Universidad de las Ciencias Informáticas

Facultad 2

Título: Plataforma de Gestión de Servicios Telemáticos en GNU\Linux.

Módulo Virtualización 1.0

Trabajo de Diploma para optar por el título de

Ingeniero Informático

Autor(es): José Carlos González Expósito

Tutor(es): Ing. Ramón Alexander Anglada Martínez

Co-tutor: Ing. Álvaro Luis Padilla Moya

"[insertar mes y año de la defensa]"

Pensamiento

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al <nombre área> de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

José Carlos González Expósito

_____

Ramón Alexander Anglada

_____

DATOS DE CONTACTO

"[insertar breve curriculum e información de contacto del tutor]"
"[insertar breve curriculum e información de contacto del asesor]"
"[insertar breve curriculum e información de contacto del consultante]"

AGRADECIMIENTOS

"[Insertar agradecimientos (opcional)]"

DEDICATORIA

"[Insertar dedicatoria (opcional)]"

RESUMEN

El mundo del software cambia constantemente, apareciendo nuevas tecnologías, entre ellas la virtualización que, gracias al aumento de las potencialidades del hardware se ha desarrollado en estos últimos años apareciendo una gran cantidad de aplicaciones para su explotación.

En una red los servicios telemáticos son imprescindibles, al virtualizar estos servicios de red, se obtienen numerosas ventajas como la disminución del tiempo de parada del servidor que contiene los servicios y el aumento de la rapidez con que se incorporan nuevos recursos a un servidor que se encuentre sobrecargado y el más importante el uso más eficiente del hardware de los servidores.

El objetivo principal de este trabajo es desarrollar una aplicación que virtualize los recursos de hardware y sistemas operativos de forma distribuida y remota, que brinde una gestión y administración de los host y máquinas virtuales de forma fácil e intuitiva, permitiendo una administración centralizada de los servidores virtualizados que contienen los servicios LDAP, DNS y DHCP. Se desea incorporar el Módulo Virtualización v1.0 a la plataforma de Gestión de Servicios Telemáticos en GNU/Linux desarrollada en el Centro Telemática de la Universidad de las Ciencias Informáticas.

PALABRAS CLAVE

Hypervisor, Virtualización, Máquina Virtual

TABLA DE CONTENIDOS

INTRODUCCIÓN

A lo largo de la historia, el hombre siempre ha tratado mejorar todo lo que con él se relaciona, como las tecnologías informáticas, consiguiendo en un tiempo relativamente corto desarrollarla hasta capacidades que nunca imaginó. Actualmente las computadoras tienen un gran poder de cómputo debido al incremento de las capacidades del hardware trayendo consigo que muchas veces este no se use al máximo, surgiendo diversas soluciones para mitigar esa deficiencia. Una de las tecnologías de la actualidad que lo consigue es la virtualización, la cual ha ido en aumento durante los últimos años consiguiendo brindar múltiples facilidades, logrando que empresas como Microsoft y VMware dediquen grandes inversiones en desarrollar tecnologías e infraestructuras para virtualizar con altos índices de utilización de los recursos, por lo cual han sido adoptadas en múltiples empresas y centros.

En Cuba desde hace varios años se está impulsando el uso del software libre especialmente en la UCI. Su uso tiene como ventaja el ahorro al país de grandes sumas de dinero en licencias de software, los productos realizados con el mismo, tienen menor costo de producción y no poseen restricciones para su comercialización.

En el proyecto Servicios Telemáticos que pertenece al centro de Telemática de la UCI se encuentra en desarrollo la Plataforma de Gestión de Servicios Telemáticos en GNU/Linux para la gestión de los servicios básicos de una red de ordenadores, contribuyendo a la migración de los mismos. Esta plataforma no ofrece la posibilidad de virtualizar los servicios que gestiona es decir DHCP, LDAP y DNS.

Es frecuente encontrar hoy en día que en cada servidor se encuentre uno de estos servicios, no utilizándose así eficientemente los recursos de hardware que estos poseen, ocupando de esa forma mayor cantidad de espacio físico en los nodos, aumentando la cantidad de recursos requeridos para los servicios, incrementando el consumo energético y dificultando la gestión de los mismos. Otras de las dificultades se encuentra en el tiempo de espera para restablecer un servicio ante la ocurrencia de un fallo en el servidor y la necesidad de adición de nuevos recursos al servidor haciéndose necesario prescindir del servicio que brinda por este tiempo afectándose a los usuarios que lo consumen, debido a esto es imprescindible ofrecer un producto que brinde solución a estas dificultades.

Teniendo en cuenta la problemática existente se plantea el siguiente **problema a resolver** ¿Cómo facilitar la virtualización distribuida de los servicios de directorio, DNS, DHCP en la Plataforma de Gestión de Servicios Telemáticos en GNU/Linux? Con motivo resolver el problema expuesto, el **objeto de estudio** es la Virtualización distribuida de los servicios. El **campo de acción** la virtualización distribuida de los servicios telemáticos directorio, DNS, DHCP de red en GNU/Linux. Se especifica como **objetivo general** desarrollar la versión 1.0 del Módulo Virtualización para la plataforma de Gestión de Servicios Telemáticos en GNU/Linux.

Para el cumplimiento del objetivo general se trazaron las siguientes tareas de la investigación:

- Realizar estado del arte sobre las tecnologías de virtualización.
- Incorporar al framework del proyecto Servicios Telemáticos los componentes y librerías de virtualización, basadas en las ya existentes.
- Investigar cómo virtualizar de manera local los recursos: Memoria RAM, dispositivos de almacenamiento, procesador y tarjeta de red.
- Investigar cómo virtualizar de manera distribuida y remota los recursos: Memoria RAM, dispositivos de almacenamiento, procesador y tarjeta de red.
- Investigar cómo virtualizar los siguientes sistemas operativos GNU/Linux: Debian, Ubuntu.
- Investigar cómo realizar operaciones de gestión (adición, inserción, edición y eliminación) sobre los recursos de hardware y sistemas operativos a virtualizar.
- Incorporar a la plataforma de Gestión de Servicios Telemáticos un módulo que permita:
    - Virtualizar los recursos de hardware y sistemas operativos de forma remota y distribuida para la virtualización de los servicios LDAP, DNS y DHCP.

Cumpliendo las tareas correctamente se espera obtener como resultado el Módulo Virtualización v1.0 para la plataforma de Gestión de Servicios Telemáticos en GNU/Linux desarrollada en el proyecto Servicios Telemáticos.

Para dar cumplimiento a las tareas investigativas se van a utilizar Métodos de Investigación Científica. Los Métodos de Investigación Científica son la forma de abordar la realidad, de estudiar la naturaleza, la sociedad y el pensamiento, con el propósito de descubrir su esencia y sus relaciones. Los métodos seleccionados son los teóricos y empíricos.

Métodos teóricos

- Método Analítico – Sintético.
- Método Inductivo – deductivo.

Se utilizaron para estudiar las características de la virtualización, se analizaron los componentes del concepto de virtualización, como los tipos de virtualización completa y para-virtualización y se realizó un estudio sobre las tecnologías actuales de virtualización como hypervisores, herramienta y librerías.

Métodos empíricos

- Método Observación.
- Método Experimento.

Estos dos métodos permitieron observar y probar cómo funcionaban las diferentes soluciones existentes, como el hypervisor KVM y la API Libvirt, obteniendo información de su funcionamiento. Durante el experimentando con estas dos herramientas de virtualización fue posible comprobar directamente como se creaba y gestionaba una máquina virtual de forma local y remota, eligiendo sin intermediarios la más adecuada para el entorno y situación presentada.

La organización del presente documento es la siguiente:

Capítulo 1:"Fundamentación teórica", contiene el resultado del estudio sobre el estado del arte de las soluciones que existen en el mundo de la problemática abordada. Se definen técnicas, metodologías y herramientas a utilizar, su caracterización y justificación de uso.

Capítulo 2, *"Características del Sistema",* se define una propuesta del sistema, especificándose los requisitos de la herramienta, los cuales ayudan a desarrollar los casos de uso basándose en la metodología.

Capítulo 3, *"Análisis y Diseño del Sistema",* se define cómo continúa el desarrollo del sistema llevándose a cabo el diseño del mismo.

Capítulo 4, *"Implementación y Prueba",* se llevan a cabo el desarrollo del diagrama de despliegue y los diagramas de componentes. Se hacen las pruebas a la aplicación.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

En el presente capítulo se aborda el actual camino de la virtualización en el mundo y sus conceptos fundamentales, centrándose en las soluciones que se ajusten más a los problemas existentes y las necesidades del proyecto. Se exponen las tecnologías y metodologías que se utilizarán para el desarrollo de la aplicación de gestión de servicios telemáticos en GNU/Linux Módulo Virtualización.

1.2 Virtualización y Software de virtualización

En este apartado se abordará la historia de la virtualización y sus ventajas, también las soluciones existentes en el mundo sobre software de virtualización, viéndolos de forma crítica, adquiriendo experiencia clave para un mejor desarrollo del producto.

Durante la década de los 60 los equipos de informática de muchas empresas y entidades tenían un problema similar: contaban con súper-computadoras o “mainframes” de alto rendimiento que deseaban “particionar lógicamente”, o utilizar para múltiples tareas simultáneas (lo que hoy conocemos como “multitasking”, trabajar más de una aplicación o proceso simultáneamente). Es por esto que IBM desarrolló un método para crear múltiples “particiones lógicas” (similar a lo que conocemos hoy como “máquinas virtuales”) las cuales trabajaban independientemente una de las otras, y cada una utilizando los recursos provistos por el “mainframe”. Ya para la década de los 80 y con la llegada de las máquinas x86, comenzó una nueva era de micro computadoras, aplicaciones cliente-servidor, y “computación distribuida”; en donde los enormes y potentes “mainframes” con muchas tareas y utilidades en una sola caja gigantesca se comenzaron a cambiar por relativamente pequeños servidores y computadoras personales de arquitectura x86 lo que se convirtió rápidamente en el estándar de la industria. Debido a esto, una vez más, el tema de la virtualización vuelve a quedar prácticamente en el olvido y no es hasta finales de la década de los 90 que gracias al alto desarrollo del hardware vuelve a alcanzar auge : el hardware existente es altamente eficiente, y utilizar cada servidor x86 o escritorio para una sola aplicación sería un desperdicio de recursos, espacio, energía y dinero; y tampoco es conveniente asignarle múltiples usos o instalar varias aplicaciones en un solo servidor convencional, por más de una razón (ej. estas aplicaciones podrían ser conflictivas entre sí, o podrían requerir diferentes configuraciones e inclusive diferentes sistemas operativos, o tener diferentes requerimientos de seguridad, entre otras variables que podrían causar problemas al ejecutar estas funciones simultáneamente). Es por esto que vuelve a resurgir la idea de dividir el hardware, de manera tal que funcione como múltiples servidores independientes pero compartiendo los recursos de un mismo servidor físico. (1) Así en 1999, VMware introdujo la virtualización en los sistemas x86 para transformar estos sistemas en infraestructuras de hardware compartido de uso general que ofrecen un aislamiento completo, movilidad y opciones de elección del sistema operativo en los entornos de aplicaciones. A diferencia de los mainframes, las máquinas x86 no fueron

diseñadas para admitir una virtualización completa, por lo que VMware tuvo que superar muchos desafíos para crear máquinas virtuales a partir de ordenadores x86.

La función básica de la mayoría de las CPU, tanto en mainframes como en PC, es ejecutar una secuencia de instrucciones almacenadas (es decir, un programa de software). En los procesadores x86, hay 17 instrucciones específicas que generan problemas al virtualizar, y provocan que el sistema operativo muestre un aviso, que se cierre la aplicación o simplemente que falle completamente. Como consecuencia, estas 17 instrucciones constituyeron un obstáculo importante para la implementación inicial de la virtualización de ordenadores x86. Para hacer frente a las instrucciones problemáticas de una arquitectura x86, VMware desarrolló una técnica de virtualización adaptable que las “atrapa” cuando se generan y las convierte en instrucciones seguras que se pueden virtualizar y, al mismo tiempo, que permite al resto de instrucciones ejecutarse sin intervención. El resultado es una máquina virtual de alto rendimiento que se adapta al hardware host y mantiene una total compatibilidad de software. VMware fue pionero en esta técnica, actualmente ofrece varios productos ejemplo de ellos son VMware Workstation y VMware Player, actualmente VMware es el líder indiscutido de la tecnología de la virtualización (2). Aparecen también otras soluciones como Windows Server 2008 R2, donde se utilizan los siguientes tipos de virtualización: virtualización de clientes y servidores desde Hyper-V (Hypervisor basado en sistemas virtualizados) y virtualización de la Presentación con Servicios de Escritorio Remoto (Remote Desktop Services, RDS). Hyper-V virtualiza los recursos de un sistema físico para poder disponer de un entorno virtualizado desde el cual ejecuta sistemas operativos y aplicaciones. Los Servicios de Escritorio Remoto virtualizan un entorno de procesamiento y aíslan la lógica del programa de la parte gráfica y los procesos de manera que se puede ejecutar un aplicación de un lugar y controlarla en otra. La virtualización de la presentación permite ejecutar desde una única aplicación a un conjunto completo de aplicaciones para desktop a través de la red. Es decir Windows Server 2008 permite a los usuarios acceder a programas Windows instalados en un servidor de terminales o acceder a un desktop Windows completo (3). Sin embargo su licencia tiene un alto coste y al ser privativo posee restricciones para su utilización.

Otra de las nuevas tendencias de la virtualización es la computación en nube (del inglés "Cloud computing"), es un paradigma que permite ofrecer varios servicios a través de Internet aumentando así el número de servicios basados en la red.

Existen dos formas fundamentales de virtualización, la virtualización completa y la para-virtualización. En el modo completa, existe una capa entre el sistema operativo virtualizado y el hardware que es la encargada de manejar íntegramente el acceso a los recursos, se le conoce con el nombre de hypervisor y sirve como capa de abstracción entre la plataforma de hardware y los sistemas operativos virtualizados. Es importante señalar que, en algunos casos, el hypervisor puede ser un sistema operativo y en tal caso se denomina host operating system. La solución comercial VMware constituye un ejemplo de este tipo de virtualización.

En el modo para-virtualizado el hypervisor o como también se le denomina, virtual machine monitor (Monitor de Máquina Virtual) opera de manera cooperativa con los sistemas operativos virtualizados. Mediante habilidades de programación, los sistemas operativos pueden “conocer” que están siendo virtualizados y entonces cooperar con el hypervisor en el acceso y manejo de los recursos. Esta técnica de virtualización es la empleada por parte del hypervisor Xen. (4)

La virtualización presenta las siguientes ventajas:

- Rápida incorporación de nuevos recursos para los servidores virtualizados.
- Administración global centralizada y simplificada.
- Aislamiento: un fallo general de sistema de una máquina virtual no afecta al resto de máquinas virtuales.
- Aporta el beneficio en la reducción del hardware necesario, así como de sus costes asociados.
- Reduce los tiempos de parada de los servidores y servicios virtualizados.
- Migración en sin pérdida de servicio de máquinas virtuales de un servidor físico a otro, eliminando la necesidad de paradas planificadas por mantenimiento de los servidores físicos. (5)

Windows Server 2008

Incluye Windows Server Virtualization (WSv), una tecnología eficaz de virtualización con sólidas características de administración y seguridad. Algunas de las características que ayudaron a mejorar la seguridad se debe a la creación de particiones fuertes: una máquina virtual (VM) funciona como un contenedor independiente de sistema operativo, completamente aislado de otras máquinas virtuales que se ejecutan en el mismo servidor físico. Windows Server Virtualization ayuda a evitar la exposición de las VM que contienen información confidencial y protege también al sistema operativo host subyacente del riesgo que significa un sistema operativo invitado. El rendimiento de las máquinas virtuales es mejorado gracias a la compatibilidad con múltiples núcleos donde a cada VM se le pueden asignar hasta ocho procesadores lógicos: esto permite la virtualización de grandes cargas de trabajo, con cálculo intensivo, que aprovechan los beneficios del procesamiento en paralelo de núcleos de VM con procesadores múltiples. (3)

Windows Server 2008 Hyper-V

Windows Server 2008 Hyper-V es la funcionalidad de virtualización basada en el hypervisor, incluida como un rol de servidor específico de Windows Server 2008. Hyper-V posee una capacidad de plataforma dinámica, fiable y escalable combinada con un conjunto exclusivo de herramientas de gestión que permiten administrar tanto los recursos físicos como los virtuales, lo que facilita la creación de un datacenter ágil y dinámico y el avance hacia un modelo de sistemas dinámicos auto gestionados. Hyper-V permite desarrollar cuatro

escenarios básicos: consolidación de servidores, continuidad de negocio, entornos de prueba y desarrollo y el datacenter dinámico. (6)

Aunque estos productos de Microsoft tengan grandes ventajas, sus precios son altos, siendo imposible para un país con la economía de Cuba costearlos, también son de código cerrado no pudiendo realizar modificaciones para ajustarlos a las necesidades del proyecto.

VMware

VMware es un sistema de virtualización por software, virtualiza los recursos de hardware de un equipo basado en x86, incluyendo la CPU, RAM, disco duro y controlador de red para crear una completa y funcional máquina virtual, en la que se pueda ejecutar su propio sistema operativo y aplicaciones. Cada máquina virtual contiene un sistema completo, eliminando los posibles conflictos. VMware funciona mediante la inserción de una delgada capa de software directamente en el hardware del ordenador o en un sistema operativo host. Este contiene un monitor de máquina virtual o "hipervisor" que asigna los recursos de hardware de forma dinámica y transparente. Al encapsular una máquina completa, incluyendo CPU, memoria, sistema operativo y los dispositivos de red, una máquina virtual es totalmente compatible con todos los sistemas operativos estándar x86, aplicaciones y controladores de dispositivos. Existen varias soluciones para virtualizar que brinda VMware como VMware Player, que es un software gratuito que permite la ejecución de máquinas virtuales creadas con otros productos de VMware, pero no permite crearlas él mismo. (8), VMware Server y VMware Workstation. (7) VMware Server soporta virtualización por emulación, virtualización nativa de forma experimental, arquitecturas de 64 bits y SMP Symmetric Multi-Processing (multiproceso simétrico).Dispone de una interfaz de administración intuitiva, soporte técnico, documentación abundante y funciona sobre Linux y Windows. (9) VMware Workstation es una versión comercial, se instala dentro de un sistema operativo anfitrión como un programa estándar, de tal forma que las máquinas virtuales se ejecutan dentro de esta aplicación. Tiene un aprovechamiento restringido de recursos. Este software no posee las características que se necesitan y es de código cerrado siendo imposible su incorporación a la plataforma.

VMwarevSphere

Es una plataforma de virtualización donde puedes construir una infraestructura virtual, se utiliza como base para la construcción de nubes privadas y públicas. Los recursos de hardware se asignan dinámicamente cuando y donde se necesitan dentro de su nube privada, por lo que no es necesario asignar servidores, almacenamiento o ancho de banda de red permanentemente a cada aplicación permitiendo que las aplicaciones de más alta prioridad siempre tengan los recursos necesarios. También da la posibilidad de conectar esta nube privada a una nube pública para crear una nube híbrida, que da como resultado la flexibilidad, la disponibilidad y la escalabilidad necesarios.

No es posible la utilización de este producto de VMware debido a que es de código cerrado siendo imposible su modificación e incorporación a la plataforma.

VirtualBox

Es un software de virtualización para arquitecturas x86 y AMD64/Intel64, siendo la única solución profesional de código abierto bajo los términos de GNU General Public License (GPL). Permite virtualizar la mayoría de los sistemas operativos. (10)Sus principales características son:

- Modularidad, lo que da la facilidad de controlar varias interfaces a la vez.
- Las descripciones de la máquina virtual están en XML, es decir los archivos de configuración pueden ser transportados fácilmente a otros equipos.
- Carpetas compartidas para el intercambio de información entre el anfitrión e invitados.
- Implementa un controlador USB virtual que permite la conexión de dispositivos USB, sin tener que instalar los controladores.

Actualmente Virtual Box 3.2.12 se puede descargar en su versión para Windows, Linux y otros sistemas operativos. Este software no es el adecuado para virtualizar debido solo sirve para virtualizar de manera local y no remota, la gestión de las máquinas virtuales se hace pesada si hay un gran número de máquinas virtuales, para su uso es necesario interfaz perdiéndose rendimiento del servidor.

Xen

Es un monitor de máquina virtual para x86 que admite la ejecución de múltiples sistemas operativos invitados con altos niveles de rendimiento y el aislamiento de los recursos. Una delgada capa de software conocida como el hipervisor Xen se inserta entre el hardware del servidor y el sistema operativo. Esto proporciona una capa de abstracción que permite a cada servidor físico ejecute uno o más "servidores virtuales" y la disociación efectiva del sistema operativo y sus aplicaciones desde el servidor físico subyacente. El hipervisor Xen es excepcionalmente delgado con menos de 150.000 líneas de código. Eso se traduce en una sobrecarga extremadamente baja y un rendimiento casi nativo para los huéspedes. Xen reutiliza los controladores de dispositivos (tanto de código cerrado y abierto) de Linux, facilitando la administración de dispositivos. Por otra parte Xen es robusto a la falta de controladores de dispositivo y protege tanto a los huéspedes y al hipervisor de controladores defectuosos o maliciosos. Con la migración de máquinas virtuales en caliente de Xen se puede conseguir hacer balance de cargas sin tiempos muertos. Xen es software de código abierto, liberado bajo los términos de la GNU General Public License. (11) Xen usa la para-virtualización que tiene un rendimiento casi nativo pero donde es imposible la virtualización de sistemas

privativos porque hay que modificar el kernel del sistema operativo invitado decidiéndose su uso en la segunda versión del software.

KVM

KVM (máquina virtual basada en el kernel) es una solución de virtualización completa para Linux en el hardware x86 con extensiones de virtualización (Intel VT o AMD-V). Consiste en un módulo cargable del núcleo, kvm.ko, que proporciona la infraestructura base de la virtualización y un módulo de procesador específico, intel.ko kvm o amd.ko kvm. Con el uso de KVM se puede ejecutar múltiples máquinas virtuales sin modificar las imágenes de Linux o Windows. (12) KVM toma lugar como módulo del kernel Linux, dicho módulo exporta un dispositivo llamado /dev/kvm que habilita un modo huésped en ingles guest en adición al resto de los modos de usuario tradicionales que soporta el kernel. En un árbol tradicional /dev los dispositivos le son comunes a todos los procesos en espacio de usuario, sin embargo en /dev/kvm a cada proceso que es lanzado se le asocia un mapeo de dispositivos diferente.

Esto garantiza el aislamiento entre las VM. Cuando el módulo KVM es instalado, el kernel Linux se convierte en un hypervisor. Un hypervisor que puede aprovechar todas las bondades de un kernel estándar de Linux. Es importante señalar que KVM no constituye el primer proyecto de virtualización a nivel del kernel Linux. User Mode Linux (UML) también hace uso de habilidades que convierten al kernel en un hypervisor.

El modelo de funcionamiento es simple y genial. Mediante KVM se pueden lanzar sistemas operativos huéspedes en espacio de usuario. Cada sistema operativo huésped constituye un proceso independiente dentro del host operating system. Más arriba pero por debajo de los sistemas operativos huéspedes existe una capa muy fina (QEMU en este caso), que constituye la interfaz para el manejo, configuración y control de las VM. (4) El componente del núcleo de KVM está incluido en la línea principal de Linux, a partir del 2.6.20. KVM es un software de código abierto. Se decidió el uso de este hypervisor debido a su creciente auge, amplia documentación y al uso de la virtualización completa, pudiendo virtualizar cualquier sistema operativo que soporte arquitectura x86, como se carga con un módulo del kernel usa todas sus funcionalidades y su rendimiento no se diferencia mucho del de Xen y también es soportado por la librería libvirt.

1.3 API y herramientas para la administración de clústeres de máquinas virtuales.

Ganeti

Es una herramienta para la administración de servidores clústeres virtualizados, en la punta de las tecnologías de virtualización existentes, como Xen o KVM.

Ganeti requiere para su funcionamiento un software de virtualización pre-instalado en sus servidores. Una vez instalado, la herramienta se encargará de la parte de gestión de las instancias virtuales, por ejemplo, la administración de creación de discos, instalación del sistema operativo, inicio, apagado, conmutación por error

entre los sistemas físicos. Se diseñó para facilitar la gestión de clústeres de servidores virtuales y para proporcionar una recuperación rápida y simple tras fallos físicos utilizando el hardware de los productos básicos.
Alguna de sus funciones para la gestión de instancias son las siguientes:
Apoyo para la virtualización de Xen:

- Apoyo a las instancias PVM (Máquina Virtual Paralela) y HVM (hardware de la máquina virtual).
- Apoyo a la migración en vivo.

Soporte para la virtualización KVM:

- Apoyo a las instancias totalmente virtualizado.
- Apoyo a las instancias semi-virtual (kernel que reside en el host)

Gestión de discos:

- Tamaño del clúster recomendado de 10-40 nodos físicos.

Ganeti tiene una licencia GNU General Public License v2 y una amplia documentación. (13) Su gestión es a través de la web y la consola por lo que su incorporación a la plataforma se hace difícil. No es un proyecto sólido lo que trae consigo la posibilidad que los desarrolladores detengan el soporte a esta herramienta, no pudiendo incorporar nuevas funcionalidades y mejoras que fueran surgiendo en la virtualización.

Virttool

Herramienta para la gestión de centros de datos utilizando tecnologías de virtualización. Está compuesto por dos módulos diferentes pero que interactúan entre sí, un demonio (Disk And Execution Monitor del español Monitor de Disco y ejecución) ligero, de alta disponibilidad y seguimiento, y una interfaz web de fácil manejo. Usa libvirt para la gestión de nodos y una base de datos, siendo capaz de gestionar todo lo que soporta libvirt. (14) . La gestión se realiza mediante una interfaz web lo que dificulta su incorporación a la plataforma, se estaría creando una aplicación de escritorio para la gestión a partir de una aplicación web que ya los gestiona programándose dos veces las y perdiéndose rendimiento. No es una herramienta que cuente con gran respaldo financiero, pudiendo ser absorbida por grandes empresas o que sus desarrolladores no brinden más soporte.

Libvirt

Libvirt es una API de virtualización capaz de interactuar con las capacidades de virtualización de una gama de sistemas operativos. Proporciona una capa común, genérica y estable para manejar con seguridad los dominios en un nodo. Como los nodos pueden estar situados remotamente, libvirt provee todas las API

necesarias para crear, modificar, supervisar, controlar y detener la migración de dominios, dentro de los límites del soporte del hipervisor para estas operaciones. Aunque se puede acceder al mismo tiempo a varios nodos con libvirt, la API se limita a las operaciones de un solo nodo. (15)

Libvirt pretende ser un bloque de construcción para un mayor nivel de herramientas de gestión y aplicaciones centradas en la virtualización de un solo nodo, con la única excepción de la migración de dominio entre las capacidades de múltiples nodos. Proporciona las API para enumerar, vigilar y utilizar los recursos disponibles en el nodo administrado, incluyendo CPU, memoria, almacenamiento, redes y la particiones Non-Uniform Memory Access (Acceso de Memoria no uniforme) (NUMA). Con libvirt se tienen dos medios de control distintos, el primero donde la aplicación de gestión y los dominios existen en el mismo nodo. En este caso, la aplicación de gestión funciona a través de libvirt para controlar los dominios locales. Los otros medios de control existen cuando la aplicación de gestión y los dominios son nodos individuales. En este caso, se requiere comunicación remota. Este modo utiliza un demonio especial llamado libvirtd que se ejecuta en nodos remotos. Este demonio se inicia automáticamente cuando libvirt se instala en un nodo nuevo y puede determinar automáticamente los hypervisores locales y establecer drivers para ellos .La aplicación de gestión se comunica a través de la libvirt local con la libvirtd remota a través de un protocolo personalizado. (16)

OpenNebula

OpenNebula es un conjunto de herramientas de código abierto para la creación de nubes privadas, públicas e híbridos, que ofrecen características para la gestión de las nubes. OpenNebula es el resultado de muchos años de investigación y desarrollo de la gestión eficiente y escalable de máquinas virtuales en infraestructuras distribuidas de gran escala. (17) Su gestión es a través de la consola y se realiza fácilmente, permitiendo a través de comandos la gestión de clúster, imágenes y máquinas virtuales.

OpenNebula consta con las funcionalidades que se requiere, se le hicieron pruebas para montar el sistema y ver su funcionamiento, pero no se logró que funcionara correctamente, por lo que no fue posible su uso.

Se decidió la utilización de la API libvirt debido a que esta posee un implementación en Python la cual es de fácil utilización, posee las funciones necesarias para la virtualización de forma remota que se requiere y sería de fácil integración para la Plataforma, soporta hypervisores como XEN, KVM, VirtualBox y VMware ESX, dando flexibilidad a la hora de escoger el hypervisor a utilizar. Maneja la seguridad de la conexión al hypervisor y cuenta con un amplio soporte y documentación, siendo usada como base de construcción de diferentes herramientas de virtualización.

1.4 Metodologías de desarrollo

En la producción de software se espera que el cliente obtenga un producto a tiempo, acorde a sus necesidades, para lograrlo el proceso de desarrollo debe ser eficaz y eficiente, esa es la finalidad de las

metodologías de desarrollo. Una de las metodologías más conocidas es RUP Rational Unified Process (Proceso Racional Unificado).

RUP

RUP es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.
Principales elementos:
Trabajadores (“quién”)
Actividades (“cómo”)
Artefactos (”qué”)
Flujo de actividades (“Cuándo”) (18) (19)

Las características de RUP son las siguientes:

- **Dirigido por casos de uso**: Los casos de uso reflejan lo que los usuarios futuros necesitan y desean, lo cual se capta cuando se modela el negocio y se representa a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso.

- **Centrado en la arquitectura**: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción. RUP se desarrolla mediante iteraciones, comenzando por los CU relevantes desde el punto de vista de la arquitectura.

- **Iterativo e Incremental:** RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros.

Las fases de RUP son:

- **Conceptualización (Inicio):** Se describe el negocio y se delimita el alcance del proyecto, describiendo sus alcances con la identificación de los casos de uso del sistema.

- **Elaboración:** Se seleccionan los casos de uso que definan la arquitectura del sistema.

- **Construcción:** Se obtiene un producto listo para su utilización que está documentado. Se obtienen uno o varios *release* del producto que han pasado las pruebas. Se ponen estos *release* a consideración de un subconjunto de usuarios.

  Transición: El release ya está listo para su instalación en las condiciones reales. Puede implicar reparación de errores. (18) (19)

Modelación de funciones .IDEF0

La traducción literal de las siglas IDEF es Integration Definition for Function Modeling (Definición de la integración para la modelización de las funciones). IDEF consiste en una serie de normas que definen la metodología para la representación de funciones modeladas.

IDEF consiste en una serie de normas que definen la metodología para la representación de funciones modeladas. Representa de manera estructurada y jerárquica las actividades que conforman un sistema o empresa y los objetos o datos que soportan la interacción de esas actividades. (18) (19)

Los conceptos de IDEF0 diseñados para mejorar la comunicación son los siguientes:

- Diagramas basados en cajas simples y flecha gráficas.
- Etiquetas de texto en inglés para describir cuadros y flechas, un glosario y un texto para definir el significado exacto de los elementos del diagrama.
- La exposición gradual de los detalles con una estructura jerárquica, con las funciones principales en la parte superior y con niveles sucesivos de subfunciones que revela detalles de arranque bien delimitadas.
- Una "tabla de nodo" que proporciona un índice rápido para la localización de información dentro de la estructura jerárquica de los diagramas.
- La limitación de detalle a no más de seis subfunciones en cada función sucesiva.

Utilizando esta técnica de modelado se pueden emplear dos notaciones diferentes: AS-IS y TO-BE. La notación AS-IS es empleada para representar los procesos del negocio de la manera en que se encuentran antes de la automatización de los mismos. Y la notación TO-BE es utilizada para representar la transformación que se le desea dar al proceso, después de identificarse las posibles mejoras y tomando en cuenta los objetivos perseguidos por la empresa. Importante es saber que se utiliza como homóloga del flujo de trabajo Modelamiento del Negocio que propone RUP. (20) (21)

1.5 Lenguajes

UML

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Permite la modelación de sistemas con tecnología orientada a objetos, combinar diferentes elementos gráficos y crear diagramas.

El modelado es el diseño de aplicaciones de software antes de codificar. El modelado es una parte esencial de los grandes proyectos de software e incluso útil para proyectos medianos y pequeños. Un modelo juega un papel análogo en el desarrollo de software igual a la que juegan los planos y otros planes (mapas del sitio, elevaciones, modelos físicos) en la construcción de un rascacielos. Ayuda a los usuarios a entender la realidad de la tecnología y la posibilidad de que reflexione antes de invertir y gastar grandes cantidades en proyectos que no estén seguros en su desarrollo, reduciendo el coste y el tiempo empleado en la construcción de los productos. (22)

UML está compuesto por Elementos, Diagramas y Relaciones, los principales diagramas de UML están clasificados en:

- Diagramas de estructura estática
- Diagramas de comportamiento.
- Diagramas de implementación.

Pretende proporcionar a los usuarios un lenguaje de modelado visual expresivo y utilizable para el desarrollo e intercambio de modelos significativos e integrar las mejores prácticas empleadas hasta el momento. No posee propietario y está basado en el común acuerdo de gran parte de la comunidad informática.7 UML posee una gran cantidad de propiedades que han sido las que en realidad han contribuido a hacer del mismo, el estándar de facto de la industria que es en realidad, entre las cuales se pueden mencionar:

- Concurrencia, es un lenguaje distribuido y adecuado a las necesidades de conectividad actuales y futuras.
- Ampliamente utilizado por la industria desde su adopción por OMG.
- Reemplaza a decenas de notaciones empleadas con otros lenguajes.
- Modela estructuras complejas.
- Las estructuras más importantes que soportan tienen su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables si es necesario.

- Comportamiento del sistema: casos de uso, diagramas de secuencia y de colaboraciones, que sirven para evaluar el estado de las máquinas. (18)

Python

Python es un lenguaje de programación de notable alcance dinámico que se utiliza en una amplia variedad de aplicaciones. Es un lenguaje interpretado, característica que ahorra un tiempo considerable en el desarrollo del programa, al no ser necesario compilar ni enlazar.

Algunas de las principales características que tiene son:

- Clara sintaxis de lectura.
- Fuerte capacidad de introspección
- Orientación a objetos intuitiva.
- Expresión natural del código de procedimiento.
- Modularidad completa, apoyado en paquetes jerárquicos.
- Manejo de errores basados en excepciones.
- Tipos de datos de alto nivel.
- Extensa biblioteca estándar.
- Extensiones y módulos fácilmente escritos en C, C++ (o Java para Jython, o .NET para IronPython). Se puede incrustar dentro de las aplicaciones como una interfaz de secuencias de comandos.

La implementación de Python está bajo una licencia de código abierto que hace que sea de libre uso y distribución, incluso para uso comercial. La licencia de Python es administrada por la Python Software Foundation. Posee una amplia documentación y se puede usar tanto en Windows como en Linux. (23)

1.6 Herramientas

Visual Paradigm

Visual Paradigm for UML es una herramienta de diseño UML para el desarrollo de aplicaciones de software que soporta el ciclo completo del desarrollo del software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. (24)

EasyEclipse IDE

EasyEclipse posee el plug-in PyDev que permite programar en Python usando este IDE y es distribuido bajo la Eclipse Public License. EasyEclipse es un IDE para desarrolladores pequeños o individuales que necesiten un conjunto específico de herramientas de desarrollo lo más pequeño y simple posible. EasyEclipse es multiplataforma, gratuito y de código abierto. (25)

1.7 Interfaces gráficas de usuario

1.7.1 QT Designer

QT Designer es una herramienta para el desarrollo de formularios y presentaciones gráficas para las aplicaciones. Permite acelerar el desarrollo de interfaces de alto rendimiento, a la vez que proporciona una forma fácil de diseñar interfaces gráficas de usuario avanzadas generando el código fuente para las mismas, lo que permite al desarrollador ajustarlo a sus necesidades.

Las características de QtDesigner se mencionan a continuación:

- Ofrece una caja de componentes muy variada para interfaces de usuario.
- Posee un inspector de objetos para explorar los componentes usados en el desarrollo.

Posee un editor de propiedades, que permite cambiar las características de los objetos presentes en la interfaz, características como el texto que presenta este en la interfaz, fuente, tamaño, el nombre del objeto, entre otros.
Posee un editor de Señales (SIGNAL) y ranuras (SLOT), donde se puede asignar comportamientos predefinidos a ciertos objetos de la interfaz.
El QT Designer utiliza como base la librería gráfica de QT, que ha sido transportada a diversas plataformas, lo que permite que el código generado por el QT Designer pueda ser utilizado en diversas plataformas. Además, el QT funciona sólo o asociándose a algunos entornos de desarrollo integrado como Visual Studio .Net o Eclipse. Esta herramienta provee características muy poderosas como la pre-visualización de la interfaz, soporte para widgets y un editor de propiedades bastante poderoso. (26)

1.7.2 Librería Qt

Qt son un conjunto de librerías multi-plataforma para el desarrollo del esqueleto de aplicaciones GUI (interfaz gráfica de usuario), escritas en código C++. Qt además está completamente orientado a objetos.

Se caracteriza por:

- La creación de interfaces gráficas. Se distribuye bajo una licencia libre GPL (o QPL) que permite incorporar las interfaces Qt en las aplicaciones open-source.
- Se encuentra disponible para una gran número de plataformas: Linux, MacOs X, Solaris, HP-UX, UNIX con X11. Además, existe también una versión para sistemas empotrados.
- Es orientado a objetos, lo que facilita el desarrollo de software. El lenguaje para el que se encuentra disponible es C++ aunque han aparecido bindings (una adaptación de una biblioteca para ser usada en un lenguaje de programación distinto al que ha sido escrito) a otros lenguajes como Python o Perl.
- Es una librería que se basa en los conceptos de widgets (objetos), Señales-Slots y Eventos.
- Las señales y los slots son el mecanismo para que unos widgets se comuniquen con otros.
- Los widgets pueden contener cualquier número de hijos. El widget "top-level" puede ser cualquiera, sea ventana, botón, etc.
- Algunos atributos como el texto de etiquetas, se modifican de modo similar al lenguaje HTML.

Qt proporciona, además, otras funcionalidades:

  - Librerías básicas: Entrada/Salida, Manejo de Red, XML
  - Plugins, librerías dinámicas (Imágenes, formatos, entre otros)
  - Unicode, Internacionalización. (27)

1.8 Conclusiones

En el presente capítulo se hizo un estudio de las principales soluciones en el mundo para la virtualización y las librerías, API y herramientas para la administración de máquinas virtuales, obteniendo experiencia de sus principales características constribuyendo a un desarrollo más eficiente del software. Se analizó también el lenguaje de programación, el IDE, la metodología de desarrollo que se emplearán en el desarrollo del sistema. Se seleccionó KVM como hypervisor ya que es libre, de amplia documentación, usa la virtualización completa que es la más usada actualmente, con el beneficio de virtualizar sobre la aquitectura x86 virtulizando cualquier sistema operativo que la soporte, también se decidió la utilización de la API libvirt como interfaz para la gestión de kvm, debido a su fácil uso, gran seguridad, amplia documentación y cuenta con una implementación para Python que es el lenguaje usado.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

En el presente capítulo se describen los procesos necesarios para la realización de la virtualización. Se utiliza notación IDEF0 para la modelación del negocio posibilitando una mejor comprensión de los procesos. También se identifican los requisitos funcionales y no funcionales para definir los casos de uso, los actores y sus relaciones dentro del sistema.

2.2 Problema y situación problemática

Poner la misma de arriba cuando esté terminada.

2.3 Información que se maneja

Es de vital importancia conocer cómo se maneja la información del sistema en los procesos que se realizan. Las principales informaciones que se manejan son ficheros .img que van ser los discos duros de las máquinas virtuales y las información de configuración de las mismas guardada en ficheros xml. También se utiliza un fichero xml para guardar la lista de host y máquinas virtuales gestionadas.

2.4 Solución propuesta

La solución propuesta brinda la posibilidad a los administradores, de adicionar y eliminar los host en los cuales estará instalado el hypervisor KVM y la API libvirt que permitirán virtualizar y gestionar los recursos virtualizados de forma remota y distribuida, además brinda la posibilidad de administrar las máquinas virtualizadas con funcionalidades como iniciar, apagar, eliminar, mostrar y aumentar la cantidad de procesadores que contienen y así disponer de servidores virtualizados en los cuales instalar los servicios LDAP, DNS, DHCP como se puede apreciar en la siguiente figura:

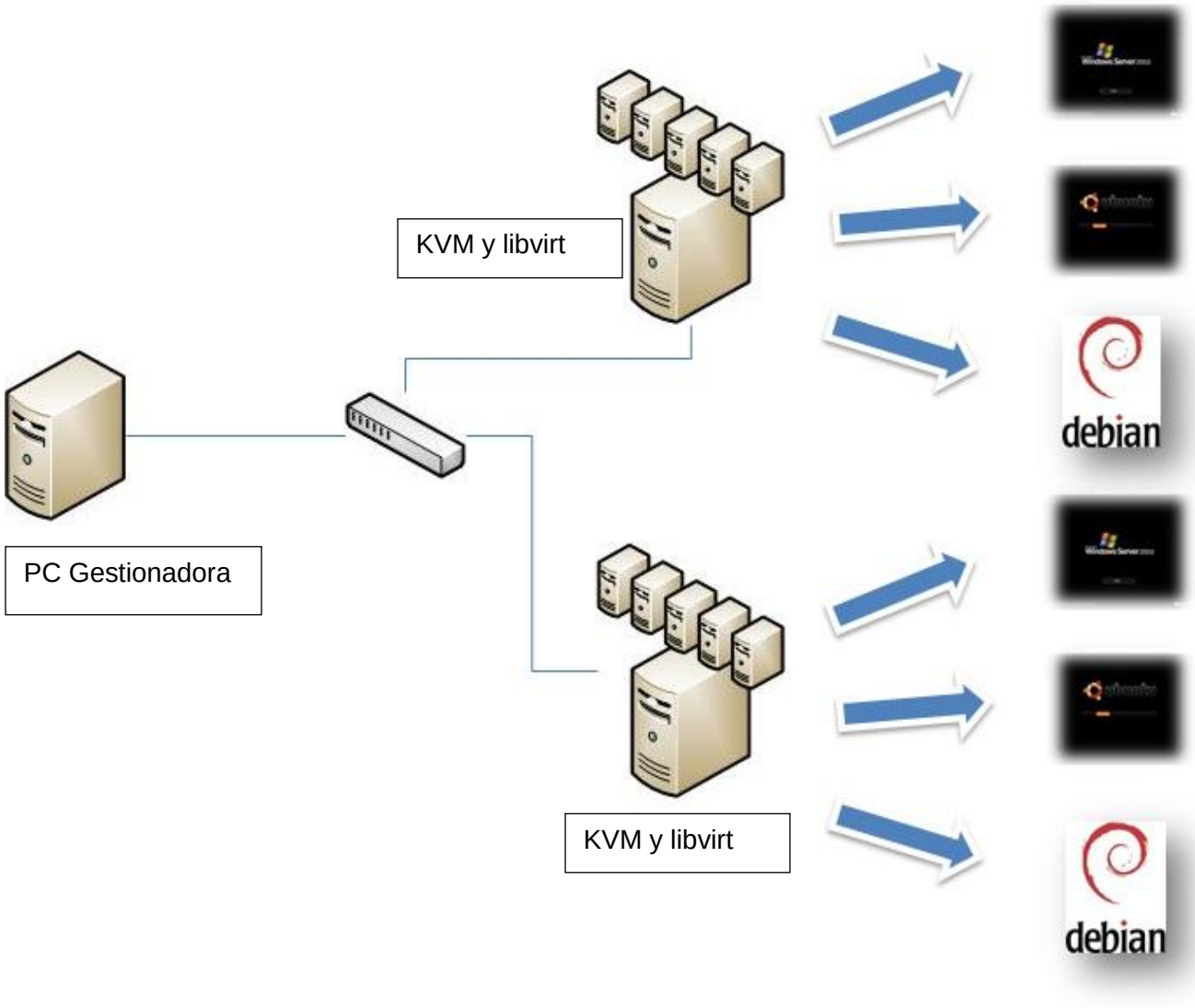


Figura 1: Solución propuesta

2.5 Modelo del Negocio

Para la modelación de los procesos del negocio se utiliza la notación IDEF0, este garantiza la identificación de estos procesos de una manera más simple. A continuación se muestran los procesos del negocio.

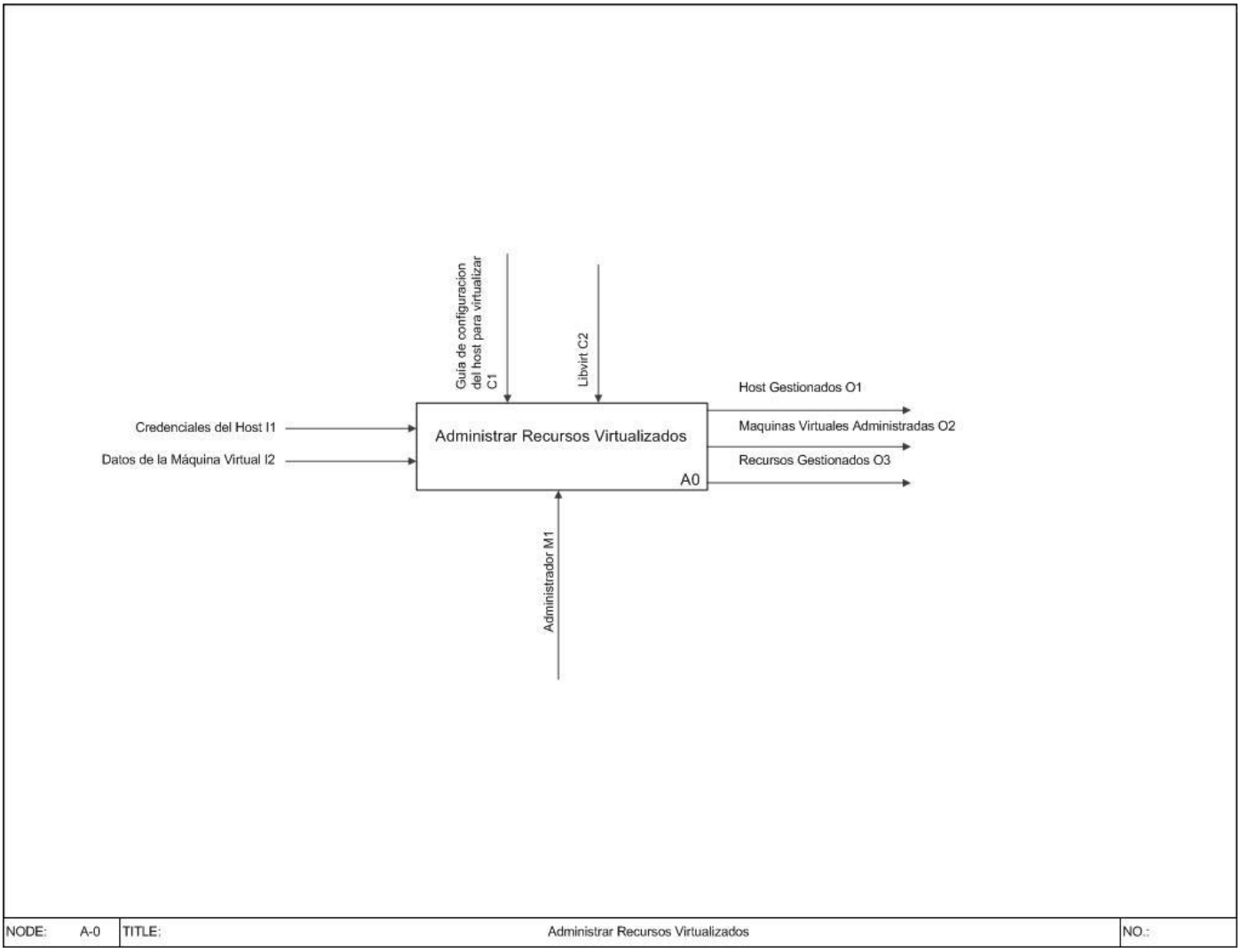


Figura 2: Representación del proceso general “Administrar Recursos Virtualizados”

Descripción

Administrar Recursos Virtualizados: Proceso principal que contiene las funcionalidades de Gestionar Host, Gestionar Máquina Virtual, Gestionar Recursos de Hardware mediante la librería libvirt y un Guía de Configuración obteniendo como salida los Host Gestionados, las Máquinas Virtuales Administradas y los Recursos de Hardware Gestionados.

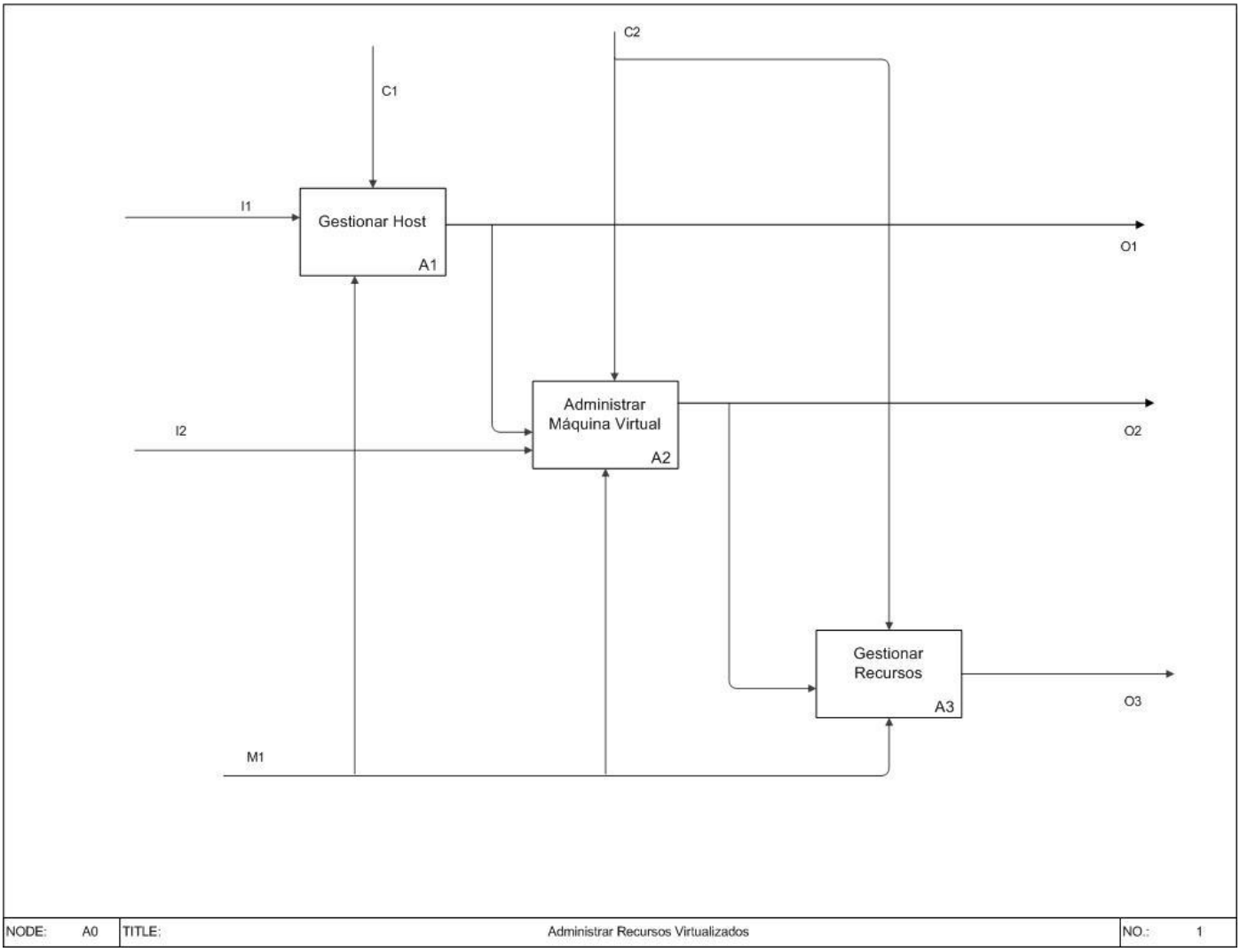
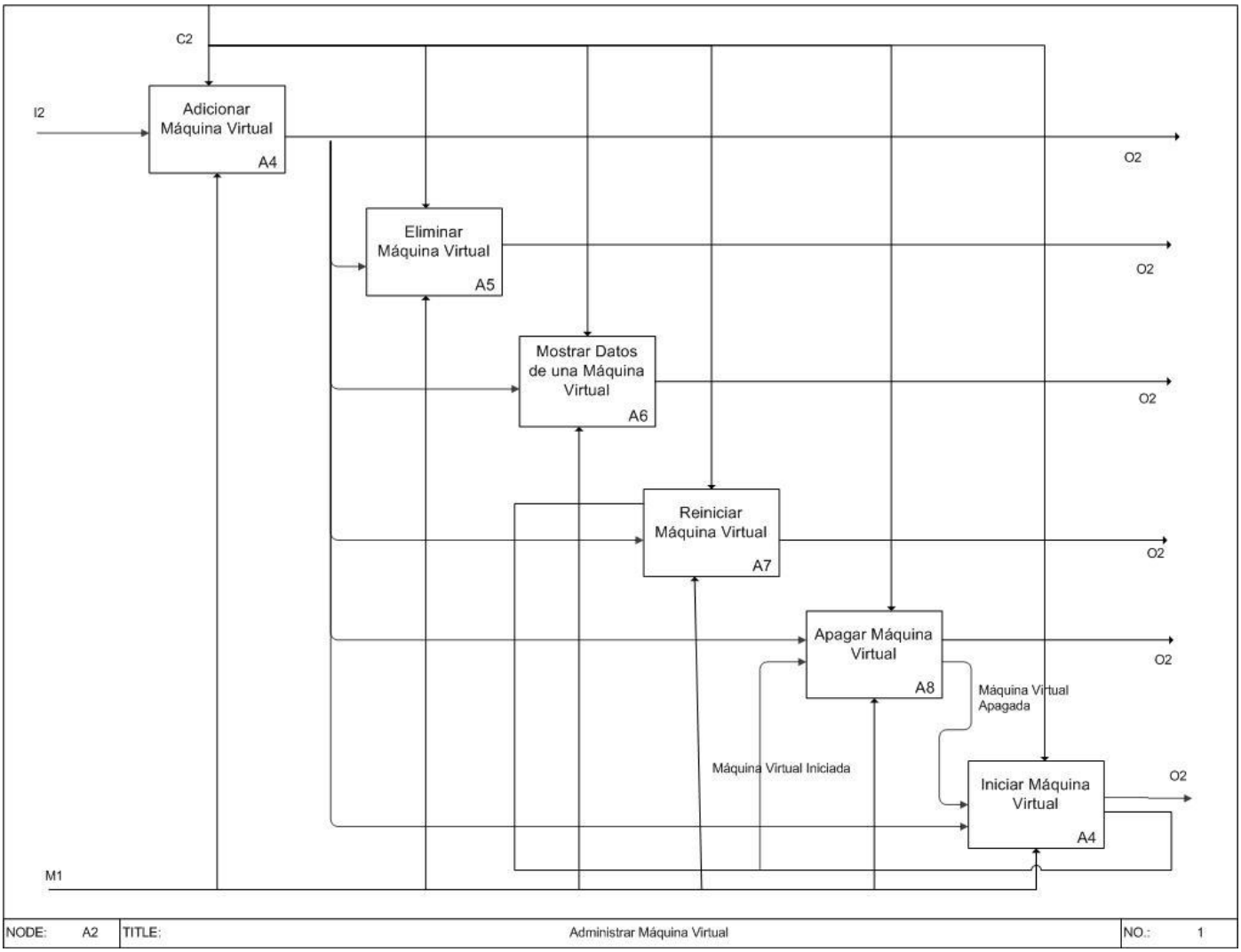


Figura 3: Detalles del proceso general

Descripción

Gestionar Host: Este proceso se encarga de adicionar, mostrar, eliminar y listar los host. La información que se muestra de un host es la cantidad de procesadores, la memoria RAM, cantidad de hilos, el nombre, la dirección ip, el uso del CPU y la memoria RAM. Tiene como entrada las credenciales del host (Ip, usuario y contraseña)

Gestionar Recursos: Proceso encargado de modificar los recursos de hardware de una máquina virtual (adicionar o eliminar procesadores y memoria RAM) siendo su entrada una máquina virtual.



Administrar Máquinas Virtuales: Proceso que se encarga mediante la API Libvirt de apagar, iniciar, reiniciar, adicionar, eliminar y mostrar información de las máquinas virtuales (nombre, tamaño del disco duro, cantidad de procesadores, memoria RAM total y usada, ip del host y el estado). Tiene como entrada los datos de una máquina virtual (nombre, sistema operativo, distribución, RAM, tamaño del disco duro, nombre del disco duro, Ip del host, cantidad de procesadores) y un host gestionado.

2.6 Especificación de los requisitos de software.

La captura de requisitos es una de las etapas más importantes del desarrollo del software, determinándose lo que el cliente quiere. Requiere una visión común entre el cliente y el equipo de

desarrollo sobre todos los procesos del negocio. Esta tarea generalmente presenta dificultades debido a la falta de conocimientos de los clientes sobre las tecnologías informáticas.

2.6.1 Requisitos funcionales

Los requisitos funcionales proyectan las capacidades con que debe contar el sistema. Se presentan a continuación los requerimientos funcionales del sistema:

RF1. Gestionar Host.

- RF1.1. Adicionar Host
- RF1.2. Mostrar Host
- RF1.3. Listar Host
- RF1.4. Eliminar Host

RF2. Administrar Máquina Virtual.

- RF2.1. Iniciar
- RF2.2. Apagar
- RF2.3. Detener
- RF2.4. Suspender
- RF2.5. Reanudar
- RF2.6. Reiniciar
- RF2.7. Mostrar Datos de Máquina Virtual
- RF2.8. Crear Máquina Virtual
- RF2.9. Eliminar Máquina Virtual

RF3. Gestionar Recursos de Hardware.

- RF3.1. Adición de Recursos de Hardware
- RF3.2. Eliminación de Recursos de Hardware

2.6.2 Requisitos no funcionales

Los requerimientos no funcionales son cualidades que el producto deberá tener, teniendo una estrecha relación con los requerimientos funcionales.

Requerimientos de hardware:

Los requisitos del hardware para la aplicación son bajos y depende de la cantidad de máquinas virtuales

controladas, se puede emplear*:*

- Microprocesador Pentium IV.
- 512 MB de memoria RAM.
- 4 GB de espacio libre en el disco duro.

Restricciones en el diseño y la implementación:

Las herramientas utilizadas están divididas en: Herramientas de diseño: QtDesigner en su versión 4.2.1 para elaborar las interfaces de usuario sencillas.

Herramientas de codificación: Eclipse con Pydev. Pydev es un plugin o complemento para el entorno de desarrollo Eclipse que permite la programación en Python. Será empleado Python 2.6 como lenguaje de programación para la construcción del sistema. La codificación estará regida por la guía de estilo propuesta por Guido van Rossum creador del lenguaje de programación Python.

Como arquitectura general del sistema se ha determinado utilizar una arquitectura en capas como estilo arquitectónico, debido a las ventajas que ello proporciona en términos de claridad, facilidad en la corrección de errores, rápido desarrollo al permitir el trabajo paralelo en cada una de las capas definidas, entre otros factores.

Han sido utilizadas bibliotecas de clases o framework creado en el proyecto Servicios Telemáticos que faciliten el desarrollo y acorten los tiempos de producción, permitiendo a los desarrolladores concentrarse en aspectos de mayor nivel ganando en tiempo de implementación y reutilización de código. El sistema propuesto se ha apoyado en un framework desarrollado en el proyecto anteriormente mencionado para resolver problemas puntuales asociados a la gestión de servidores.

Requerimientos de apariencia o interfaz externa:

El sistema deberá contener una interfaz agradable, sencilla y sugerente. No se presentarán excesivas imágenes o gráficos que puedan retardar el tiempo de respuesta de la aplicación, se hará uso fundamentalmente de pequeños íconos capaces de ilustrar las funcionalidades en combinación con textos claros. La navegación a través del sistema se realizará empleando un menú en la región superior que elimine el exceso de interfaces extras y ayude a una mejor orientación de los usuarios.

Requerimientos de seguridad:

Los requerimientos de seguridad son de gran importancia porque el sistema propuesto utiliza la red como medio fundamental de comunicación entre el software de gestión y las Máquinas Virtuales. La gestión remota se hace a través de libvirt que usa el cifrado TLS, el certificado X509 y ssh.

Requerimientos de usabilidad:

Estos describen los niveles apropiados de usabilidad, según los usuarios finales del producto.

- La interfaz será fácil de usar para los diversos administradores que interactúen con ella.
- El sistema estará bien documentado, con el fin de lograr un mejor uso de los servicios que este ofrecerá, para ello se realizará una ayuda que explique paso a paso cada una de las funcionalidades del software.

Requerimientos de soporte:

El sistema contará antes de su puesta en marcha con un período de pruebas que permitirá identificar y corregir errores cometidos durante el desarrollo.
Los ordenadores donde se vaya a ejecutar la aplicación necesitan tener instalado una distribución de GNU/Linux. Además debe poseer los paquetes, librerías y dependencias mínimas necesarias que van a hacer usadas por la aplicación. Los servidores donde se va a realizar la virtualización necesitan KVM, Libvirt.

Requerimientos legales:

El Módulo Virtualización v1.0 será propiedad exclusiva de la Universidad de las Ciencias Informáticas (UCI).

Requerimientos políticos y culturales:

El sistema podrá ser utilizado en primera instancia en el territorio nacional y posteriormente en otros países que se ajusten a las normas de comercialización dispuestas por la UCI. El sistema constará inicialmente con el idioma español.

2.7 Modelo de Casos de Uso del Sistema

2.7.1 Actores del Sistema

El actor principal del Módulo Virtualización v1.0 es el administrador del servidor.

Actor	Justificación
Administrador	Es la persona encargada de realizar todos los procesos de gestión y administración. Interactúa con la aplicación para realizar las distintas operaciones de la misma.

Tabla 1: Actor del sistema

Un diagrama de casos de uso del sistema representa gráficamente a los procesos y su interacción con los actores. Para el Módulo Virtualización v1.0 queda de la siguiente forma:

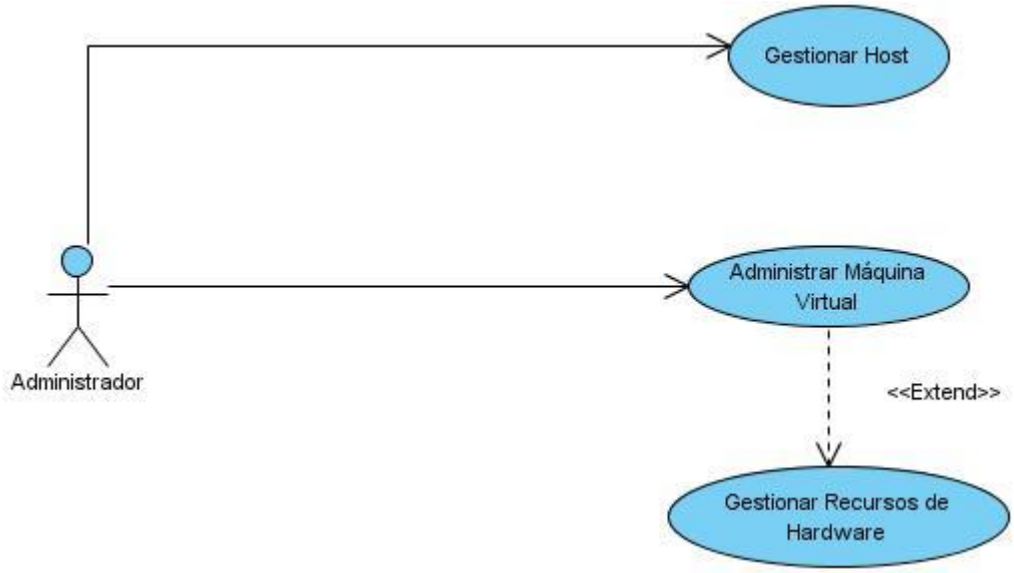
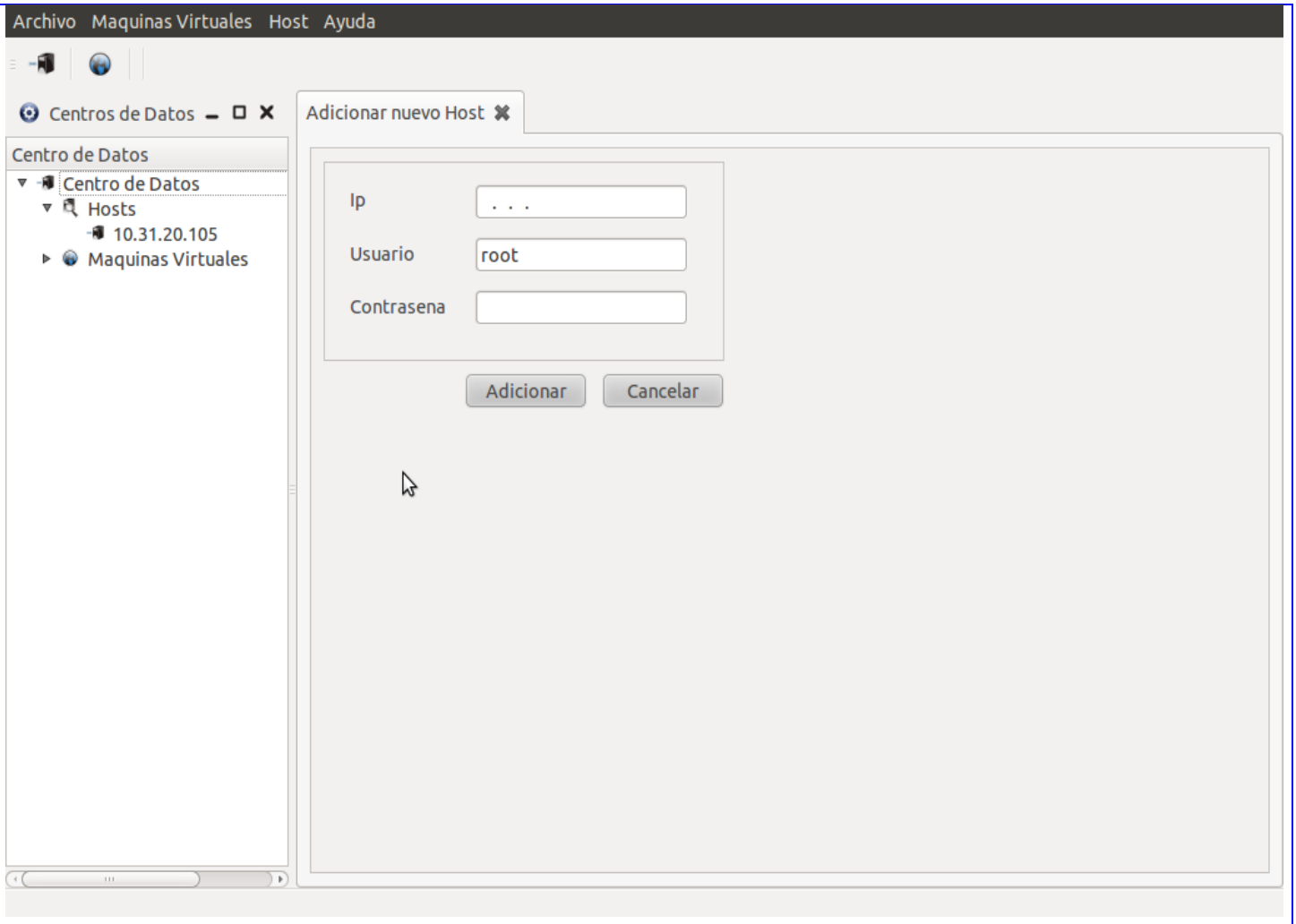


Figura 4: DCUS

2.7.4 Descripción de los CUS y prototipos de interfaces

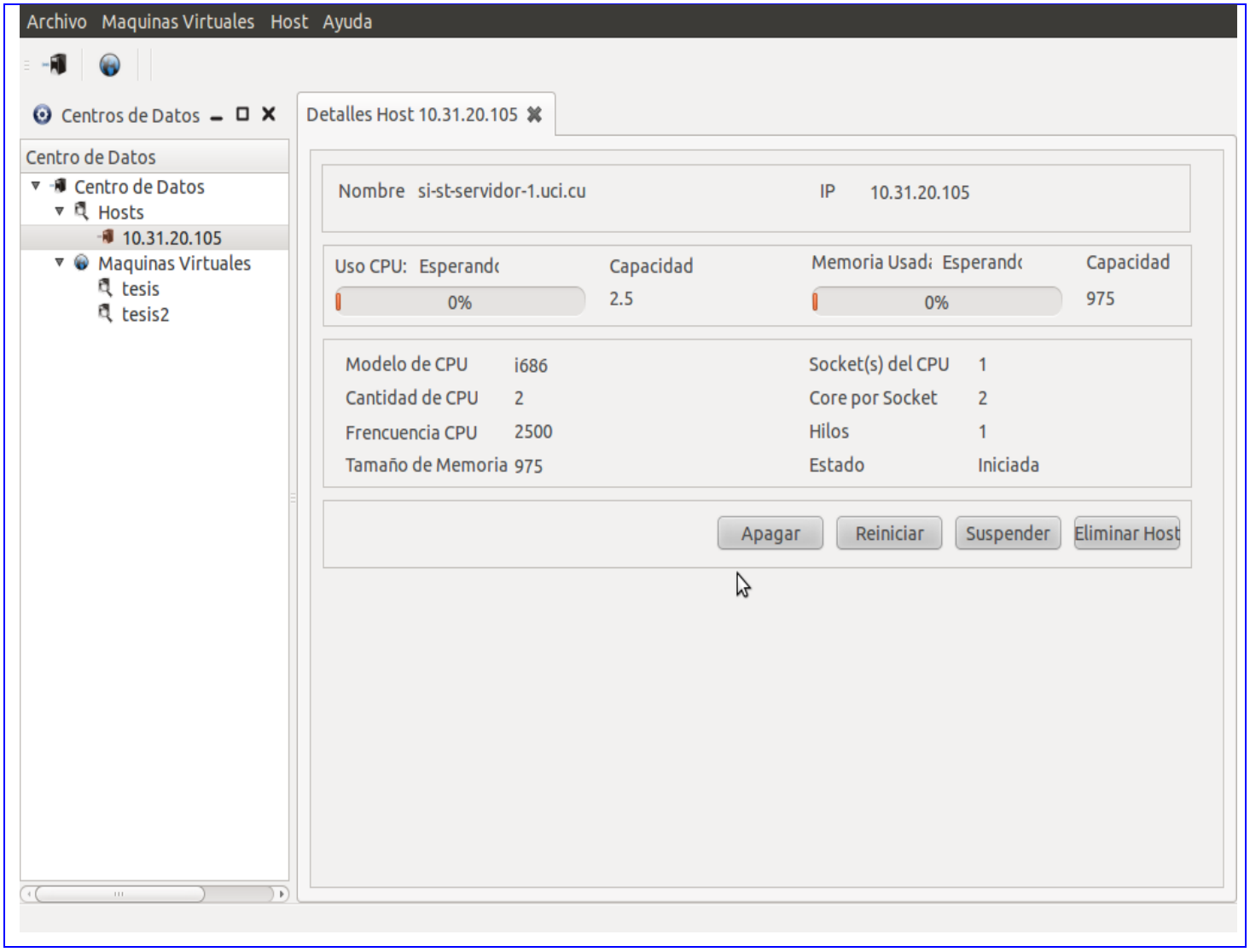
A continuación se realizará la descripción de los casos de uso del sistema permitiendo conocer más detalladamente el funcionamiento del sistema.

<table>
<tr><td>Caso de Uso:</td><td colspan="2">Gestionar Host</td></tr>
<tr><td>Actores:</td><td colspan="2">Administrador</td></tr>
<tr><td>Resumen:</td><td colspan="2">En el presente caso de uso el administrador realiza las siguientes operaciones: Adicionar un nuevo host introduciendo los datos del mismo, mostrar datos del host, la cual brinda información detallada de las características de su hardware y su uso, y listar mostrando en todo momento la cantidad de host gestionados.</td></tr>
<tr><td>Precondiciones:</td><td colspan="2">Administrador autenticado</td></tr>
<tr><td>Referencias</td><td colspan="2">RF-1</td></tr>
<tr><td>Prioridad</td><td colspan="2"></td></tr>
<tr><td colspan="3">Flujo Normal de Eventos</td></tr>
<tr><td colspan="3">Sección “Adicionar Host”</td></tr>
<tr><td colspan="2">Acción del Actor</td><td>Respuesta del Sistema</td></tr>
<tr><td colspan="2">1. Selecciona en el menú Host, la opción Nuevo Host.</td><td>2. Muestra los campos para introducir los datos del host.
Ip
Usuario
Contraseña</td></tr>
<tr><td colspan="2">3. Introduce los datos del host y oprime el botón adicionar.</td><td>4. Comprueba los datos y adiciona el nuevo host a la lista.</td></tr>
<tr><td colspan="2"></td><td>5. Informa que se adicionó correctamente.</td></tr>
<tr><td colspan="3">Prototipo de Interfaz</td></tr>
</table>



Flujo Alterno 5a "Datos incompletos o Incorrectos"	
Acción del Actor	**Respuesta del Sistema**
	1. El campo ip o el campo contraseña están vacíos o son erróneos; se muestra los mensajes de error: "Existe campos vacíos" o "Existen Datos Inválidos".
2. Retorna al paso 3 del flujo normal de eventos.	
Flujo Alterno 3a "Cancelar"	
Acción del Actor	

Flujo Alterno 3A "Cancelar"	Flujo Alterno 3A "Cancelar"
1. Selecciona el botón Cancelar	2. Se cancela la operación
3. Retorna al paso 3 del flujo normal de eventos.	
Flujo Normal de Eventos	
Sección "Mostrar Host"	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona un host.	2. Muestra la información detalla del host seleccionado.
Flujo Normal de Eventos	
Sección "Listar Host"	
Acción del Actor	**Respuesta del Sistema**
1. Despliega la lista de host.	2. Muestra la lista de host.
Flujo Normal de Eventos	
Sección "Eliminar Host"	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona un host y oprime el botón eliminar.	2. Muestra un interfaz de confirmación de eliminación del host.
3. Oprime el botón aceptar	4. Elimina el host de la lista.
Flujo Alterno 2a "Cancelar"	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona el botón Cancelar	2. Se cancela la operación
3. Retorna al paso 3 del flujo normal de eventos.	
Prototipo de Interfaz	



Caso de Uso:	Administrar Máquina Virtual
Actores:	Administrador
Resumen:	En el caso de uso el administrador realiza las operaciones de administración sobre las Máquinas Virtuales como adicionar, apagar , iniciar, reiniciar, mostrar consola donde se visualiza la MVM, eliminar y mostrar información detallada de las Máquinas Virtuales
Precondiciones:	Administrador autenticado
Referencias	RF-2

Prioridad	
Flujo Normal de Eventos	
Sección “Iniciar”	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona una Máquina Virtual y oprime en el botón “Iniciar”.	2. Inicia la Máquina Virtual seleccionada.

Flujo Normal de Eventos	
Sección “Apagar”	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona una Máquina Virtual y oprime el botón “apagar”.	2. Apaga la Máquina Virtual.

Flujo Normal de Eventos	
Sección “Mostrar Consola”	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona una Máquina Virtual y oprime el botón “Mostrar Consola”.	2. Detiene la Máquina Virtual.

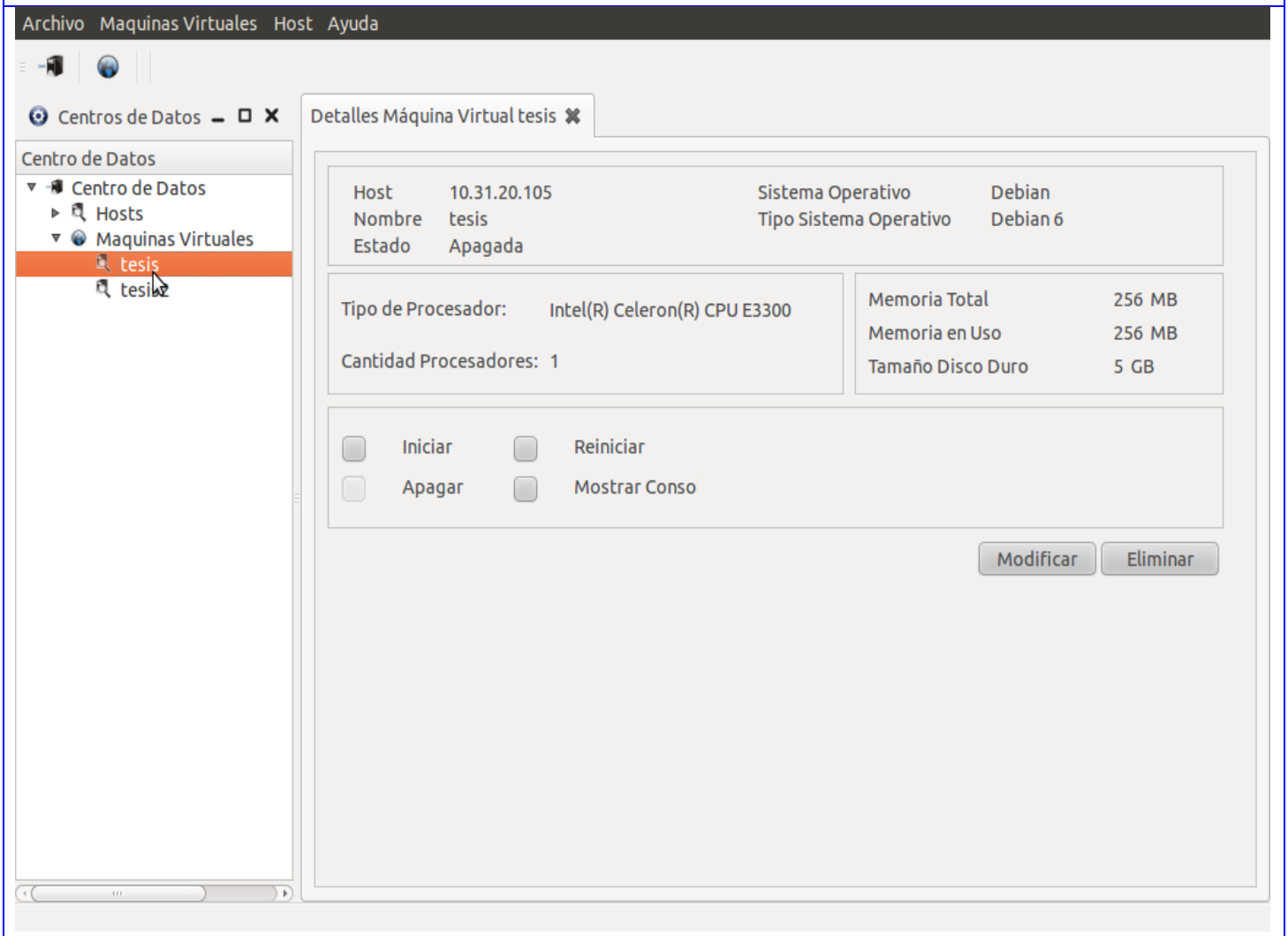
Flujo Normal de Eventos	
Sección “Reiniciar”	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona una Máquina Virtual y oprime el botón “Reiniciar”.	2. Reinicia la Máquina Virtual.

Flujo Normal de Eventos	
Sección “Mostrar Datos de Máquina Virtual”	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la una Máquina Virtual.	2. Muestra los datos de la Máquina Virtual.

Flujo Normal de Eventos	
Sección “Eliminar Máquina Virtual”	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona un una máquina virtual y oprime el	2. Elimina el host de la lista.

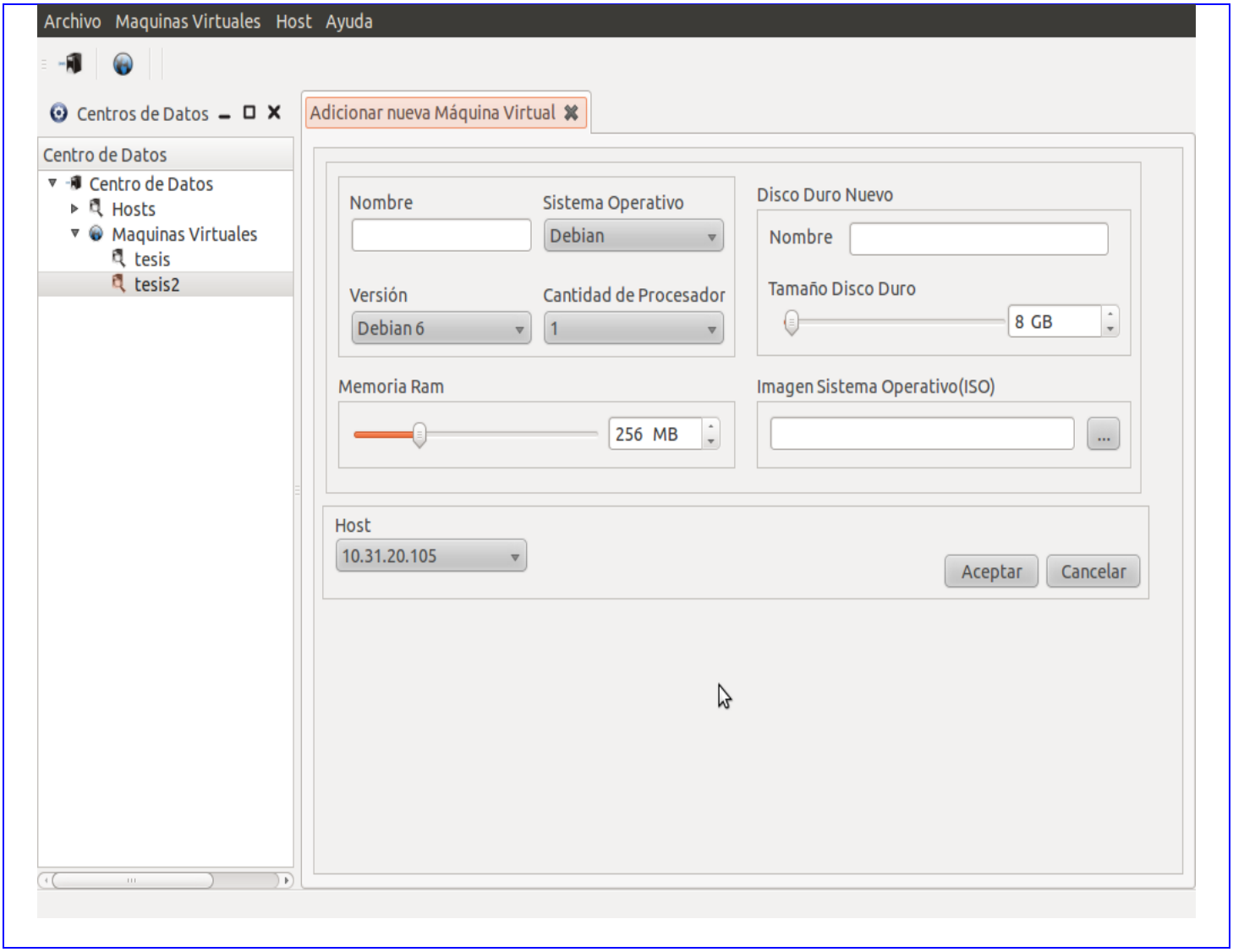
botón eliminar.	

Prototipo de Interfaz



Flujo Normal de Eventos	
Sección "Crear Máquina Virtual"	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona en el menú Máquinas Virtuales la opción "Nueva Máquina Virtual"	2. Muestra los campos para introducir los datos de la Máquina Virtual.
3. Introduce los datos de la nueva Máquina Virtual y oprime el botón aceptar.	4. Comprueba los datos y crea la nueva Máquina Virtual.
Flujo Alterno 4a "Datos incompletos o Incorrectos"	

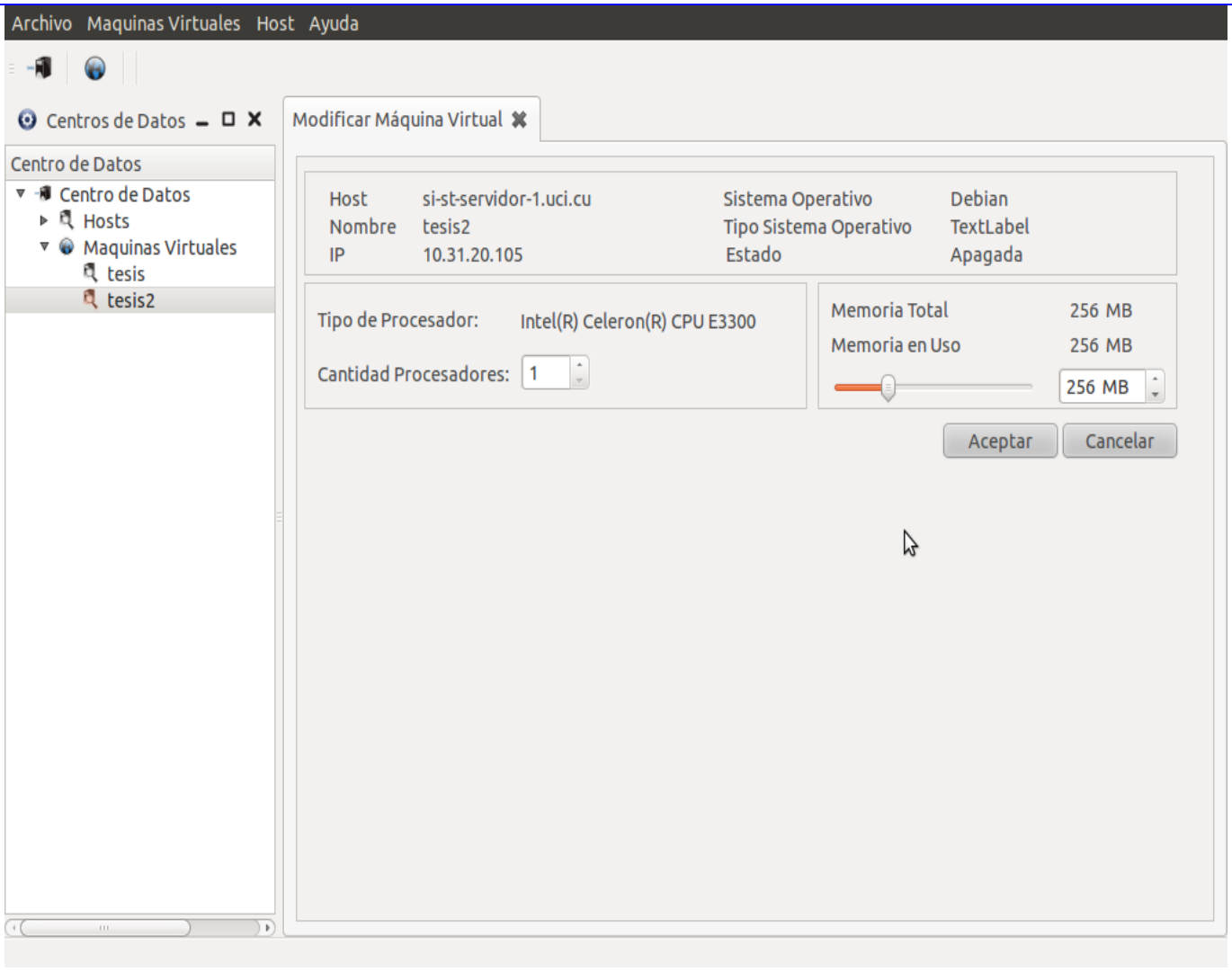
Acción del Actor	Respuesta del Sistema
	3. El campo nombre está vacío o es erróneo; se muestra los mensajes de error: “Existe campos vacíos” y “Existen Datos Inválidos”.
4. Retorna al paso 3 del flujo normal de eventos.	
Flujo Alterno 3a “Cancelar”	
Acción del Actor	**Respuesta del Sistema**
1. `Selecciona el botón Cancelar	2. Se cancela la operación
3. Retorna al paso 3 del flujo normal de eventos.	
Prototipo de Interfaz	



Caso de Uso:	Gestionar Recursos de Hardware
Actores:	Administrador
Resumen:	En el caso de uso el administrador realiza las operaciones de gestión sobre los recursos de hardware, adicionando o eliminando la cantidad de procesadores y cantidad de memoria RAM a una máquina virtual.
Precondiciones:	Administrador autenticado
Referencias	RF-3
Prioridad	

Flujo Normal de Eventos	
Sección “Adición de Recursos de Hardware”	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona una máquina virtual.	2. Muestra la información de la máquina virtual.
3. Da clic en el botón modificar.	4. Muestra la Interfaz para modificar los recursos de hardware cantidad de procesadores y memoria RAM.
5. Adiciona los recursos.	6. Adiciona los recursos a la máquina virtual.

Prototipo de Interfaz



Flujo Normal de Eventos	
Sección “Eliminación de Recursos de Hardware”	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona una máquina virtual.	2. Muestra la información de la máquina virtual.
3. Oprime el botón modificar.	4. Muestra la Interfaz para modificar los recursos de hardware cantidad de procesadores y memoria RAM.
5. Elimina los recursos.	6. Elimina los recursos de la máquina virtual.

2.8 Conclusiones

En el presente capítulo se realizó un estudio de los procesos del negocio que serán automatizados por la herramienta Módulo Virtualización v1.0. Se identificaron también los requisitos funcionales y no funcionales, y posteriormente se diseñaron los casos de uso que guiarán el desarrollo de la solución propuesta. Por último se describieron los casos de uso del sistema, y se mostraron los prototipos de interfaces del sistema.

CAPÍTULO 3: DISEÑO DEL SISTEMA

3.1 Introducción

Con el objetivo de crear un punto de partida para actividades de implementación, capturando los requisitos, interfaces y clases se realiza el diseño. Esta constituye una actividad de suma importancia para descomponer los trabajos de implementación en partes más fácil de manejar. En el presente capítulo se realizará el Diagrama de clases del Diseño de cada caso de uso.

3.2 Patrones

En la terminología de objetos, el patrón es una descripción de un problema y su solución que recibe un nombre y que puede emplearse en otros contextos; en teoría, indica la manera de utilizarlo en circunstancias diversas. Muchos patrones ofrecen orientación sobre cómo asignar las responsabilidades a los objetos ante determinada categoría de problemas. Los patrones no se proponen descubrir ni expresar nuevos principios de la ingeniería de software. Todo lo contrario: intentan codificar el conocimiento, las expresiones y los principios ya existentes. (28)En el sistema se hace uso de diferentes patrones de diseño utilizándose una arquitectura en capas y en específico el patrón n-capas el cual da una gran flexibilidad a la aplicación dado el bajo acoplamiento funcional que se logra con la misma, permitiendo a la aplicación adaptarse de forma relativamente fácil a cambios.

Dentro de los patrones creacionales que se utilizaron en el diseño de la aplicación se destaca:

Patrón **singleton:** este asegura que solo se pueda crear una instancia de la clase y ofrece un punto global de acceso a esta instancia. El uso de este patrón permite que la clase Centro de Datos solo pueda tener una instancia.

Patrón **Creador:** el patrón creador nos ayuda a identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases. Se utiliza en la clase Centro de datos y Host.

Patrón **controlador:** el patrón controlador es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Se utiliza principalmente en la capa de entorno en todas sus clases.

Patrón **Abstract Factory**: El propósito de este patrón es proporcionar una interfaz para crear familias de objetos relacionados sin especificar su clase. Se usa en el Framework.

Patrón **Decorator:** responde a la necesidad de añadir dinámicamente funcionalidad a un Objeto. Esto nos permite no tener que crear sucesivas clases que hereden de la primera. Se usa en el Framework en la librería de virtualización.

Patrón **Observer**: Define una relación de uno a muchos y al cambiar un objeto de estado los otros son notificados y actualizados automáticamente. (29) Se utiliza en todos los eventos que son lanzados.

3.3 Modelo de Diseño

El modelo de diseño describe la realización física de los casos de uso, se centra en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Sirve como una abstracción de la implementación y es utilizada como entrada fundamental de las actividades de implementación. La arquitectura escogida va estrechamente ligada al diseño de un sistema. La Arquitectura en Capas cuenta con disímiles ventajas como son la organización del modelo de diseño en capas, que pueden estar físicamente distribuidas, reflejándose en que los componentes de una capa sólo pueden hacer referencia a componentes en capas inmediatamente inferiores. Este es importante porque simplifica la comprensión y la organización del desarrollo de sistemas complejos, reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. Además, ayuda a identificar qué puede reutilizarse.

A continuación se muestra la estructura general del sistema en términos de capas y componentes:

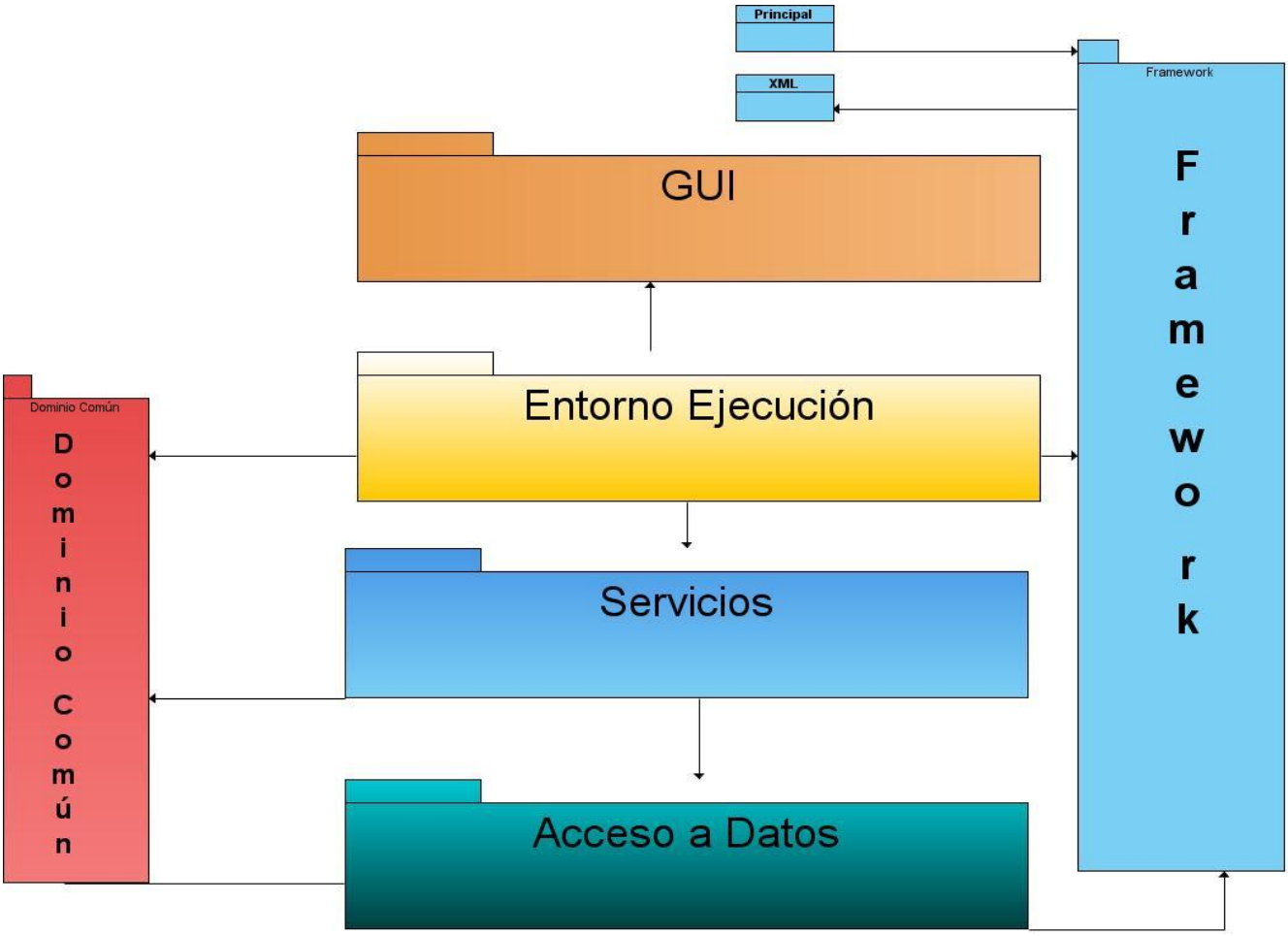


Fig. 8: Estructura General del Sistema

Existen cuatro capas principales las cuales son: GUI, Entorno de Ejecución, Servicios, y Acceso y Modificación de Datos. En el caso de la capa Entorno de Ejecución constituye una capa de abstracción entre GUI y Servicios. Es importante aclarar que se ha decidido construir un *framework* que se encargue de resolver problemas puntuales relacionados con la inicialización del sistema, la ejecución de funciones en paralelo, las conexiones remotas, entre otros aspectos; con el objetivo de poder reutilizar estas funcionalidades en futuros desarrollos de software y darle al desarrollador la posibilidad de concentrarse en aspectos puramente de negocio que se encuentran a un nivel superior de abstracción. La configuración de la estructura general del sistema estará reflejado en un archivo XML que será utilizado por el framework para conocer que componentes serán empleados y donde residirá su implementación

particular. Otro aspecto importante es el funcionamiento del componente Dominio Común, el cual contiene entidades o atributos que serán reutilizados por más de una clase en el sistema.

3.2.1 Estructura detallada Del Sistema

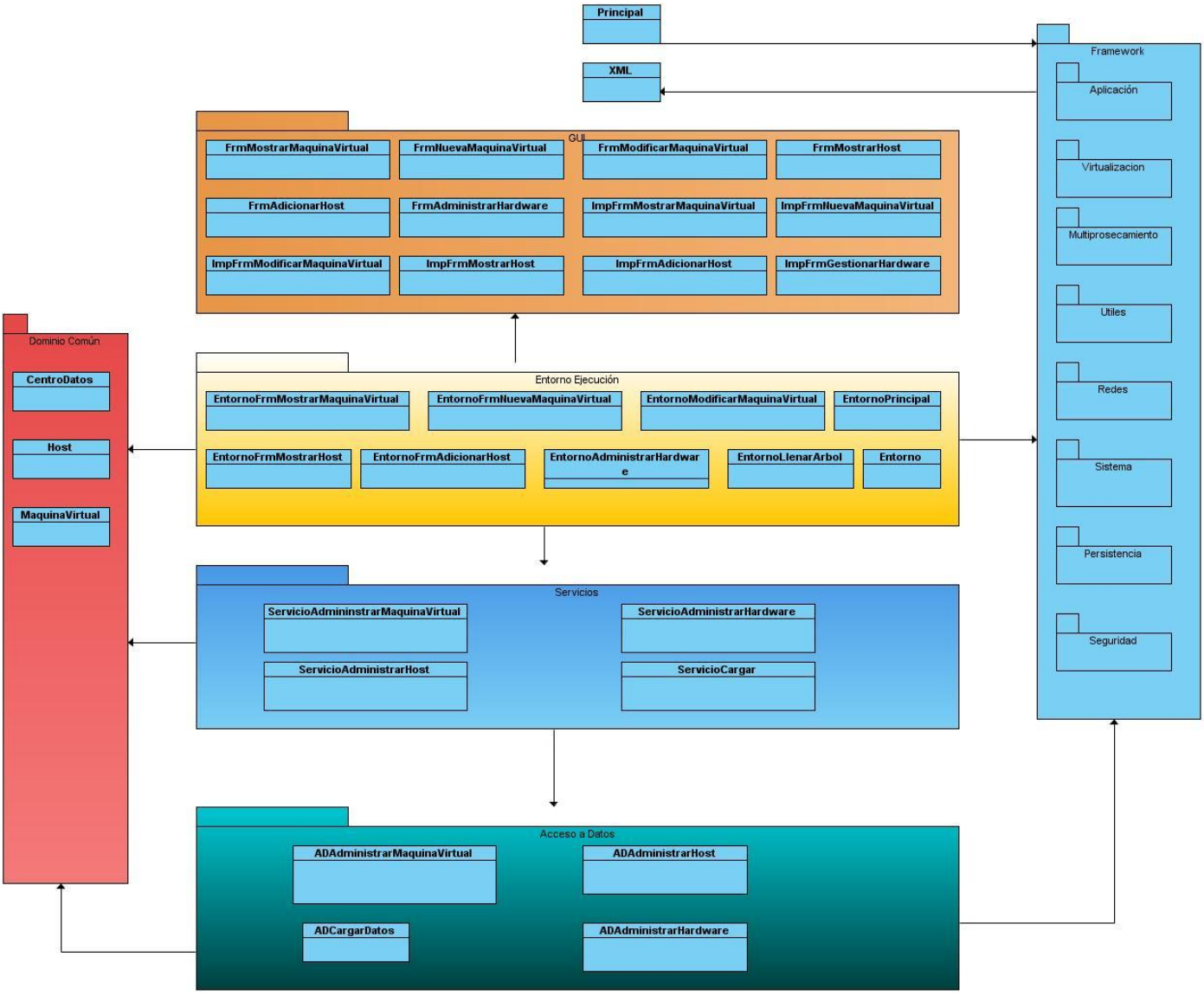


Fig. 9: Estructura detallada del Sistema

Diagrama de Clases del Diseño

A continuación se muestran los diagramas de clases de diseño por casos de uso.

Tabla 1: Diagrama de Clases del Diseño. CU Gestionar Host

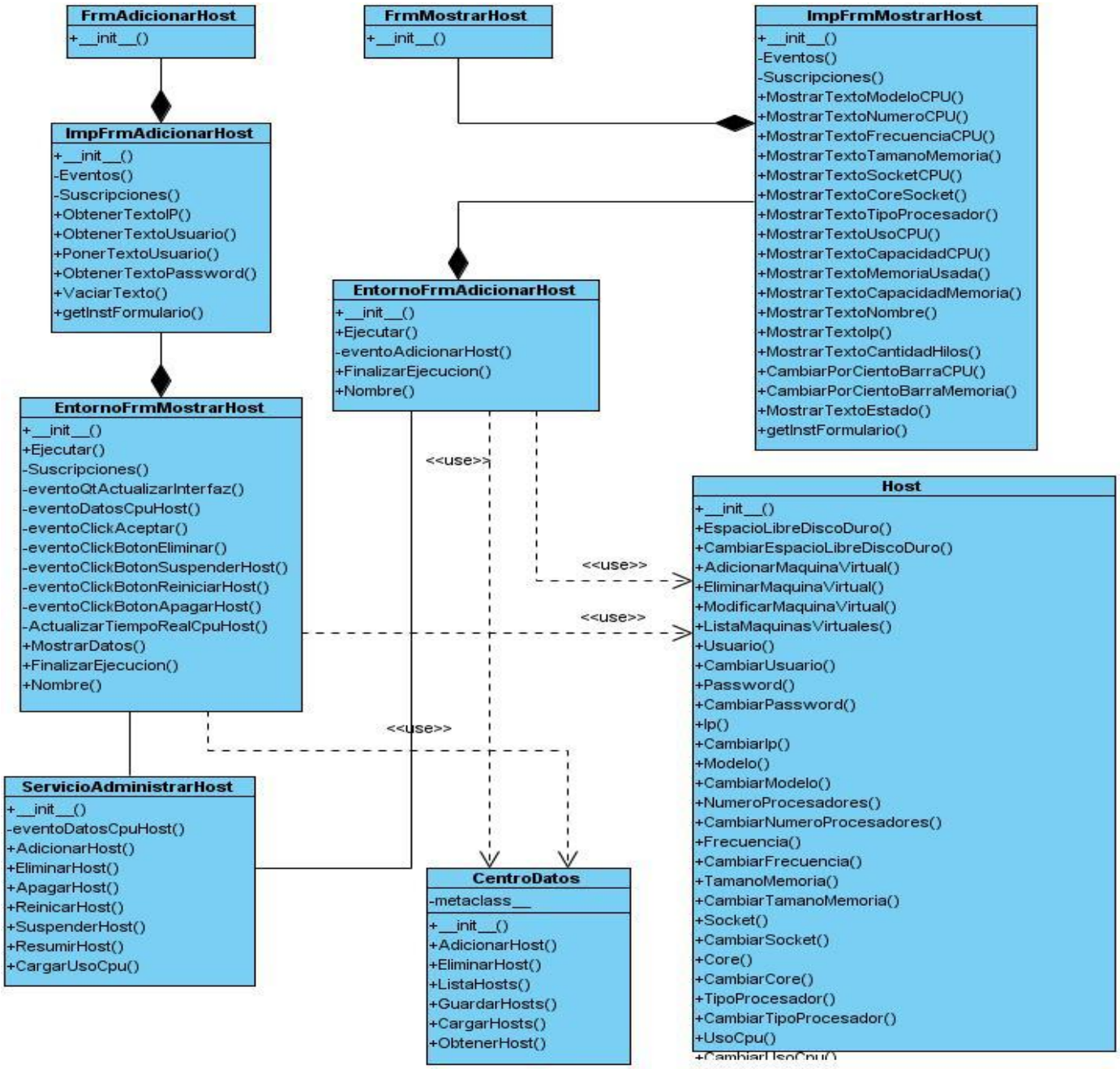


Tabla 2: Diagrama de Clases del Diseño. CU Administrar Máquina Virtual

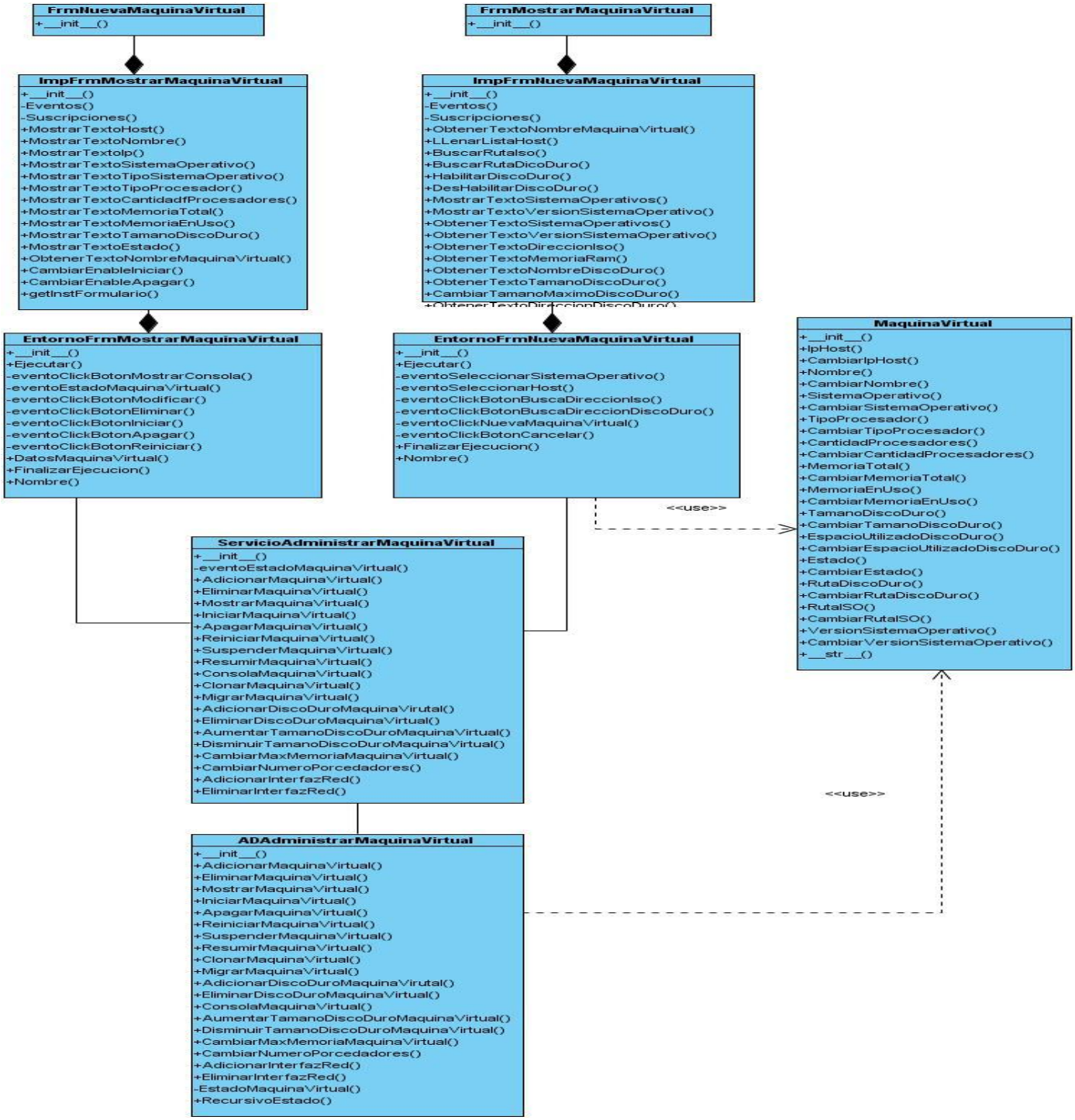
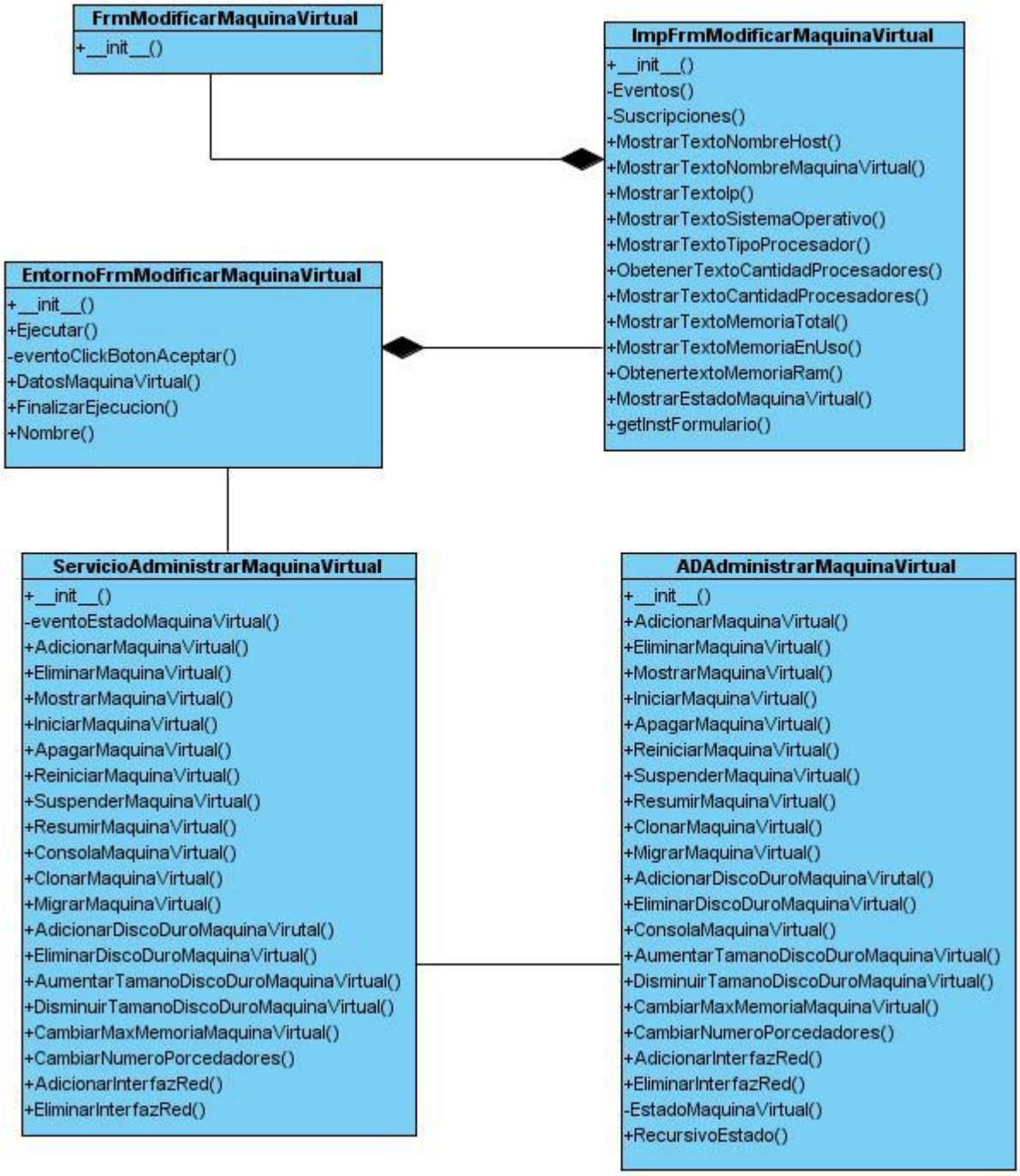


Tabla 3: Diagrama de Clases del Diseño. CU Gestionar Hardware



3.4 Conclusiones

En el presente capítulo fueron presentados los artefactos fundamentales correspondientes al diseño del sistema, entre ellos los modelos diseño, sentando las bases para una futura implementación del sistema, también se presentó la arquitectura empleada, donde se hizo una explicación detallada de la misma, se abordaron las capas existentes y su funcionamiento, buscando que la aplicación sea lo más escalable y extensible posible.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

4.1 Introducción

En el presente capítulo se desarrolla el flujo de trabajo correspondiente a la implementación y prueba. Para el desarrollo del mismo se tuvieron en cuenta artefactos que fueron generados en el diseño del sistema, a partir de los cuales se desarrolló la implementación del sistema en términos de componentes. Además se representa el diagrama de despliegue que contiene la topología hardware sobre la que se ejecuta el sistema. También para las pruebas al sistema donde se escoge una estrategia de prueba, haciéndose en diferentes niveles de prueba utilizando un método prueba con su respectiva técnica.

4.2 Diagrama de despliegue

El diagrama de despliegue representa la arquitectura en tiempo de ejecución de los procesadores, dispositivos y los componentes de software que se ejecutan en esa arquitectura. Es la última descripción física de la topología del sistema y describe la estructura de las unidades de hardware. Además, representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Un nodo es un recurso de ejecución tal como un procesador, un dispositivo o memoria. En los procesadores es donde se encuentran alojados los componentes.

Figura 1 Diagrama de Despliegue

4.3 Diagrama de componentes

Para lograr una mejor comprensión de los componentes que forman el sistema, se presentarán los diagramas de componentes. Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo de diseño.

Tabla 4: Diagrama de Componentes General

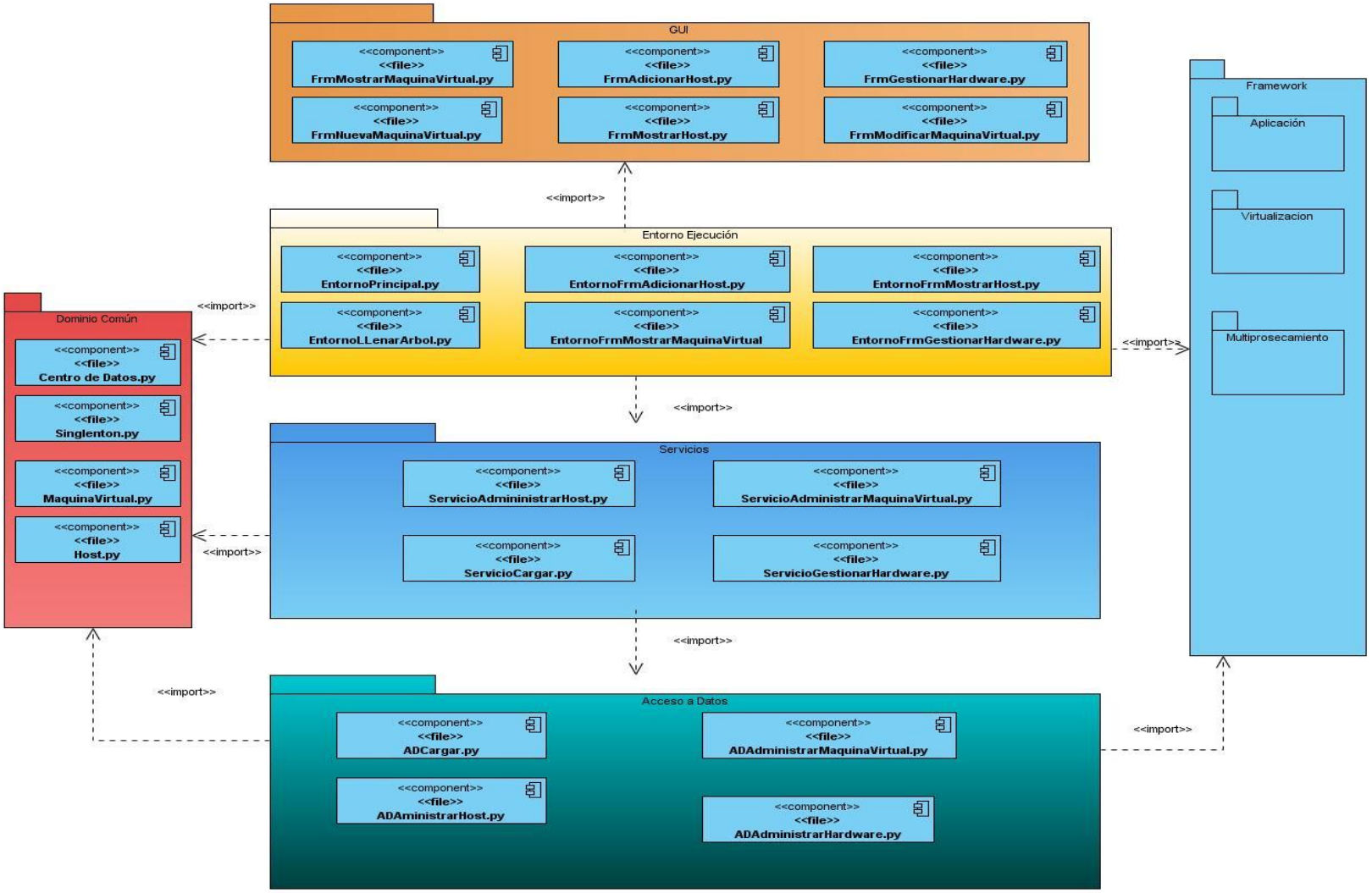
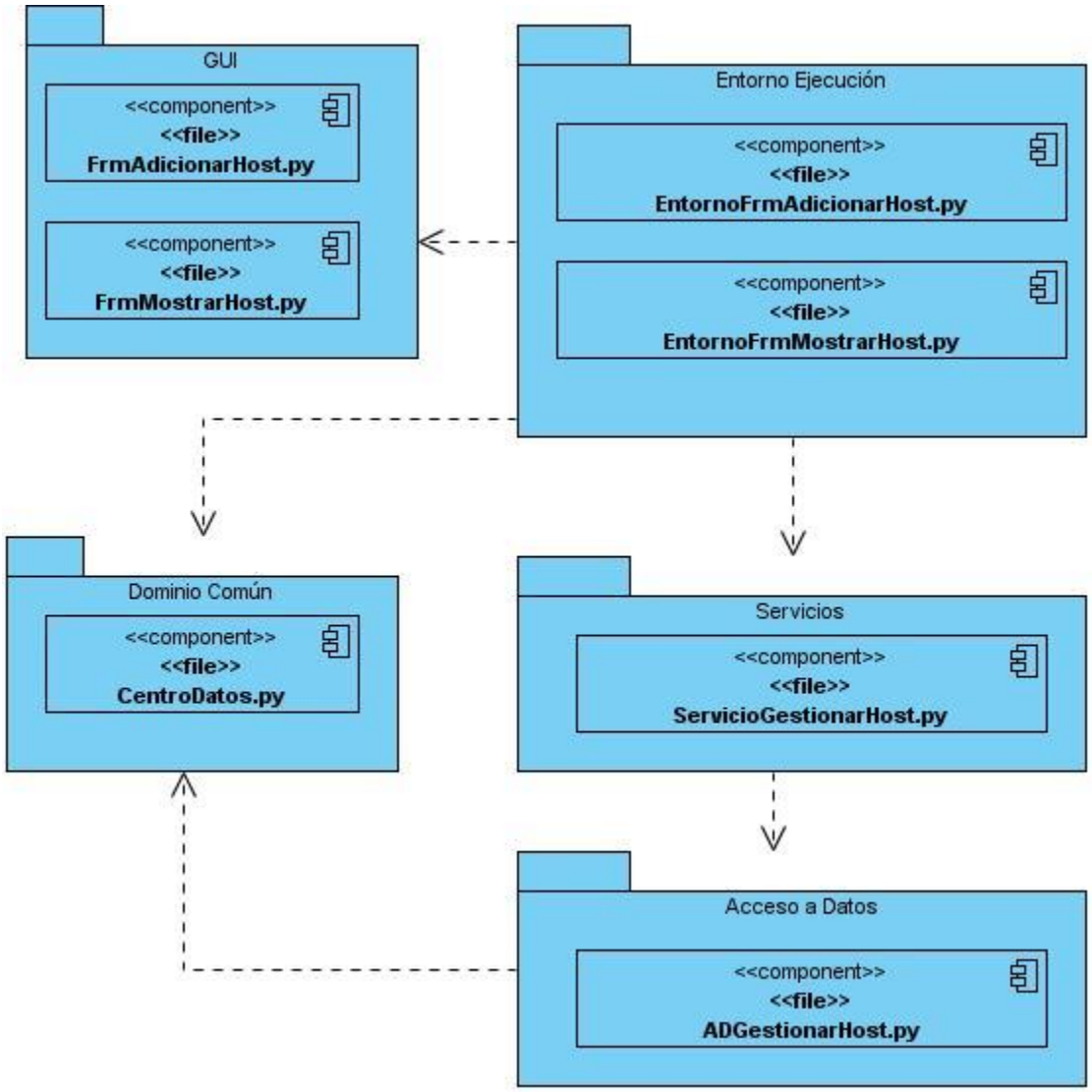


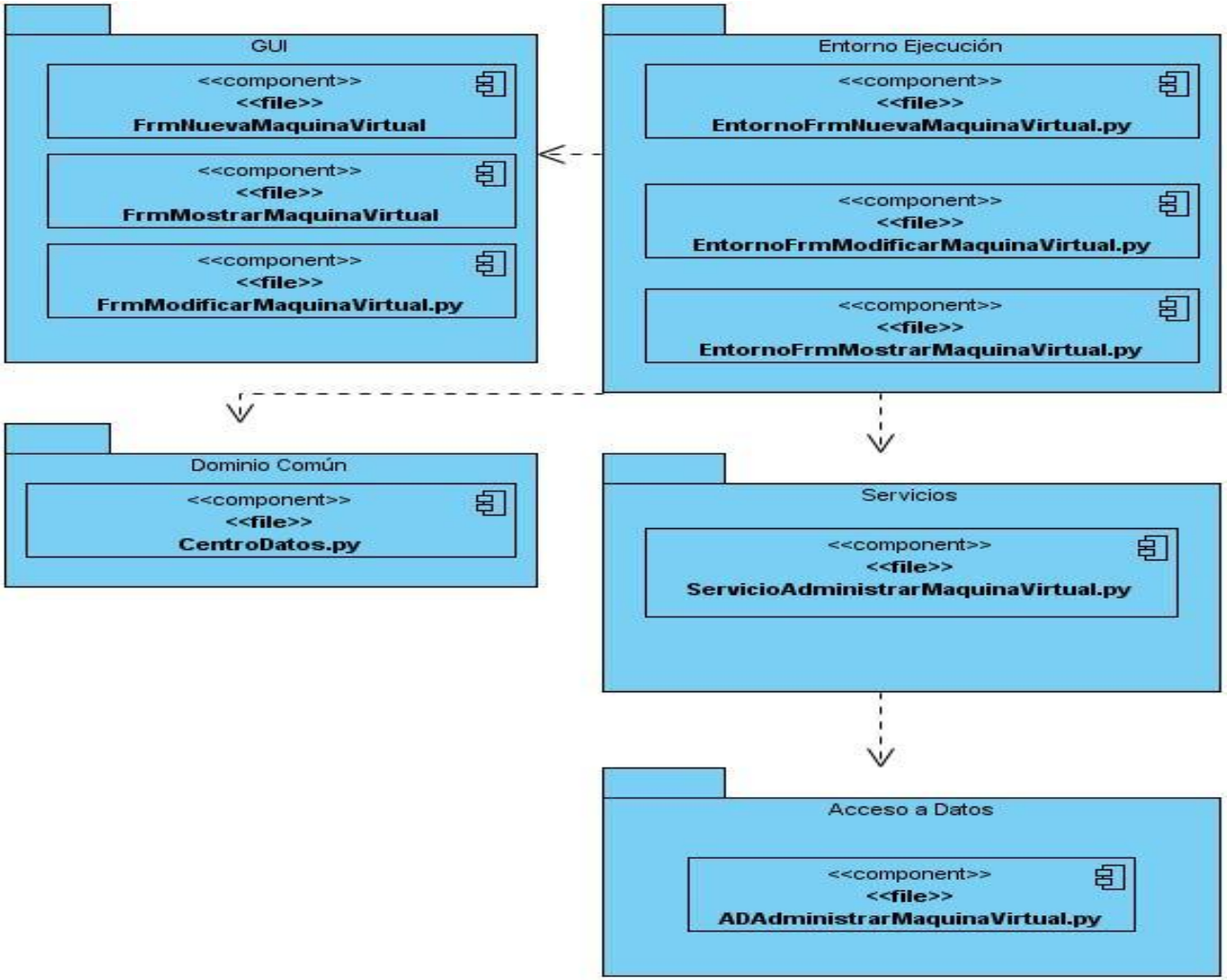
Tabla 5: Diagrama de Componentes. CU Gestionar Host


GUI
<<component>>
<<file>>
FrmAdicionarHost.py
<<component>>
<<file>>
FrmMostrarHost.py
Entorno Ejecución
<<component>>
<<file>>
EntornoFrmAdicionarHost.py
<<component>>
<<file>>
EntornoFrmMostrarHost.py
Dominio Común
<<component>>
<<file>>
CentroDatos.py
Servicios
<<component>>
<<file>>
ServicioGestionarHost.py
Acceso a Datos
<<component>>
<<file>>
ADGestionarHost.py

Descripción de los componentes del CU Gestionar Host:

El componente FrmAdicionarHost se encarga de la todo lo relacionado con la interfaz de la funcionalidad adicionar host, es donde el administrador introduce los datos del nuevo host y mediante el EntornoFrmAdicionarHost se comunica con el ServicioAdmininstrarHost y este a su vez con el ADAdmininstrarHost donde el nuevo host es adicionado a un lista del CentroDatos. El componente FrmMostrarHost se encarga de mostrar todos los datos del host, los cuales se obtienen a través del EntornoFrmMostrarHost utilizando al ServicioAdmininstrarHost quien realiza la petición al ADAdmininstrarHost el cual es el encargado de obtener los datos del host a mostrar utilizando la lista de host del CentroDatos.

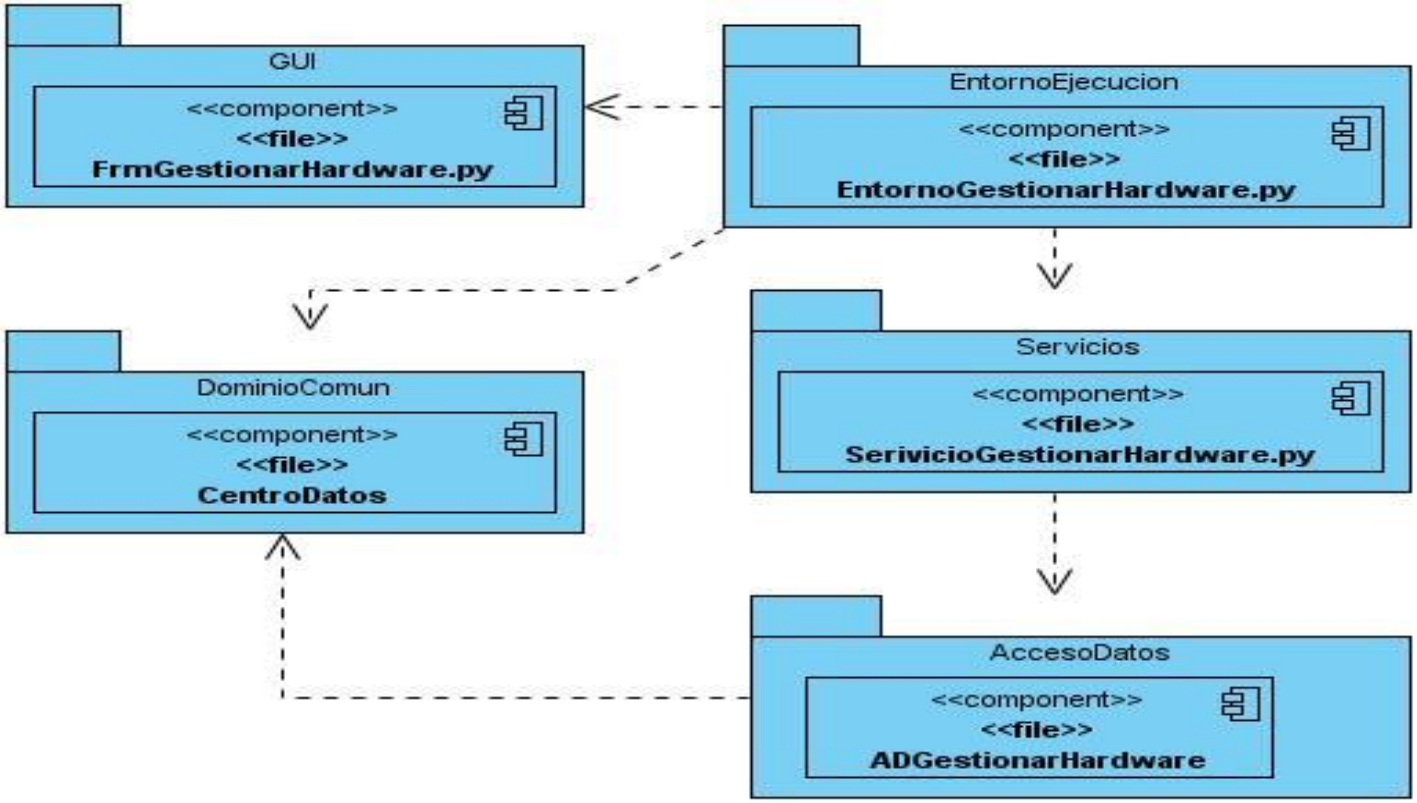
Tabla 6: Diagrama de Componentes. CU Administrar Máquina Virtual



Descripción de los componentes del CU Administrar Máquina Virtual:

El componente FrmNuevaMaquinaVirtual se utiliza para obtener los datos introducidos por el administrador de la nueva máquina virtual, los cuales son enviados al ServicioAdministrarMaquinaVirtual a través del EntornoFrmNuevaMaquinaVirtual y el servicio al ADAdministrarMaquinaVirtual donde se adiciona a un host de la lista de host del CentroDatos. FrmMostrarMaquinaVirtual es el componente que brinda la interfaz para visualizar los datos de una máquina virtual, este usa al EntornoFrmNuevaMaquinaVirtual para llamar al ServicioAdminstrarMaquinaVirtual quien se encarga de la lógica y utiliza el ADAdministrarMaquinaVirtual para obtener los datos del host que contiene a la máquina virtual.

Tabla 7: Diagrama de Componentes. CU Gestionar Hardware



4.4 Prueba

La calidad del producto se obtiene de diferentes formas, una de estas son las pruebas de software, con lo cual se asegura que este cumpla con todas sus funcionalidades. El primer paso para la realización de las pruebas es escoger una Estrategia de Prueba, la cuales incluyen 3 niveles de prueba, Unidad, Integración y Sistema, cada nivel posee sus tipos de prueba y a su vez estos los

métodos que pueden ser de Caja Negra o Caja Blanca. Las pruebas de Caja blanca de realiza mediante la técnica del Camino Básico y las de Caja Negra mediante la técnica de Partición de Equivalencia.

Se escogió como estrategia hacer las pruebas utilizando el nivel de Integración, dentro de este nivel el tipo de prueba de Funcionalidad–Función mediante el método de caja negra y la técnica de Partición de Equivalencia.

Integración

Es ejecutada para asegurar que los componentes en el modelo de implementación operen correctamente cuando son combinados para ejecutar un caso de uso. Se prueba un paquete o un conjunto de paquetes del modelo de implementación. Estas pruebas descubren errores en las especificaciones de las interfaces de los paquetes. Esta prueba debe ser responsabilidad de desarrolladores y de independientes, sin solaparse las pruebas.
Es el proceso de combinar y probar múltiples componentes juntos. El objetivo es tomar los componentes probados en unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño.
Se llama integración incremental cuando el programa se construye y se prueba en pequeños segmentos en los que los errores son más fáciles de aislar y corregir, es más probable que se pueda probar completamente las interfaces y se puede aplicar un enfoque de prueba sistemática.

Tipos de pruebas

- Funcionalidad
  - Función

    Pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso.

Métodos de pruebas

- Caja Negra

La prueba de caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del software. O sea, los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, así como que la integridad de la información externa se mantiene.

- Técnicas de Caja Negra

  Partición equivalente es una técnica de prueba de Caja Negra que divide el dominio de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. El diseño de estos casos de prueba para la partición equivalente se basa en la evaluación de las clases de equivalencia.

- Sección Iniciar Máquina Virtual

Escenario	**Descripción**	**Respuesta del sistema**	**Resultado de la prueba**	**Flujo central**
EC 1.0 Iniciar Máquina Virtual satisfactoriamente.	Inicia una Máquina Virtual detenida o que no esté activa	Inicia la Máquina Virtual.	Satisfactoria	El administrador selecciona una Máquina Virtual que su estado esté detenida o no esté activa y oprime el botón “Iniciar”.

- Sección Apagar Máquina Virtual

Escenario	**Descripción**	**Respuesta del sistema**	**Resultado de la prueba**	**Flujo central**
EC 2.0 Apagar *Máquina Virtual satisfactoriamente.*	Apaga una Máquina Virtual que se encuentre activa.	Apaga la Máquina Virtual.	Satisfactoria	El administrador selecciona una Máquina Virtual esté activa y oprime el botón “Apagar”.

- Sección Nueva Máquina Virtual

Escenario	Descripción	Nombre	RAM	Nombre Disco Duro	Tamaño Disco Duro	Respuesta del sistema	Resultado de la prueba	Flujo central
EC3.0 Crear Máquina Virtual Satisfactoriamente	Adiciona una nueva Máquina Virtual a la lista de máquinas Virtuales	*V vmdebian4*	*V 256*	*V Vmdebian4*	*V 5*	Adiciona una máquina Virtual con las características introducidas y muestra un mensaje de confirmación.	Satisfactoria	El Administrador selecciona en el menú la opción Máquinas Virtuales y dentro de esta: Nueva Máquina Virtual.
EC3.1 Crear Máquina Virtual con Datos Incorrectos	Adiciona una nueva Máquina Virtual a la lista de máquinas Virtuales con datos incorrectos.	*V Vmdebina5*	*V 256*	*I Vmde+**	*V 7*	El sistema lanza un error de campo inválido	Satisfactoria	
		V Maquina1	*V 512*	*V Maquina 1*	*I letras*	El sistema no permite introducir letras en el campo tamaño del disco duro.	Satisfactoria	
		*I Vmubuntu***	*V 51*	*V vmubuntu*	*V 6*	El sistema lanza un error de campo con	Insastifacoria	

			2			caracteres no válidos		
		V vmdebian	*I letras*	*V vmdebian*	*V 10*	El sistema no deja introducir letras en el campo RAM.	Satisfactoria	
EC3.2 Crear Máquina Virtual con datos en blanco	Adiciona una nueva Máquina Virtual a la lisa de máquinas Virtuales	*I*	*V 512*	*V vmdebian*	*V 8*	El sistema lanza un error “No pueden existis campos vacíos”	Satisfactoria	
		V vmdebian	*I*	*V Vmdebian3*	*V 6*	El sistema lanza un error “no pueden existir campos vacíos”	Satisfactoria	
		V prueba	*V 256*	*I*	*V 6*	El sistema lanza un error “no pueden existir campos vacíos”	Satisfactoria	
		V Prueba 2	*V 312*	*V Prueba2*	*I*	El sistema lanza un error “no pueden existir campos vacíos”	Satisfactoria	

Resultado Esperado

Se realizó el caso de prueba al caso de uso Administrar Máquina Virtual obteniéndose los siguientes resultados.

Cantidad de Casos de Prueba	**Cantidad de No Conformidades**	**Cantidad que Proceden**	**Cantidad que No Proceden**
1	**1**	**1**	**0**

4.5 Conclusiones

En el capítulo fueron presentados los artefactos fundamentales correspondientes a la Implementación del Sistema, entre los que se destacan el modelo de despliegue que deja clara la arquitectura en términos de nodos o computadoras y, el diagrama general de componentes del sistema y los diagramas de componentes de cada Caso de Uso, donde se puede identificar todas las clases empleadas para realizar las funcionalidades planteadas en la aplicación. También se realizaron las pruebas mediante el Método de Caja Negra obteniendo no conformidades, las cuales se resolvieron y ayudaron a incrementar la calidad de la aplicación.

CONCLUSIONES

Se llevó a cabo una investigación sobre la tecnología de virtualización y las herramientas que permiten la gestión de máquinas virtuales, permitiendo utilizar las herramientas adecuadas para la construcción de un software de alto nivel que ayuda a aumentar el nivel de la Plataforma de Gestión de Servicios Telemáticos en GNU\Linux. Se logró una aplicación capaz de administrar máquinas virtuales remotamente de manera distribuida en diferentes host, mostrando una información detallada de estas y de los host donde se virtualiza, además la aplicación desarrollada permite trabajar de una forma intuitiva y fácil.

RECOMENDACIONES

Se recomienda seguir ampliando las funcionalidades de la aplicación:

- Hacer clúster de balanceo de carga entre los host, es decir que automáticamente asigne recursos de un host determinado a una máquina virtual que lo requiera.
- Adicionar tabla de eventos, la cual notificará todos los eventos que ocurran en la aplicación, brindando información constante de todas las actividades o fallos en las máquinas virtuales y host que gestiona la aplicación
- Extender la aplicación para otros hypervisores como Xen.
- Implementar el migrar de las máquinas virtuales de un host al otro, permitiendo hacer copias de las mismas rápidamente en caso de algún fallo o necesidad.

RERENCIA BIBLIOGRÁFICA

1. consultaunitpro.com. *historia-de-la-virtualizacion.* [En línea] [Citado el: 10 de 2 de 2011.] http://www.consultaunitpro.com/tag/historia-de-la-virtualizacion.

2. vmware.com. *history.* [En línea] [Citado el: 10 de 2 de 2011.] http://www.vmware.com/es/virtualization/history.html.

3. microsoft.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.microsoft.com/windowsserver2008/en/us/product-information.aspx.

4. suite101.net. *comprendiendo-kernel-virtual-machine-kvm.* [En línea] [Citado el: 10 de 2 de 2011.] www.suite101.net/content/comprendiendo-kernel-virtual-machine-kvm-a30276.

5. virtualizacion.com. [En línea] [Citado el: 15 de 4 de 2010.] http://www.virtualizacion.com/?page_id=7.

6. microsoft.com. *Hyper-v.* [En línea] [Citado el: 10 de 12 de 2010.] http://www.microsoft.com/spain/windowsserver2008/virtualization/hyperv_intro.mspx.

7. vmware.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.vmware.com.

8. Vmware.com. *Workstation.* [En línea] [Citado el: 10 de 12 de 2010.] http://www.vmware.com/products/workstation.

9. vmware.com. *Server.* [En línea] [Citado el: 10 de 12 de 2010.] http://www.vmware.com/products/server.

10. virtualbox.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.virtualbox.org.

11. xen.org. [En línea] [Citado el: 10 de 12 de 2010.] http:// www.xen.org.

12. linux-kvm.org. *Main_Page.* [En línea] [Citado el: 10 de 2 de 2011.] http://www.linux-kvm.org/page/Main_Page.

13. code.google.com. [En línea] [Citado el: 10 de 12 de 2010.] http://code.google.com/p/ganeti.

14. code.google.com. [En línea] [Citado el: 10 de 12 de 2010.] http://code.google.com/p/virttool.

15. libvirt.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.libvirt.org.

16. ibm.com. *l-libvirt.* [En línea] [Citado el: 10 de 2 de 2011.] http://www.ibm.com/developerworks/linux/library/l-libvirt/index.html.

17. opennebula.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.opennebula.org/about:about.

18. **Padilla Molla, Alvaro Luis y Pérez Hurtado, Alexei.** *Plataforma de Gestión de Servicios Telemáticos en GNU/Linux. Módulo DNS v2.0.* Ciudad de la Habana : s.n.

19. **de la Rosa Pasteur, Jeni y Zulueta Sarria, Edgardo.** *, Plataforma de Gestión de Servicios Telemáticos en GNU/Linux.Módulo de Directorio v2.0.* Ciudad de la Habana : s.n., 2010.

20. idef.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.idef.com/idef0.htm.

21. itl.nist.gov. [En línea] [Citado el: 10 de 12 de 2010.] http://www.itl.nist.gov/fipspubs/idef02.doc.

22. omg.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.omg.org/gettingstarted/what_is_uml.htm.

23. python.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.python.org/doc.

24. visual-paradigm.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.visual-paradigm.com/aboutus.

25. easyeclipse.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.easyeclipse.org/site/about/index.html.

26. di-mare.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.di-mare.com/adolfo/cursos/2007-1/pp-GenGUI.pdf.

27. revistauxi.files.wordpress.com. [En línea] [Citado el: 10 de 12 de 2010.] http://revistauxi.files.wordpress.com/2009/01/uxi08_v02.pdf.

28. **Larman, Craig.** *UML Y PATRONES.*

29. dei.uc.edu.py. [En línea] [Citado el: 10 de 12 de 2010.] http://www.dei.uc.edu.py/tai2002/SD/discom.htm.

30. Vmware.com. *Player.* [En línea] http://www.vmware.com/products/player.

31. Vmware.com. *Vsphere.* [En línea] [Citado el: 10 de 12 de 2010.] http://www.vmware.com/products/vsphere.

32. rational.com.ar. [En línea] [Citado el: 10 de 12 de 2010.] http://www.rational.com.ar/index.html.

33. elai.upm.es. [En línea] [Citado el: 10 de 12 de 2010.] http://www.elai.upm.es/spain/Investiga/GCII/personal/vcorte/informeqt.PDF.

BIBLIOGRAFÍA

1. consultaunitpro.com. *historia-de-la-virtualizacion.* [En línea] [Citado el: 10 de 2 de 2011.] http://www.consultaunitpro.com/tag/historia-de-la-virtualizacion.

2. vmware.com. *history.* [En línea] [Citado el: 10 de 2 de 2011.] http://www.vmware.com/es/virtualization/history.html.

3. microsoft.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.microsoft.com/windowsserver2008/en/us/product-information.aspx.

4. suite101.net. *comprendiendo-kernel-virtual-machine-kvm.* [En línea] [Citado el: 10 de 2 de 2011.] www.suite101.net/content/comprendiendo-kernel-virtual-machine-kvm-a30276.

5. virtualizacion.com. [En línea] [Citado el: 15 de 4 de 2010.] http://www.virtualizacion.com/?page_id=7.

6. microsoft.com. *Hyper-v.* [En línea] [Citado el: 10 de 12 de 2010.] http://www.microsoft.com/spain/windowsserver2008/virtualization/hyperv_intro.mspx.

7. vmware.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.vmware.com.

8. Vmware.com. *Workstation.* [En línea] [Citado el: 10 de 12 de 2010.] http://www.vmware.com/products/workstation.

9. vmware.com. *Server.* [En línea] [Citado el: 10 de 12 de 2010.] http://www.vmware.com/products/server.

10. virtualbox.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.virtualbox.org.

11. xen.org. [En línea] [Citado el: 10 de 12 de 2010.] http:// www.xen.org.

12. linux-kvm.org. *Main_Page.* [En línea] [Citado el: 10 de 2 de 2011.] http://www.linux-kvm.org/page/Main_Page.

13. code.google.com. [En línea] [Citado el: 10 de 12 de 2010.] http://code.google.com/p/ganeti.

14. code.google.com. [En línea] [Citado el: 10 de 12 de 2010.] http://code.google.com/p/virttool.

15. libvirt.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.libvirt.org.

16. ibm.com. *l-libvirt.* [En línea] [Citado el: 10 de 2 de 2011.] http://www.ibm.com/developerworks/linux/library/l-libvirt/index.html.

17. opennebula.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.opennebula.org/about:about.

18. **Padilla Molla, Alvaro Luis y Pérez Hurtado, Alexei.** *Plataforma de Gestión de Servicios Telemáticos en GNU/Linux. Módulo DNS v2.0.* Ciudad de la Habana : s.n.

19. **de la Rosa Pasteur, Jeni y Zulueta Sarria, Edgardo.** *, Plataforma de Gestión de Servicios Telemáticos en GNU/Linux.Módulo de Directorio v2.0.* Ciudad de la Habana : s.n., 2010.

20. idef.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.idef.com/idef0.htm.

21. itl.nist.gov. [En línea] [Citado el: 10 de 12 de 2010.] http://www.itl.nist.gov/fipspubs/idef02.doc.

22. omg.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.omg.org/gettingstarted/what_is_uml.htm.

23. python.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.python.org/doc.

24. visual-paradigm.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.visual-paradigm.com/aboutus.

25. easyeclipse.org. [En línea] [Citado el: 10 de 12 de 2010.] http://www.easyeclipse.org/site/about/index.html.

26. di-mare.com. [En línea] [Citado el: 10 de 12 de 2010.] http://www.di-mare.com/adolfo/cursos/2007-1/pp-GenGUI.pdf.

27. revistauxi.files.wordpress.com. [En línea] [Citado el: 10 de 12 de 2010.] http://revistauxi.files.wordpress.com/2009/01/uxi08_v02.pdf.

28. **Larman, Craig.** *UML Y PATRONES.*

29. dei.uc.edu.py. [En línea] [Citado el: 10 de 12 de 2010.] http://www.dei.uc.edu.py/tai2002/SD/discom.htm.

30. Vmware.com. *Player.* [En línea] http://www.vmware.com/products/player.

31. Vmware.com. *Vsphere.* [En línea] [Citado el: 10 de 12 de 2010.] http://www.vmware.com/products/vsphere.

32. rational.com.ar. [En línea] [Citado el: 10 de 12 de 2010.] http://www.rational.com.ar/index.html.

33. elai.upm.es. [En línea] [Citado el: 10 de 12 de 2010.] http://www.elai.upm.es/spain/Investiga/GCII/personal/vcorte/informeqt.PDF.

GLOSARIO

"[Insertar glosario]"

ANEXOS

"[Insertar anexos]"

1