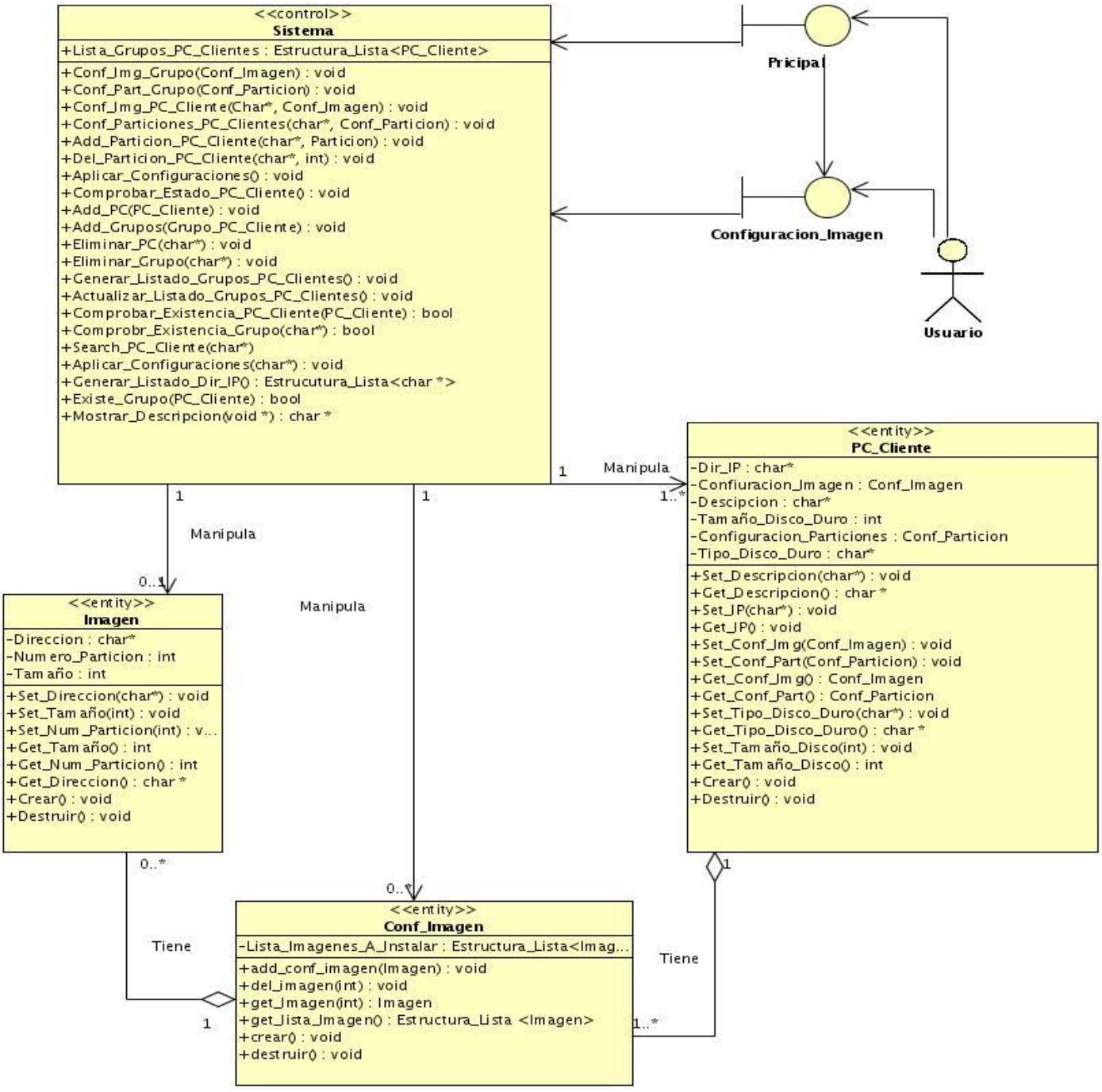

Figura A2.2 Diagrama de clases del CU - Configurar instalación de imagen.

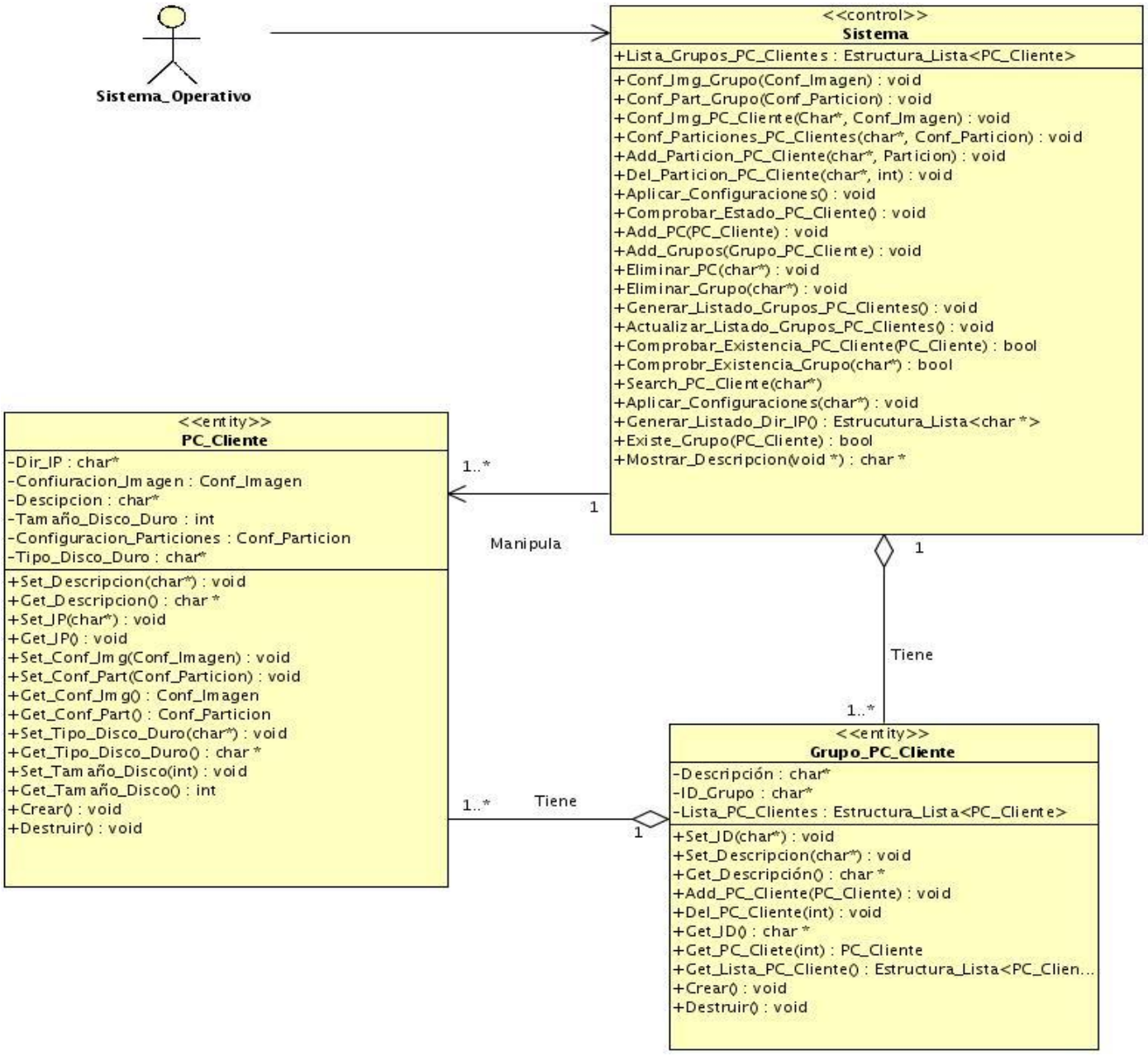

Figura A2.3 Diagrama de clases del CU - Gestionar información del PC_Cliente.

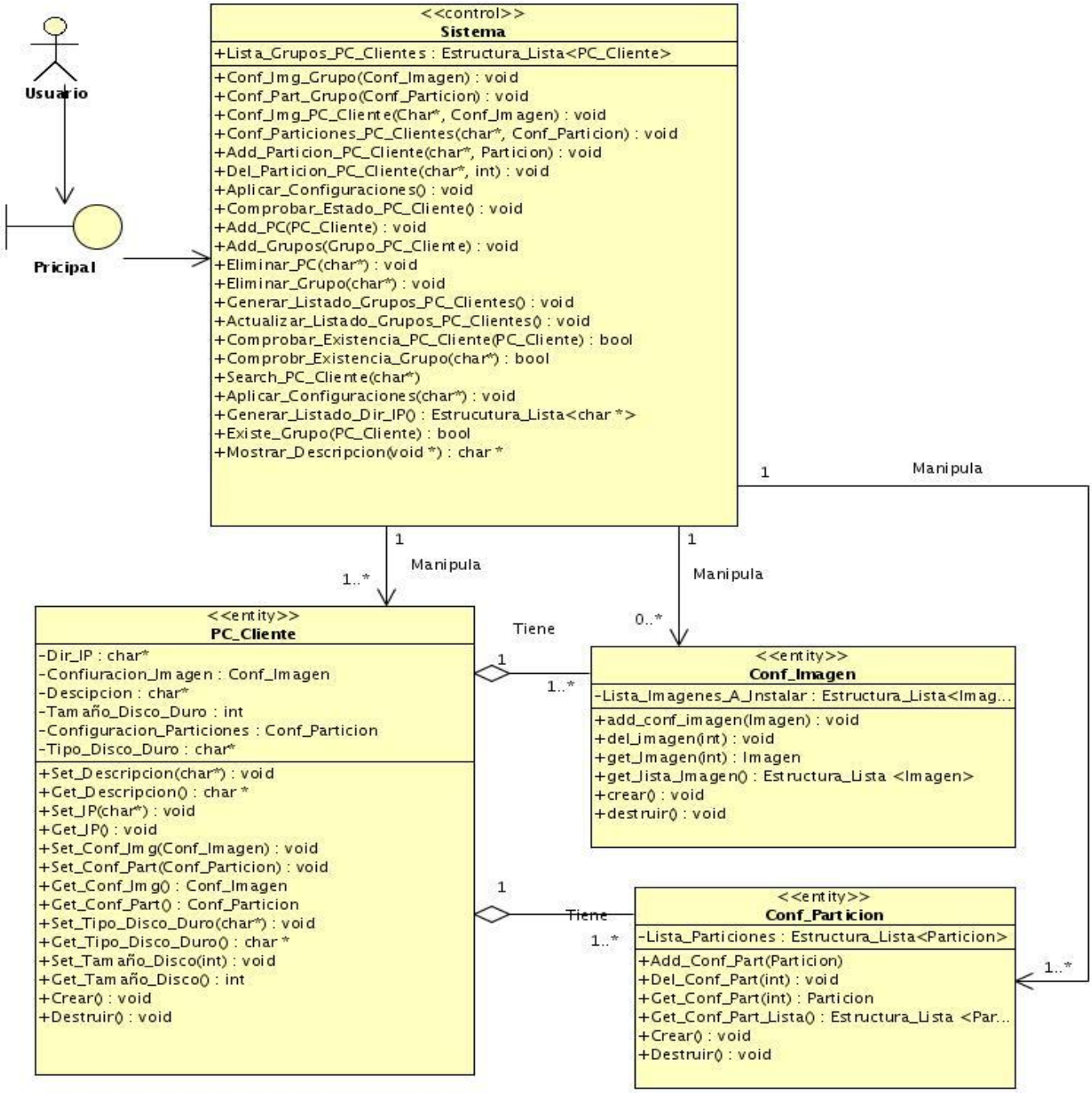

Figura A2.4 Diagrama de clases del CU - Aplicar configuraciones.

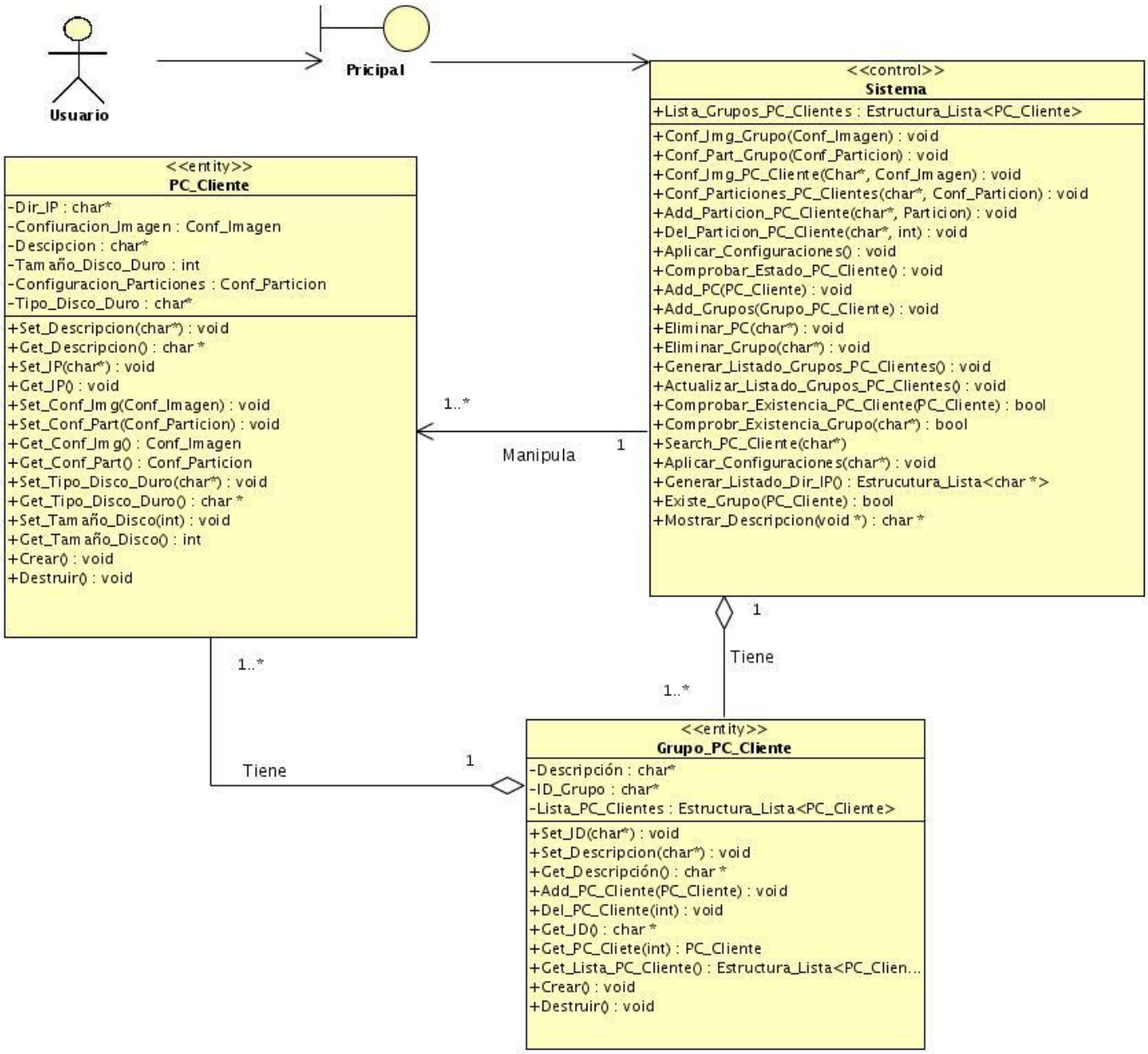

Figura A2.5 Diagrama de clases del CU - Mostrar información del PC_Cliente

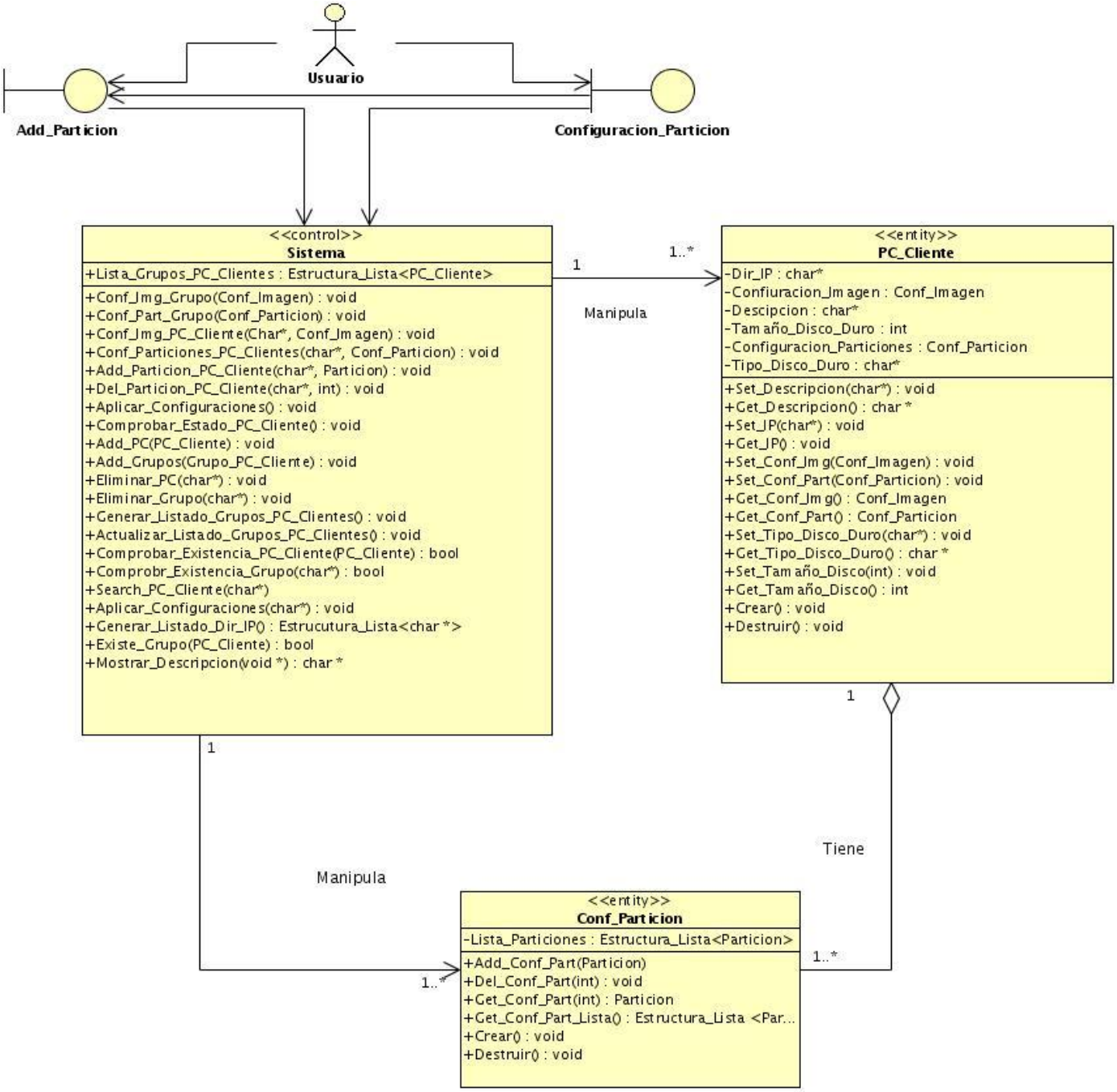

Figura A2.6 Diagrama de clases del CU - Configurar particionamiento.