Universidad de las Ciencias Informáticas
Facultad 1

Título: Módulo para la administración de NAS en Nova para servidores

---

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor:

Eugenio Rosales Rosa

Tutor:

Ing. Abel Alfonso Fírvida Donestévez

La Habana

2012

Declaración de la 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 firmamos la presente a los ____ días del mes de _____ del año ____

Eugenio Rosales Rosa

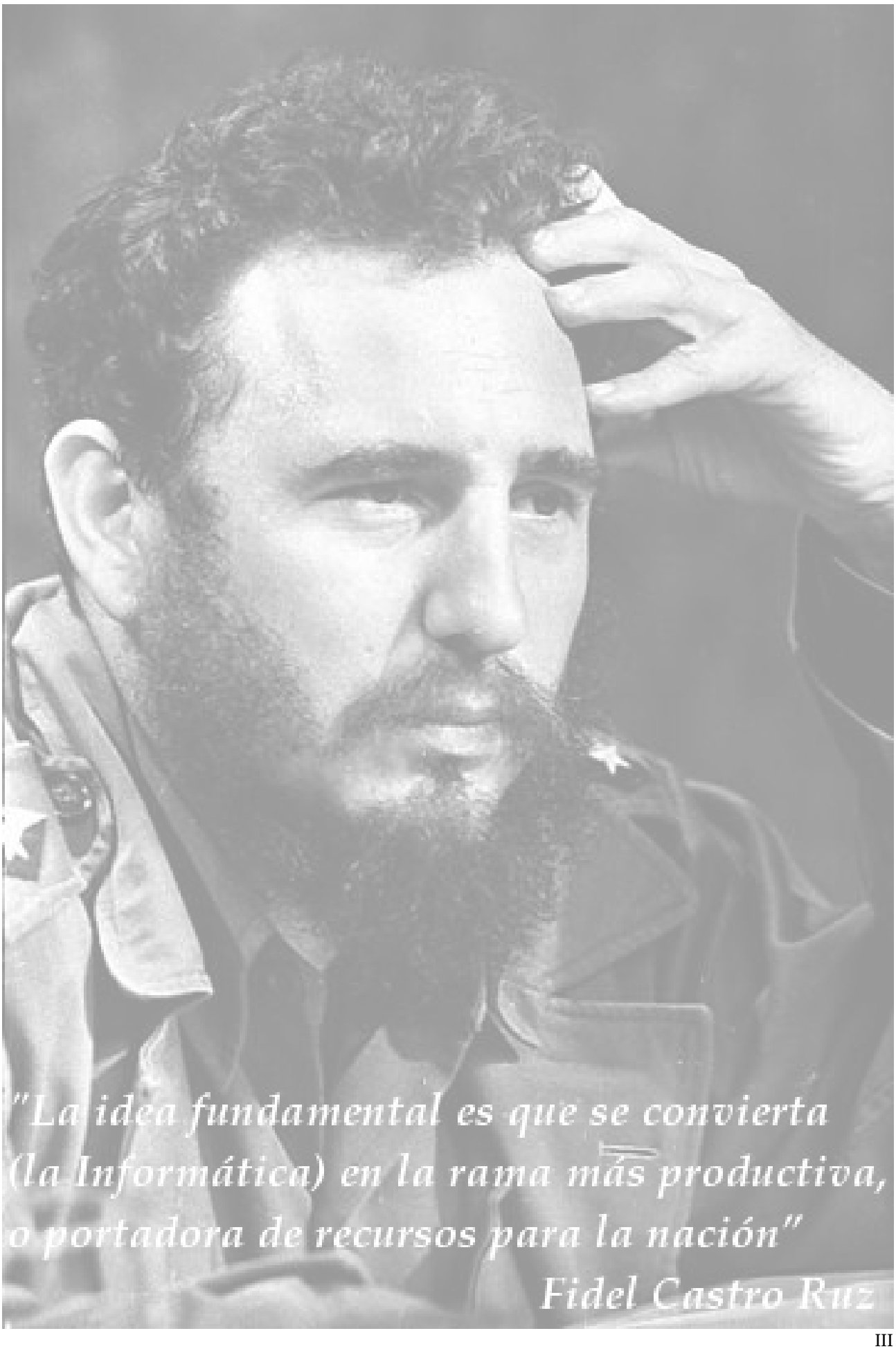
_____

Firma del Autor

Ing. Abel Alfonso Fírvida Donestévez

_____

Firma del Tutor


"La idea fundamental es que se convierta
(la Informática) en la rama más productiva,
o portadora de recursos para la nación"
Fidel Castro Ruz

DEDICATORIA

A mi abuela Estelvina por acompañarme todos los días de mi vida a pesar de no poder contar con su presencia física.

A mis padres por darme toda la confianza del mundo, por creer en mí, por haber permitido que sea lo que hoy soy.

A mis hermanitos por darme tanta felicidad con sus sonrisas inocentes.

A mi princesa Yani, por aguantar mis malacrianzas y haber estado conmigo en los mejores y más difíciles momentos de mi carrera.

Al concluir esta etapa de mi vida, quiero hacer constar mi más profundo agradecimiento a todas las personas que con su apoyo, comprensión y ayuda hicieron posible lograr lo que soy.

A Abel, un excelentísimo tutor a quien le estaré siempre agradecido y por quien estaré en deuda eterna por su aporte desinteresado a este trabajo.

A mis padres, por ser todo para mí, por haberme convertido en un hombre dichoso de tenerlos para contar con con su comprensión, consejos y amor. Gracias por la educación que me dieron.

A mis hermanos, abuelos, tíos, primos, por sus muestras de cariño y admiración hacia mí, porque también han sido imprescindibles para la realización de este sueño.

A mi niña Yani, por todo el amor que me ha dado, por saberme convertido en un hombre feliz, por entregar su vida a mí, por darlo todo tan solo por amor. Gracias por estar en mi vida.

A Yulién por ser una excelente persona, por ser mi segunda madre.

A Edy, por ser un digno ejemplo a seguir, por ser un hombre admirable, por ser como ha sido conmigo, como un segundo padre.

A mis suegros por haberme dado la oportunidad de tener una novia tan linda y por ser personas maravillosas de los que me siento orgulloso.

A la profe Matilde por haberme enseñado de sí lo mejor de una persona, por su cariño, su bondad, su amor.

A mi grupo de siempre, el a, porque creamos una familia, un grupo de amigos inseparables, por comportarse como hermanos.

A la Revolución y a nuestro Comandante en Jefe Fidel Castro, por darme la posibilidad de estudiar en la mejor y más bonita escuela del país, tan solo para realizar mi propio sueño de convertirme en un profesional.

A todas mi amistades, a las personas que me alegraron la vida y las que no porque me enseñaron a vivir.

Gracias a mi Señor por estar presente en los momentos en que más desobediente fui.

Sinceramente, muchas gracias.

Resumen

El presente trabajo de diploma titulado “Módulo para la administración de NAS en Nova para servidores” contiene un estudio realizado sobre las principales aplicaciones enfocados a soluciones de almacenamiento en red. Además se describen y seleccionan una serie de herramientas, así como la metodología empleada para el desarrollo de la solución.

Se realiza una planificación y definición de la aplicación, la cual consiste en el desarrollo de una variante para Nova para Servidores orientada al almacenamiento de recursos en red, y se realizan las pruebas correspondientes al sistema.

Palabras Clave: Módulo, administración, NAS, Nova para servidores, aplicaciones, almacenamiento, red, herramientas, metodología, pruebas.

Índice

Introducción

El hombre ha creado su propia historia, imponiéndose retos más grandes a medida que ha ido evolucionado en el tiempo. No se puede decir exactamente cuándo fue que se hizo la primera cuenta, pero sí se conocen algunos de los primeros dispositivos mecánicos para su uso como lo son el ábaco y la Pascalina. Si embargo el desarrollo de las matemáticas y la imperiosa necesidad de realizar cálculos demasiados complejos condujo a la creación de equipos que hicieran menos tedioso este trabajo.

El surgimiento de la computadora marcó una nueva era en el desarrollo de la humanidad. En principio su uso estuvo basado en la realización de cálculos, para determinar ciertos resultados que se hacían muy engorrosos hacerlo manualmente por la complejidad que estos demandaban, además de ser una vía de facilitar este tipo de trabajo como necesidad social. Pero con el la evolución de los sistemas de computación se fueron añadiendo características a los ordenadores en cuanto al diseño, el tamaño y otros factores que hacen posible llamar la atención al usuario, a causa de la creación de la primera computadora comercial, trayendo consigo la competencia mundial en este tipo de temas en la quinta y presente generación.

Existe una tendencia actual de los fabricantes de *hardware* a hacer computadoras con características más acordes a los deseos de los usuarios ante la demanda mundial que estos exigen. Este fenómeno se expresa directamente en programas que demandan un procesamiento en paralelo mediante arquitectura y diseños y circuitos que proporcionan gran velocidad, así como capacidad de almacenamiento. Otros aspectos pueden ser relacionados pero resulta muy interesante lo previsible que resulta el futuro de la informática, que seguirá siendo objeto de atención por la sociedad.

Con la creciente cantidad de información almacenada y por la necesidad de tener disponibles miles de datos han surgido varias soluciones de almacenamiento; las redes **SAN** (por sus siglas en inglés de *Storage Area Network*) y los sistemas **NAS** (*Network Attached Storage*) se destacan en este aspecto. Estos permiten mejorar el aprovechamiento de la memoria y los discos duros y compartir la capacidad de almacenamiento de una computadora, respectivamente.

Estas aplicaciones ponen a disposición del usuario una memoria de alta velocidad con amplias capacidades. Es flexible y sin complicaciones. En una infraestructura típica, el servidor NAS se conecta a la red del área local (LAN). Aunque tanto estos sistemas NAS como los SAN separan la memoria de cada servidor, los sistemas NAS están unidos a la red local, a diferencia de los SAN. Gracias a esta disposición, los sistemas NAS permiten la rápida ampliación de la memoria necesaria sin modificar la arquitectura existente de cliente/servidor. Cuba es un país que presenta dificultades económicas notables y se hace necesario tomar medidas para no ceder ante las tentaciones que el desarrollo tecnológico expone y hacer más eficientes los recursos con los que

se cuenta. Con ese fin en la Universidad de las Ciencias Informáticas se creó el sistema operativo Nova, con soporte especializado proveniente de entidades cubanas, garantizado sobre la soberanía tecnológica y la seguridad nacional en el ámbito de la informática y las telecomunicaciones.

Con vistas a que **GEDEME**[1] planea distribuir servidores de almacenamiento, para ello es necesario adicionar soporte a Nova para la implementación cómoda de servicios de almacenamiento en red vía NAS. Pero en estos momentos la versión de Nova para Servidores no permite implementar este servicio, lo cual es **situación problemática**.

Por lo antes expuesto se define como **problema científico:**

¿Cómo lograr una variante de Nova para servidores con soporte para implementar un servicio NAS?

El **objeto de estudio** lo constituyen los sistemas operativos orientados a servidores de almacenamiento en red.

El **campo de acción** está enmarcado en las herramientas para la configuración del servicio NAS en Nova.

Para realizar la investigación se trazó como **objetivo general:**

Obtener una versión de Nova para servidores con soporte para implementar un servicio NAS.

Para dar cumplimiento al objetivo general se han definido los siguientes **objetivos específicos:**

- Estudiar los principios de funcionamiento del servicio NAS para su implementación.
- Obtener una aplicación que facilite la implementación del servicio NAS.
- Validar la solución propuesta.
- Documentar el proceso de construcción de un módulo de configuración de NAS para la plataforma Zentyal.

El autor pretende obtener como **posible resultado** de este trabajo una variante de la distribución cubana de GNU\Linux Nova para servidores garantizando soporte para implementar un servicio NAS rápido y estable y que el desarrollo de la misma pueda ser guiado y auditado por especialistas nacionales.

Para el cumplimiento de los objetivos descritos anteriormente se proponen las siguientes **tareas de investigación:**

- Estudio del estado del arte de la reutilización de los requisitos necesarios para la implementación de la herramienta con soporte para servicio NAS.

[1] *Gente de Mérito (**GEDEME**), Empresa Cubana de Producción de Medios Técnicos de Computación.*

- Estudio de los sistemas que existen para el almacenamiento y compartición de recursos en red, tanto como propietarios como libres, para analizar sus características, ventajas e inconvenientes y poder adoptar una posición al respecto.
- Levantamiento de los requisitos necesarios para la construcción de la aplicación.
- Realización del análisis y el diseño de la herramienta.
- Implementación de los requerimientos a partir del análisis y el diseño antes desarrollado.
- Realización de las pruebas de funcionalidad de la herramienta para comprobar su correcto funcionamiento.

La **idea que se defiende** es que con el desarrollo de la variante de la distribución Nova para servidores se podría garantizar una versión que permita el soporte de un servicio NAS.

Para solucionar las tareas antes mencionadas se utilizaron los siguientes **métodos científicos**:

Métodos teóricos

Histórico-Lógico: Mediante este método se realizó una revisión histórica y análisis de la trayectoria del desarrollo de aplicaciones similares, para tomar una posición al respecto en cuanto a características y funcionalidades.

Inductivo-Deductivo: Este método fue utilizado con la finalidad de retroalimentarse sobre el funcionamiento de la herramienta y a partir de ahí llegar a conclusiones sobre cómo lograr su construcción.

Método empírico

Observación:

Esta técnica permitió obtener un registro visual del comportamiento de algunas herramientas que auxilian el desarrollo de esta aplicación.

El presente trabajo de diploma está estructurado en tres capítulos, los cuales se describen a continuación:

Capítulo 1. Fundamentación teórica de la aplicación: Se procede a realizar un estudio del estado del arte referente al tema que se aborda. De esto se deriva el estudio e investigación sobre los principales sistemas operativos que soportan servicio NAS, así como la metodología a seguir, las herramientas, servicios y servidores necesarios para el desarrollo de la aplicación propuesta.

Capítulo 2. Análisis y diseño de la aplicación: Se realiza un análisis de los componentes existentes para su utilización, además de hacer la representación el modelo de clases del diseño.

Se reflejan los elementos y funcionalidades que poseerá el sistema siguiendo la metodología necesaria. Se detalla la propuesta de solución.

Capítulo 3. Implementación y realización de pruebas: Se muestran los modelos de datos e implementación con sus respectivos diagramas de clases persistentes, de componentes y estándares de codificación a emplear en la solución de la propuesta. Además, se realizan las pruebas de aceptación de las historias de usuario correspondientes a cada iteración durante el desarrollo de la herramienta propuesta y se muestran los resultados esperados de las mismas.

Capítulo 1: Fundamentación teórica de la aplicación

1. Introducción

En este capítulo se abordará una serie de elementos que son fundamentales para la comprensión de las etapas del proceso de construcción de la aplicación que se propone. Se definirán algunos conceptos y características que servirán de apoyo para el conocimiento del tema. Asimismo, se plasmará la metodología de desarrollo y se hará referencia al lenguaje de programación a utilizarse, además de otros aspectos de interés.

1.1 SAN y NAS como administradores del almacenamiento

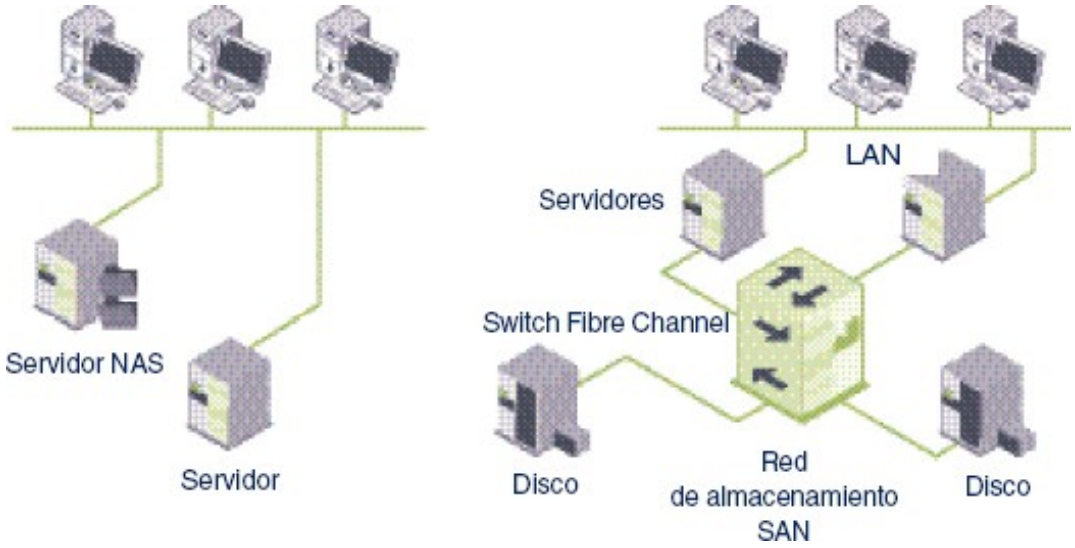


Figura 1. Almacenamiento NAS y SAN.

SAN y NAS son las palabras clave que actualmente predominan en el sector del almacenamiento de las Tecnologías de la Información (TI). Sin embargo, en muchos casos no queda claro el significado y las ventajas de estos conceptos de almacenamiento tan distintos.

1.1.1 Redes SAN

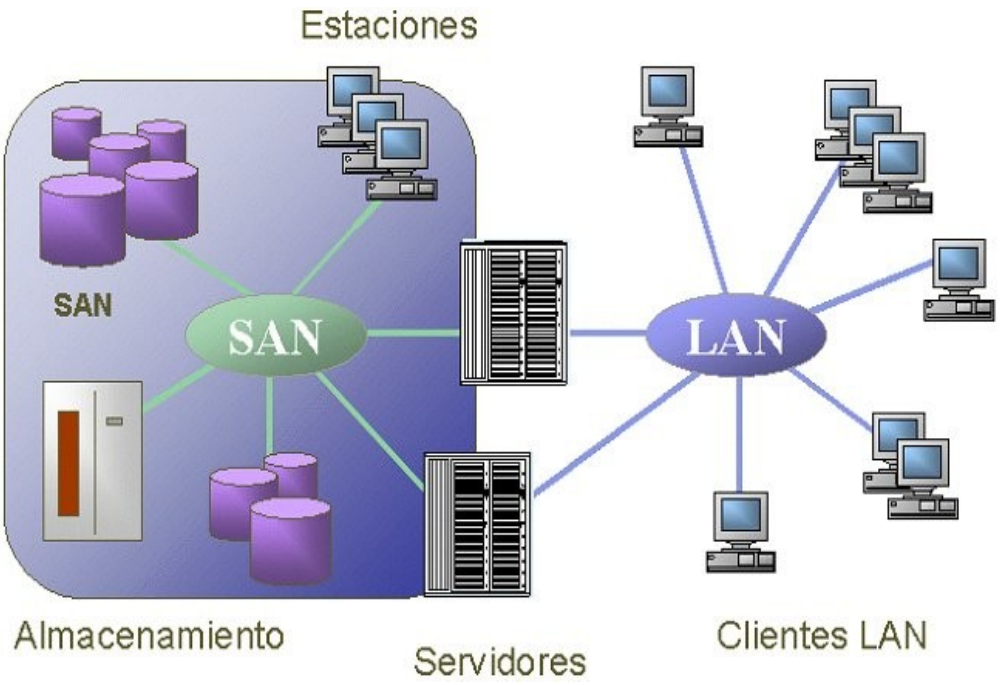


Figura 2. Redes SAN.

Una **SAN** es una red de alta velocidad diseñada especialmente para el almacenamiento de datos y que está conectada a uno o más servidores a través de fibra. Los usuarios pueden acceder a cualquiera de los dispositivos de almacenamiento de la red a través de los servidores, y los datos son escalables hasta 50TBs. El almacenamiento de datos centralizado reduce la administración necesaria y proporciona un tipo de almacenamiento de alto rendimiento y flexible para entornos multiservidor/multivendedor.

El SAN es un sistema de discos que se conecta a los servidores mediante redes de altísima velocidad (generalmente *fibre channel*[2]). Se suele usar en servidores de BBDD[3] donde los *filesystems* no se pueden montar por NFS o SAMBA y se necesita una comunicación rápida.

Las soluciones SAN permiten mejorar el aprovechamiento de la memoria y los discos duros en unos dos tercios. De esta forma, pueden reducirse los gastos de hardware y mantenimiento, así como los costes de las licencias de software y la electricidad. Además, una solución SAN ofrece un rendimiento mayor de lo habitual. Pero esto no es todo. La principal ventaja de un sistema SAN es que ofrece una administración mucho más sencilla. La consolidación reduce los costes de personal aunque aumente el número de recursos de memoria pues en estos sistemas las capacidades de memoria se administran centralmente. Es una característica muy atractiva si tiene en cuenta que más de la mitad de los costes derivados de la administración de memoria son precisamente de personal. Un argumento convincente para la implantación de una solución SAN.

[2]***Fibre channel:*** *es el equivalente en español a canal de fibra, tecnología de red utilizada principalmente para redes de almacenamiento.*
[3]***BBDD:*** *abreviatura referente a base de datos, pero en número plural, o sea, bases de datos.*

Ventajas de la consolidación mediante SAN

- Reducción del coste total de propiedad.
- Reducción de los costes administrativos hasta un 50 %.
- Reducción de los costes de personal.
- No se carga la red de área local (LAN).

1.1.2 Redes NAS

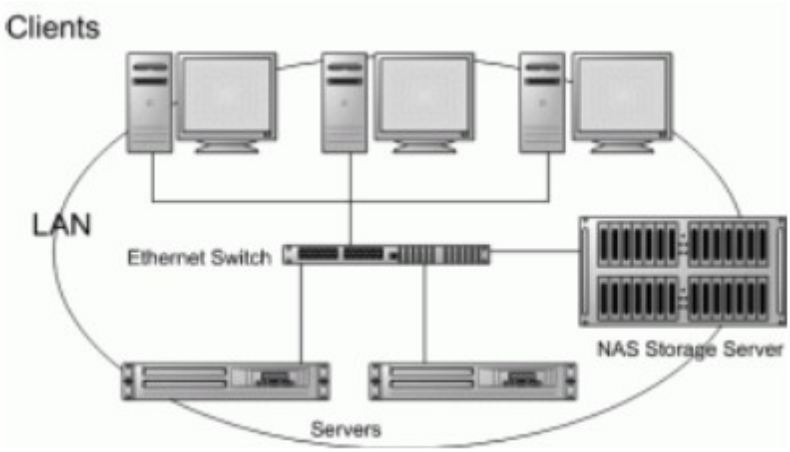


Figura 3. Redes NAS.

NAS es el nombre dado a una tecnología de almacenamiento dedicada a compartir la capacidad de almacenamiento de un computador (servidor) con ordenadores personales o servidores clientes a través de una red, haciendo uso de un sistema operativo optimizado para dar acceso con los protocolos CIFS, NFS, FTP o TFTP.

Generalmente, los sistemas NAS son dispositivos de almacenamiento específicos a los que se accede desde los equipos a través de protocolos de red (normalmente TCP/IP). También se podría considerar un sistema NAS a un servidor (Linux, Windows o cualquier otro) que comparte sus unidades por red, pero la definición suele aplicarse a sistemas específicos.

Los protocolos de comunicaciones NAS están basados en ficheros por lo que el cliente solicita el fichero completo al servidor y lo maneja localmente, están por ello orientados a información almacenada en ficheros de pequeño tamaño y gran cantidad. Los protocolos usados son de compartición de ficheros como NFS, *Microsoft Common Internet File System (CIFS).*

Muchos sistemas NAS cuentan con uno o más dispositivos de almacenamiento para incrementar su capacidad total. Normalmente, estos dispositivos están dispuestos en RAID o contenedores de almacenamiento redundante.

En la tecnología NAS, las aplicaciones y programas de usuario hacen las peticiones de datos a los

sistemas de ficheros de manera remota mediante protocolos CIFS y NFS, y el almacenamiento es local al sistema de ficheros.

Algunas ventajas del uso de NAS

- Capacidad de compartir las unidades.
- Menor coste.
- Utilización de la misma infraestructura de red con una gestión más sencilla.

1.1.3 Diferencias entre SAN/NAS

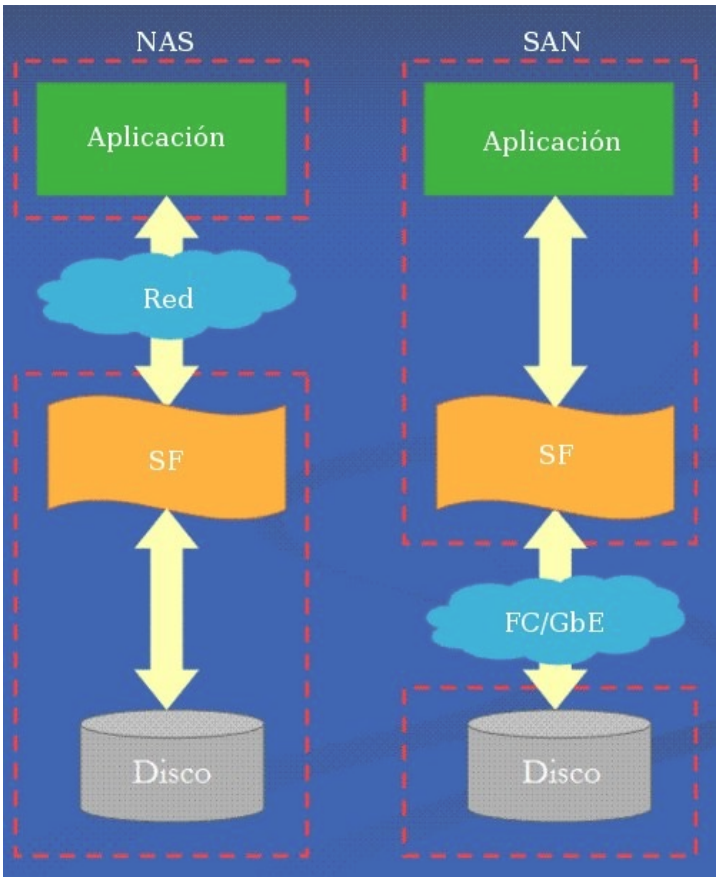


Figura 4. Diferencias entre NAS y SAN.

Una Red de Área de Almacenamiento (SAN) o un Almacenamiento Conectado a Red (NAS) permitirá gestionar de manera efectiva las exigencias de almacenamiento de manera independiente a los servidores existentes.

En todos los casos, las redes pueden ser creadas para unos cuantos *Gygabytes*, hasta 50TBs, y la flexibilidad de la solución proporciona la capacidad de incrementar los datos almacenados y utilizar soluciones de diferentes fabricantes. La elección entre SAN o NAS dependerá de las exigencias de velocidad, tamaño del almacenamiento y consideraciones relativas a la conectividad

existente.

La mayor diferencia entre el SAN y el NAS es que el primero está conectado a los servidores mediante redes de altísima velocidad (normalmente canales de fibra) y el segundo está conectado a la red local, donde su desempeño depende de la velocidad de la misma.

En una SAN la información se almacena en la red SAN, y en el modelo NAS los clientes tienen que solicitar los archivos a los servidores para que estos se los suministren.

	NAS	SAN
Tipo de datos	Archivos compartidos	Datos a nivel de bloque, por ejemplo, base de datos.
Cableado utilizado	Ethernet LAN	Fibre Chanel dedicado
Clientes principales	Usuarios finales	Servidores de aplicaciones
Acceso a discos	A través de dispositivos NAS	Acceso directo

Tabla 1. Trabajo con NAS y SAN

1.2 Soporte para almacenamiento en red

El almacenamiento (en red) se puede consolidar implementando servidores de alto rendimiento con conexiones de red de alta velocidad y configurados con grandes cantidades de almacenamiento rápido. Con la configuración apropiada, es posible suministrar acceso al almacenamiento a velocidades comparables al almacenamiento conectado directamente. Más aún, la naturaleza compartida de tal configuración a menudo hace posible reducir los costos, ya que los gastos asociados con suministrar almacenamiento centralizado y compartido pueden ser menores que suministrar el almacenamiento equivalente para cada uno de los clientes. Además, el espacio libre está consolidado, en vez de esparcido (pero no utilizable globalmente) entre los clientes. [1]

Los servidores de almacenamiento centralizado también puede hacer muchas tareas administrativas más fáciles. Por ejemplo, monitorizar el espacio libre es mucho más fácil cuando el almacenamiento a supervisar existe en un sólo servidor centralizado. Los respaldos también se pueden simplificar en gran medida usando un servidor de almacenamiento centralizado. Es posible hacer respaldos basados en la red para múltiples clientes, pero se requiere más trabajo para configurar y mantener. [1]

Existen varias tecnologías disponibles de almacenamiento en red; seleccionar una puede ser complicado. Casi todos los sistemas operativos en el mercado hoy día incluyen alguna forma de acceder a almacenamiento en red.

1.2.1 Software FreeNAS

FreeNAS es un sistema operativo basado en **FreeBSD**[4] que proporciona servicios de almacenamiento en red.

Este sistema operativo gratuito, de código abierto y software libre (basado en **Licencia BSD**[5]) permite convertir un ordenador personal en un soporte de almacenamiento accesible desde red, por ejemplo para almacenamientos masivos de información, música, *backups***,** entre otros.

1.2.5 Software NASLite

NASLite es una distribución comercial de Linux diseñada para convertir en un dispositivo de almacenamiento conectado a red simple equipos basados en x86 con interfaz PCI. Arranca desde el disco y se ejecuta en un disco RAM 4 MB, permitiendo la plena capacidad de las unidades del disco duro para utilizarlo como almacén. Además, es capaz de soportar archivos de servidor a los cliente.

Cuenta con tres variantes que apoyan los diferentes protocolos de servicio de archivos: Samba, NFS y/o FTP. Puede ser administrado remotamente a través de *telnet*, con la inclusión de un servidor web para mostrar el uso y los registros de error.

Existen versiones de NASLite para diferentes protocolos de red, o arrancar el sistema operativo del el CD-ROM, dispositivos USB de almacenamiento masivo o la unidad del disco duro.

1.2.6 Software NanoNAS

NanoNAS, como su nombre lo indica es una Red de Almacenamiento en Red (NAS por sus siglas en inglés) de sistema operativo servidor para transformar un equipo de base en un dedicado HTTP, SMB/CIFS, AFP o en el servidor de archivos. Es estable y compactado como para caber perfectamente en un disquete. Es entendido en entornos de baja seguridad o aplicaciones que requieren la disponibilidad simultánea de grandes cantidades de bajo costo de diseño de almacenamiento de red. Proporciona almacenamiento de archivos en red de forma rápida, segura y estable.

Es una versión de NASLite superior que en comparación con esta presenta algunas ventajas en cuanto a su forma optimizada, velocidad, rendimiento, además de la estabilidad operativa.

1.3 Almacenamiento unificado

Es de suma importancia poder contar con una infraestructura de almacenamiento en red segura y compartida, capaz de erradicar los "silos" de almacenamiento que se generan cuando se cuenta

[4] ***FreeBSD:*** *sistema operativo avanzado para arquitecturas x86 compatibles.*
[5] ***Licencia BSD:*** *licencia de software otorgada principalmente para los sistemas BSD (Berkeley Software Distribution).*

con varios niveles de aplicaciones. La eficacia inherente a las arquitecturas de almacenamiento unificado también contribuye a ahorrar costos en concepto de consumo eléctrico, refrigeración y espacio físico para los equipos. Los sistemas unificados resultan decisivos para incrementar la eficacia de TI y garantizar el éxito de la transformación del centro de datos a través de la virtualización y la consolidación.

1.3.1 Software Openfiler

Es un sistema operativo para almacenamiento en red, que utiliza una interfaz web para su gestión. Con las funciones incorporadas en **Openfiler**, es posible tener las ventajas de un servidor NAS basado en sistemas de archivos y de un servidor SAN basado en bloques, todo dentro del mismo sistema operativo.

A partir de lo antes expuesto se puede concluir que entre sus principales características posee un almacenamiento unificado, utilizando los servicios NAS/SAN, además de la presencia de *cluster* de alta disponibilidad.

1.3.2 Aspectos clave del almacenamiento de datos en red con Openfiler:

- **Fiabilidad**: Soporta *software* y *hardware* RAID con utilidades para monitorización y alertas, *snapshots*[6] de volúmenes y recuperación.
- **Disponibilidad**: Soporta *clusters* activo/pasivo de alta disponibilidad, MPIO y replicación a nivel de bloques.
- **Rendimiento**: el kernel de Linux 2.6 soporta las últimas CPU's, *hardware* de red y almacenamiento.
- **Escalabilidad**: escalabilidad del sistema de archivos mayor de 60TB y soporte para hacer crecer el sistema de archivos y los volúmenes en caliente.

Openfiler convierte la red en un centro de almacenamiento, tiene *backups* de los archivos, organiza la información, añade más espacio a cualquier máquina en la red o simplemente puede ser usado para cualquier función que requiera más espacio.

A través de su interfaz web, el administrador puede realizar diversas tareas administrativas tales como la creación de volúmenes, las cuotas de la red, la asignación de cuota de disco para usuarios y grupos y la gestión de matrices RAID.

1.4 Sistema de archivos

Un **sistema de archivos** son los métodos y estructuras de datos que un sistema operativo utiliza para seguir la pista de los archivos de un disco o partición; es decir, es la manera en la que se

[6] ***Snapshots:*** *también conocido copia instantánea de volumen. Es una función de algunos sistemas que realizan copias de seguridad de ficheros almacenándolos tal y como fueron capturados en el pasado.*

organizan los archivos en el disco. El término también es utilizado para referirse a una partición o disco que se está utilizando para almacenamiento, o el tipo del sistema de archivos que utiliza. De esta manera se puede decir que se tienen dos sistemas de archivos refiriéndose a que se tienen dos particiones en las que almacenar archivos, o que se utiliza el sistema de archivos extendido, refiriéndose al tipo del sistema de archivos. [2]

1.4.1 Zettabyte File System (ZFS)

ZFS es relativamente un nuevo sistema de archivos que ofrece características como integridad de datos, soporte para almacenamiento de cantidades abrumadoras de información, integración de los conceptos de “Sistema de Archivo” y “Administrador de Volúmenes”, Snapshots, clonación *copy-on-write*, chequeo de integridad continua, reparación automática, entre otras. Está implementado como software open source, bajo licencia CDDL (Common Development and Distribution License). En un principio conocido como Zettabyte File System, bautizado de esta forma por sus diseñadores debido a que su capacidad de almacenamiento es de 2^{58} Zettabytes, donde si se toma en consideración que 1 ZB= 10^9 TB (Terabytes) se puede discernir que se trata de una cantidad de información realmente abrumadora. [3]

A diferencia de los sistemas de archivo tradicionales, los cuales residen en un solo dispositivo y por consiguiente requieren de un manejador de volumen para usar más de un disco; ZFS está diseñado sobre agrupaciones de almacenamiento virtual llamados zpools. Un ZPool está conformado a partir de dispositivos virtuales (vdevs) los cuales a su vez están construidos a partir de dispositivos de bloques: archivos, particiones, o la totalidad de un disco. [3]

Debido a que ZFS es un sistema de archivos de 128 bits, es capaz de direccionar 1.84×10^{19} veces más datos que un sistema de archivos de 64 bits como por ejemplo NTFS. Los límites del sistema de archivos ZFS están diseñados para virtualmente nunca ser alcanzados. [3]

Los bloques que contienen los viejos datos pueden ser retenidos, permitiendo así realizar un *snapshot* del sistema de archivos a ser conservado. Estos son creados rápidamente debido a que os datos que lo conforman ya están almacenados en el sistema de archivos y además hacen un uso eficiente de espacio en disco, pues los datos no modificados son compartidos entre el sistema de archivos y sus *snapshots*. [3]

1. 5 Lenguaje de programación

Un **lenguaje de programación** es un idioma artificial para expresar procesos que pueden ser llevados a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Está formado por un conjunto de símbolos y

reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al proceso por el cual se escribe, se prueba, se depura, se compila y se mantiene el código fuente de un programa informático se le llama programación.

La implementación de un lenguaje es la que provee una manera de que se ejecute un programa para una determinada combinación de software y hardware.

1.5.1 PERL

PERL (por sus siglas en inglés de *Practical Extraction and Report Language*) es un sofisticado lenguaje de programación diseñado a finales de los años 80 por el lingüista norteamericano Larry Wall. PERL combina en forma concisa las mejores características de lenguajes como C, sed, awk y sh. En general, es posible reducir extensos programas escritos en C a pocas líneas de código de un programa PERL, con la ventaja adicional de que corren sin cambio sobre casi cualquier plataforma existente, lo que convierte a PERL en el lenguaje ideal para desarrollo de prototipos y aplicaciones robustas 100% portables. [4]

Junto con de las facilidades para desarrollo de aplicaciones web, PERL es útil en la resolución de cualquier tarea y posee habilidades para integrarse con sistemas operativos, bases de datos, redes, protocolos, ambientes gráficos, otros lenguajes de programación (Java, C, etc.), etc. Su versatilidad y eficiencia en el manejo de texto y, específicamente, de "expresiones regulares" no tiene equivalente en ningún otro lenguaje de programación actual. [4]

Este lenguaje dinámico de programación interpretado de alto nivel y de propósito general también está orientado a objetos aunque el programador no está forzado a programar con este esquema.

1. 6 Herramientas de desarrollo

Las **herramientas de desarrollo** son aquellos programas o aplicaciones que tienen un nivel de importancia en el desarrollo de un programa (en la programación). Pueden ser de importancia vital como un ensamblador, un compilador o un editor, o de importancia secundaria como un IDE (por sus siglas en inglés de *Integrated Development Environment*).

1.6.1 Gedit

Gedit es el editor de texto predeterminado del entorno de escritorio GNOME, que diseñado con propósitos generales, enfatiza en la simplicidad y facilidad de uso. Incluye herramientas para la edición de código fuente y textos estructurados como lenguajes de marcado.

Además de las funcionalidades básicas que son habituales en un editor de texto, como copiar, cortar y pegar texto, imprimir, etc., gedit incorpora coloreado de sintaxis para diversos lenguajes de programación y marcado. También posee pestañas en su interfaz para editar múltiples archivos

a la vez. Puede editar archivos de manera remota usando la biblioteca GVFS[7]. Otras características orientadas al código incluyen numeración de líneas, resaltado de la línea actual, indentación automática y copiado de seguridad del archivo. [5]

1. 7 Importación de datos de configuración

Zentyal es una plataforma de red unificada para las **PYMEs**[8], conocida anteriormente con el nombre de **eBox Platform**. Puede actuar gestionando la infraestructura de red, como puerta de enlace a Internet (Gateway), gestionando las amenazas de seguridad, como servidor de oficina, como servidor de comunicaciones unificadas o una combinación de estas. Incluye, además un marco de desarrollo para facilitar el desarrollo de nuevos servicios basados en Unix. Su propietario y patrocinador es la empresa española eBox Technologies S.L.

A pesar de que la interfaz de Zentyal facilita enormemente la labor del administrador de sistemas, existen tareas de configuración a través de la misma que pueden resultar tediosas si hay que repetirlas muchas veces.

Mediante la interfaz de programación de aplicaciones (API) que brinda Zentyal se pueden automatizar fácilmente sus tareas. Para lograrlo solamente se necesitan algunos conocimientos básicos del lenguaje de programación Perl, además de tener conocimiento de los métodos del módulo que se quiere utilizar. Una ventaja que se tiene con esto es el hecho de que la interfaz web emplea la misma interfaz de programación.

1.7.1 Entorno de desarrollo de nuevos módulos

Zentyal está diseñado precisamente pensando en la extensibilidad y es relativamente sencillo crear nuevos módulos. Cualquiera con conocimientos del lenguaje *Perl* puede aprovecharse de las facilidades que proporciona Zentyal para la creación de interfaces *Web*, y también beneficiarse de la integración con el resto de módulos y las demás características comunes de Zentyal. [6]

El diseño de Zentyal es completamente orientado a objetos y hace uso del patrón Modelo-Vista-Controlador (MVC), de forma que el desarrollador sólo necesita definir qué características desea en su modelo de datos, y el resto será generado automáticamente por Zentyal. Por si esto no fuese suficiente, se dispone de una herramienta llamada **zmoddev** que facilita más todavía el desarrollo de nuevos módulos, proporcionando plantillas auto-generadas en función de parámetros definidos por el usuario, lo que se traduce en un ahorro de tiempo. [6]

Acceso al repositorio de **zmoddev** https://svn.zentyal.org/zentyal/trunk/extra/zmoddev.

[7] ***GVFS:*** *es un reemplazo para GNOME VFS, el sistema virtual de archivos de GNOME.*

[8] ***PYMEs:*** *acrónimo de pequeña y mediana empresa.*

1.8 Augeas

Augeas es una herramienta de edición de configuración. Analiza los archivos de configuración en sus formatos nativos y los transforma en un árbol. Los cambios se realizan mediante la manipulación de este árbol y guarda de nuevo en los archivos de configuración nativos.

1.8.1 Lentes

Las **lentes de augeas** son los bloques de construcción básicos para establecer la cartografía de los archivos en el árbol de augeas. Se podría pensar a estas como un registro de tres funciones *get*, *put* y *create*, en el que la función *get* toma los contenidos de un archivo de texto, lo analiza y produce parte del árbol de augeas. Por otra parte, las funciones *put* y *create* toman un árbol y lo transforman nuevamente en un archivo de texto. La diferencia entre estas dos últimas es que la primera de ellas se utiliza cuando la parte del árbol que se transformó en el archivo corresponde a algo en el archivo de entrada, mientras la otra se utiliza cuando lo hace. [7]

Existen dos tipos de lentes: las primitivas y las combinadoras. La primera toma alguna alguna pieza del archivo y lo procesa de alguna manera y esta última se combina para formar pequeñas lentes de un objetivo de gran tamaño.

1.9 Metodología de desarrollo

Las **metodologías de desarrollo** de software definen una serie de procedimientos, técnicas y herramientas para la realización de un producto de software. Guían todo el proceso de desarrollo del software, motivo por el que son muchos los especialistas que se han dedicado a estudiarlas y definirlas. Dada la variedad de características y necesidades que posee un proyecto, estas metodologías se ha dividido en dos grandes grupos: Metodologías Ágiles/Ligeras y Metodologías Pesadas/Tradicionales.

1.9.1 OpenUp

OpenUP es un proceso simple si se compara con RUP, que permite más libertad pues se puede extender el modelo con parte de los demás modelos, para hacer frente a una amplia variedad de tipos de proyectos.

Se trata de un proceso más condensado del RUP pero es bastante completo, con la facilidad de aplicación a proyectos pequeños y medianos y para equipos más pequeños de desarrollo, más fácil de aprender.

Se aplica iterativo e incremental enfoques dentro de un ciclo de vida estructurado. OpenUP abraza una filosofía pragmática y ágil que se centra en la naturaleza colaborativa de desarrollo de software.

Esta metodología de desarrollo se puede emplear en al realización de proyectos para los que se precisa un corto período de tiempo, ideal por su contenido fundamental y necesario incluido.

1.10 Herramienta de modelado

Las **herramientas de modelado** de sistemas informáticos son herramientas que se emplean para la creación de modelos de sistemas que ya existen o que se desarrollan. Estas permiten crear un "simulacro" del sistema a bajo costo, pues son un conjunto de gráficos y textos que representan el sistema. Minimizan, además, los riesgos porque los cambios que se consideren necesarios hacer son más fáciles y rápidamente sobre el modelo del sistema ya implementado.

1.10.1 Visual Paradigm

Visual Paradigm es una herramienta **CASE** (*Cumputer Aided Software Engineering*) de Lenguaje de Modelaje Unificado (**UML**, por sus siglas en inglés de *Unified Modeling Language*) profesional que soporta el ciclo de vida completo del desarrollo de software, es decir, el análisis y diseño orientados a objetos, construcción, pruebas y despliegue.

Esta herramienta posibilita el aumento de la calidad del software, a través de la mejora de la productividad en el desarrollo y mantenimiento del software. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación, además de la reutilización de software, portabilidad y estandarización de la documentación. Por estas razones ha sido seleccionada para el modelado de la aplicación en cuestión.

1.11 Conclusiones del capítulo

Durante este capítulo se ha explicado y detallado la base sobre la que se trabajará en virtud de desarrollar la aplicación. A partir del estudio se definió lo siguiente: trabajar sobre la plataforma de desarrollo Zentyal, usar la herramienta ZFS, para la implementación usar el editor de texto gedit y lenguaje de programación Perl, como herramienta de modelado Visual Paradigm y como metodología de desarrollo OpenUP.

Capítulo 2: Análisis y diseño de la aplicación

2. Introducción

A partir de los elementos a los que se hizo referencia en el capítulo anterior, en el presente se recogen los aspectos que, relacionados con el análisis y el diseño basado en la metodología mencionada, harán posible contar con una aplicación robusta. Para ello se definirá la arquitectura de la herramienta, así como su proceso de funcionamiento en virtud de cumplir con los requisitos funcionales de la misma. Se hará la modelación de los diagramas fundamentales.

2.1 Actores del sistema

El **actor del sistema** es toda entidad externa a él con la que mantiene una relación y que le demanda una funcionalidad. Esto incluye a los operadores humanos como definiciones de rol por lo que un mismo individuo puede corresponder a uno o más actores, pero también a todos los sistemas externos, además de entidades abstractas, como el tiempo.

Nombre del actor	Descripción
Administrador	El administrador del sistema es la persona encargada de hacer toda la gestión de la información que brindará el sistema.

Tabla 2. Actores del sistema

2.2 Requisitos del sistema

Se construirá un *plugin* para la plataforma de desarrollo Zentyal como variante de Nova para Servidores, para administrar y servirse a través del almacenamiento de información en red. El mismo deberá cumplir con los requisitos que a continuación se relacionan:

Requisitos funcionales

Un **requisito funcional** define el comportamiento interno del software como detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica, es decir, capacidades o condiciones que el sistema debe cumplir.

CU1: Administrar plugin

RF1: Administrar plugin

CU2: Instalar antivirus

RF2: Instalar antivirus

CU3: Instalar cortafuegos

RF3: Instalar cortafuegos

CU4: **Gestionar piscinas**

RF4: Adicionar piscina

RF5: Eliminar piscina

CU5: **Mostrar listado de discos**

RF6: Mostrar listado de discos

CU6: **Adicionar discos**

RF7: Adicionar discos

CU7: Gestionar carpetas

RF8: Adicionar carpeta

RF9: Modificar carpeta

RF10: Eliminar carpeta

Requisitos no funcionales

Un **requisito no funcional** especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos. Se refieren a todos los requisitos que ni describen información a guardar, ni funciones a realizar. En otras palabras son las propiedades o cualidades que el producto debe tener.

Apariencia o interfaz externa

RNF1: La aplicación deberá presentar y mantener una interfaz amigable, interactiva, intuitiva y entendible por el usuario.

RNF2: Los mensajes mostrados al usuario deben seguir los patrones definidos por la plataforma Zentyal.

Usabilidad

RNF3: Se debe utilizar el idioma que predomina en el entorno donde será utilizado.

RNF4: Para el trabajo con el sistema se requerirán conocimientos mínimos.

RNF5: El administrador permanecerá en el sistema el tiempo que así lo considere.

Accesibilidad

RNF6: La información y las funcionalidades estarán disponibles y el administrador podrá acceder a ella en todo momento.

Disponibilidad

RNF7: El sistema siempre estará disponible para el administrador, dependiendo solamente de la operatividad de la plataforma Zentyal.

Rendimiento

RNF8: La aplicación permitirá que múltiples usuarios estén conectados a la vez.

RNF9: El tiempo de respuesta y procesamiento de la información serán rápidos.

Legales

RNF10: Las herramientas seleccionadas para el desarrollo del producto están respaldadas por licencias libres, bajo las condiciones de software libre. En el caso de la herramienta Visual Paradigm como no es libre se utiliza la licencia que posee la universidad.

RNF11: La aplicación y toda la documentación generada pertenecen al grupo de proyecto Gestión Documental y Archivística y a la Universidad de las Ciencias Informáticas.

Software

RNF12: Las PC's clientes deben tener instalado el navegador web *Google Chromium*.

RNF13: La PC servidor debe contar con servidor web Apache 2.0.

RNF14: La PC servidor debe tener instalada la plataforma de desarrollo Zentyal.

Hardware

Procesamiento:

RNF15: El servidor de aplicaciones requiere de una CPU Intel Pentium o compatible para su correcto funcionamiento.

Memoria RAM:

RNF16: El servidor de aplicaciones requiere una RAM de 2 Gb o superior para su correcto funcionamiento.

Capacidad en disco:

RNF17: El servidor de aplicaciones requiere de 1TB disponible para su correcto

funcionamiento.

Soporte:

RNF18: Realizar pruebas y mantenimiento necesarios para lograr el mejoramiento y evolución en el tiempo.

Confidencialidad

RNF 19: La información manejada por el sistema está protegida de acceso no autorizado.

Restricciones de diseño e implementación:

1. El sistema se desarrollará utilizando como lenguaje de programación Perl y Augeas.
2. El servidor debe tener instalado servidor web Apache 2.0.
3. El entorno de desarrollo integrado será Geany.
4. La interfaces destinadas al administrador deben programarse en lenguaje Perl.
5. Para la modelación del sistema se utilizará la herramienta Visual Paradigm 6.4.
6. La metodología de desarrollo de software empleada será OpenUp, haciendo uso del Lenguaje de Modelación Unificado (UML).

2.3 Casos de uso del sistema

Un **caso de uso** es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso. Los personajes o entidades que participarán en un caso de uso se denominan actores. Sus diagramas sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas. Los casos de uso no son parte del diseño (cómo), sino parte del análisis (qué). De tal forma que al ser parte del análisis ayudan a describir qué es lo que es sistema debe hacer. Son qué hace el sistema desde el punto de vista del usuario. Es decir, describen un uso del sistema y cómo este interactúa con el usuario, en este caso el administrador. [8]

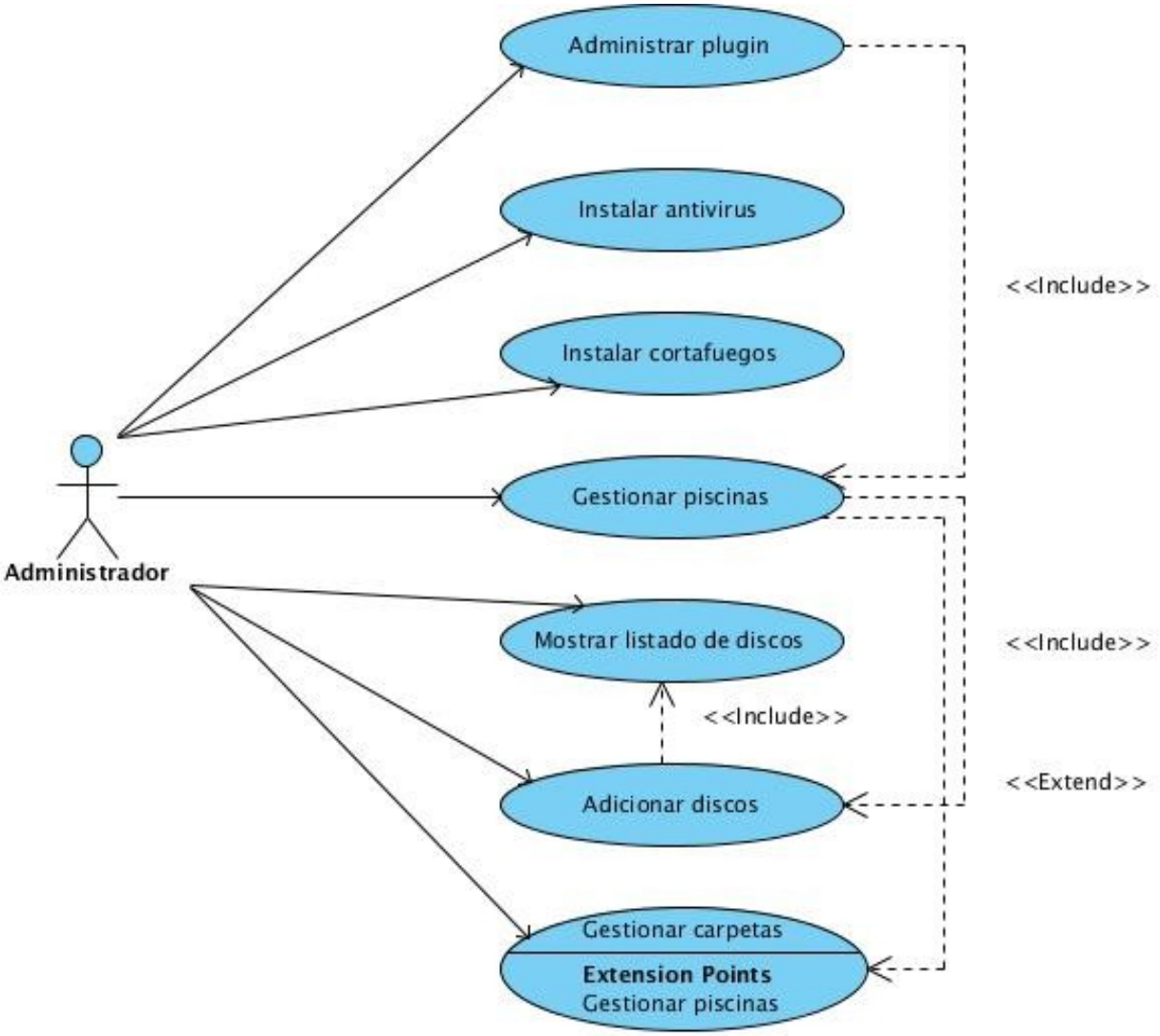


Figura 5. Diagrama de Casos de Uso del Sistema

2.3.1 Descripción de los casos de uso

A continuación se describen textualmente los casos de uso del sistema que fueron modelados en el diagrama anterior, especificando su propósito y sus condiciones de existencia.

Descripción del caso de uso Administrar plugin

Caso de uso	Administrar plugin.
Objetivo	Permite al administrador administrar el módulo NAS.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el administrador selecciona la opción Componentes de Zentyal en el que sistema muestra tres pestañas con las opciones Instalar, Actualizar y Borrar dando la posibilidad de escoger la acción que se desee.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe estar autenticado en la plataforma Zentyal y el

	módulo NAS debe existir en el repositorio.
Postcondiciones	El módulo NAS será instalado, actualizado o eliminado.
Flujo Normal de Eventos	
Acción del actor	**Acción del sistema**
1. Selecciona la opción Gestión de software/Componentes de Zentyal.	2. Muestra la interfaz Componentes de Zentyal que contiene las opciones de Instalar, Actualizar y Borrar.
3. Selecciona una opción: a. Si selecciona la opción Instalar véase la sección Instalar módulo. b. Si selecciona la opción Actualizar véase la sección Actualizar módulo. c. Si selecciona la opción Borrar véase la sección Borrar módulo.	
Sección "Instalar módulo"	
Acción del actor	**Acción del sistema**
	1. Muestra un formulario para añadir una nueva carpeta.
2. Marca el cuadro de selección correspondiente al módulo NAS. 3. Selecciona el botón Instalar.	4. Instala el módulo. 5. Muestra un mensaje de verificación.
Sección "Actualizar módulo"	
Acción del actor	**Acción del sistema**
	1. Muestra un listado con los módulos que tengan actualizaciones.
2. Marca el cuadro de selección correspondiente al módulo NAS. 3. Selecciona el botón Actualizar.	4. Actualiza el módulo. 5. Muestra un mensaje de verificación.
Sección "Modificar carpeta"	
Acción del actor	**Acción del sistema**
	1. Muestra un listado con los módulos instalados.
2. Marca el cuadro de selección correspondiente al módulo NAS. 3. Selecciona el botón Borrar.	4. Borra el módulo. 5. Muestra un mensaje de verificación.

Descripción del caso de uso Instalar antivirus

Caso de uso	Instalar antivirus.
Objetivo	Permite al administrador instalar el antivirus.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el administrador selecciona la opción Componentes de Zentyal en el que el sistema muestra la opción de Instalar, permitiendo instalar el módulo Antivirus.
Complejidad	Alta.

Prioridad	Crítico.
Precondiciones	El administrador debe estar autenticado en la plataforma Zentyal y el módulo de antivirus debe existir en el repositorio.
Postcondiciones	El antivirus será instalado.
Flujo Normal de Eventos	
Acción del actor	**Acción del sistema**
1. Selecciona la opción Gestión de software/Componentes de Zentyal.	2. Muestra la interfaz Componentes de Zentyal que contiene la opción de Instalar.
3. Selecciona la opción Instalar.	4. Muestra un listado con los módulos disponibles incluyendo el módulo Antivirus.
5. Marca el cuadro de selección correspondiente al módulo Antivirus. 6. Selecciona el botón Instalar.	7. Instala el módulo. 8. Muestra un mensaje de verificación.

Descripción del caso de uso Instalar cortafuegos

Caso de uso	Instalar cortafuegos.
Objetivo	Permite al administrador instalar cortafuegos.
Actores	Administrador (inicia).
Resumen	El caso de uso inicia cuando el administrador selecciona la opción Componentes de Zentyal en el que el sistema muestra la opción de Instalar, permitiendo instalar el módulo Cortafuegos.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe estar autenticado en la plataforma Zentyal y el módulo de cortafuegos debe existir en el repositorio.
Postcondiciones	El cortafuegos será instalado.
Flujo Normal de Eventos	
Acción del actor	**Acción del sistema**
1. Selecciona la opción Gestión de software/Componentes de Zentyal.	2. Muestra la interfaz Componentes de Zentyal que contiene la opción de Instalar.
3. Selecciona la opción Instalar.	4. Muestra un listado con los módulos disponibles incluyendo el módulo Cortafuegos.
5. Marca el cuadro de selección correspondiente al módulo Cortafuegos. 6. Selecciona el botón Instalar.	7. Instala el módulo. 8. Muestra un mensaje de verificación.

Descripción del caso de uso Gestionar piscina

Caso de uso	Gestionar piscina.
Objetivo	Permite al administrador gestionar (adicionar y eliminar) las piscinas del sistema.
Actores	Administrador (inicia).
Resumen	El caso de uso se inicia cuando el administrador selecciona el módulo NAS, se posiciona en la vista Piscinas, luego introduce los datos necesarios, se

	auxilia de las vistas Discos o Carpetas (de forma opcional) para añadir la piscina.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe estar autenticado en la plataforma Zentyal.
Postcondiciones	Nueva piscina registrada en el sistema, o eliminada.
Flujo Normal de Eventos	
Acción del actor	**Acción del sistema**
1. Selecciona el módulo NAS.	2. Muestra una breve descripción sobre las operaciones con las piscinas.
3. Selecciona una opción: a. Si selecciona la opción de añadir nueva piscina, véase sección Adicionar piscina. b. Si selecciona la opción eliminar, véase sección Eliminar piscina.	
Sección “Adicionar piscina”	
Acción del actor	**Acción del sistema**
1. Selecciona la opción añadir nueva piscina.	2. Muestra un formulario para añadir una nueva piscina.
2. Introduce los datos de la nueva piscina. 3. Selecciona la acción Añadir. 4. Selecciona la opción Discos.	5. Verifica que se haya añadido al menos un disco a la piscina. 6. Si no se ha añadido disco a la piscina véase flujo alterno 5.1. 7. Si los datos son correctos el sistema inserta una nueva piscina, terminando así el CUS.
Flujo Alterno 5.1	
Acción del actor	**Acción del sistema**
	5.1 Si no se ha añadido al menos un disco el sistema muestra un mensaje de error notificando que se debe insertar al menos un disco. Luego retorna al paso 1.
Sección “Eliminar piscina”	
Acción del actor	**Acción del sistema**
	1. Muestra todas las piscinas que han sido añadidas al sistema.
2. Selecciona la piscina a eliminar.	3. Elimina la piscina seleccionada, terminando así el CUS.

Descripción del caso de uso Mostrar listado de discos

Caso de uso	Mostrar listado de discos.
Objetivo	Permite al administrador obtener el listado de los discos de la máquina.
Actores	Administrador (inicia).

Resumen	El caso de uso se inicia cuando el administrador selecciona la opción Discos en la que se mostrará el listado de los discos de la máquina.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe estar autenticado en la plataforma Zentyal.
Postcondiciones	Obtener el listado de discos.
Flujo Normal de Eventos	
Acción del actor	**Acción del sistema**
1. Selecciona la opción Discos de una piscina. 2. Selecciona la opción de añadir nuevo disco.	3. El sistema muestra un listado con los discos que posee para ser seleccionados, terminando así el CUS.

Descripción del caso de uso Adicionar discos

Caso de uso	Adicionar discos.
Objetivo	Permite al administrador adicionar discos a las piscinas del sistema.
Actores	Administrador (inicia).
Resumen	El caso de uso se inicia cuando el administrador selecciona la opción Discos, luego selecciona los discos necesarios para añadirlos a la piscina correspondiente.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe estar autenticado en la plataforma Zentyal.
Postcondiciones	Nuevo disco añadido a la piscina.
Flujo Normal de Eventos	
Acción del actor	**Acción del sistema**
1. Selecciona la opción Discos de una piscina.	2. El sistema muestra un listado con los discos que posee para ser seleccionados.
3. Selecciona el disco. 4. Selecciona una opción: a. Si selecciona la opción cancelar, véase sección Cancelar. b. Si selecciona la opción añadir, véase sección Adicionar disco.	
Sección “Cancelar”	
Acción del actor	**Acción del sistema**
	1. Muestra un listado con los discos que posee para ser seleccionados.
2. Selecciona la opción Cancelar.	3. Retorna al paso 1.
Sección “Adicionar disco”	
Acción del actor	**Acción del sistema**
	1. Muestra un listado con los discos que posee para ser seleccionados.

2. Selecciona los discos que serán añadidos a la piscina. 3. Selecciona la opción Añadir.	4. Muestra una tabla con los discos añadidos a la correspondiente piscina, terminando así el CUS.

Descripción del caso de uso Gestionar carpetas

Caso de uso	Gestionar carpetas.
Objetivo	Permite al administrador gestionar (adicionar, modificar y eliminar) las carpetas de las piscinas.
Actores	Administrador (inicia).
Resumen	El caso de uso se inicia cuando el administrador selecciona la opción Carpetas, en la vista correspondiente introduce los datos necesarios para añadir las carpetas.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El administrador debe estar autenticado en la plataforma Zentyal.
Postcondiciones	Nueva carpeta registrada en el sistema, modificada o eliminada.
Flujo Normal de Eventos	
Acción del actor	**Acción del sistema**
1. Selecciona la opción Carpetas de una piscina.	2. Muestra una breve descripción sobre las operaciones con las carpetas.
3. Selecciona una opción: a. Si selecciona la opción de añadir nueva carpeta, véase sección Adicionar carpeta. b. Si selecciona la opción editar, véase sección Modificar carpeta. c. Si selecciona eliminar, véase sección Eliminar carpeta.	
Sección “Adicionar carpeta”	
Acción del actor	**Acción del sistema**
1. Selecciona la opción de añadir nueva carpeta.	2. Muestra un formulario para añadir una nueva carpeta.
3. Introduce los datos de la nueva carpeta. 4. Selecciona la opción Añadir.	5. El sistema inserta una nueva carpeta, terminando así el CUS.
Sección “Modificar carpeta”	
Acción del actor	**Acción del sistema**
	1. Muestra una tabla con las carpetas añadidas.
2. Selecciona la opción editar.	3. Muestra el formulario correspondiente a la carpeta que se desea modificar.
4. Realiza los cambios pertinentes.	5. Muestra la lista de carpetas con las modificaciones hechas a la carpeta en cuestión, terminando así el CUS.
Sección “Eliminar carpeta”	
Acción del actor	**Acción del sistema**

	1. Muestra todas las carpetas que han sido añadidas a su correspondiente piscina.
2. Selecciona la carpeta a eliminar.	3. Elimina la carpeta seleccionada, terminando así el CUS.

2.4 Diagrama de Clases del diseño

Representa las clases que serán utilizadas dentro del sistema y las relaciones que existen entre ellas. Sirve para visualizar las relaciones entre las clases que involucran el sistema, las cuales pueden ser asociativas, de herencia, de uso y de convencimiento. Un diagrama de clases está compuesto por los siguientes elementos: Clase: atributos, métodos y visibilidad; y Relaciones: herencia, Composición, Agregación, Asociación y Uso. [9]

La siguiente figura muestra las clases que intervienen en la propuesta de solución. La clase NAS es la clase principal la cual se encarga de realizar todas las operaciones del sistema, esta contiene una instancia de la clase *Pool*, la cual contiene una vista general del sistema, esta tiene a su vez una instancia de las clases *Disk* y *Path.* La primera muestra un listado de los discos que hay en la máquina para ser utilizados por las piscinas, y la última muestra una vista de todas las carpetas correspondientes a las diferentes piscinas.

Diagrama de clases del sistema

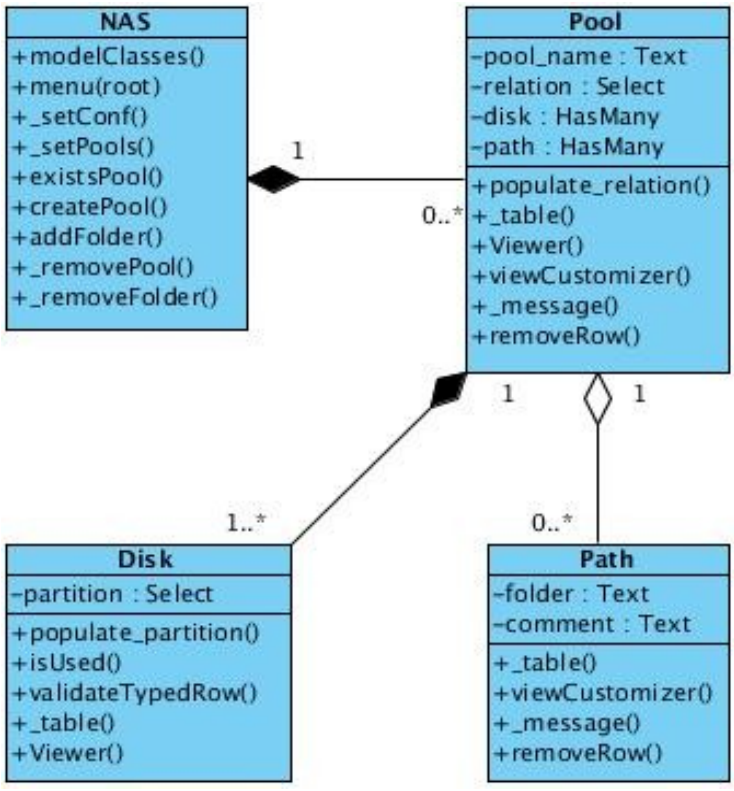


Figura 6. Diagrama de clases del sistema

Diagrama de clases con estereotipos web

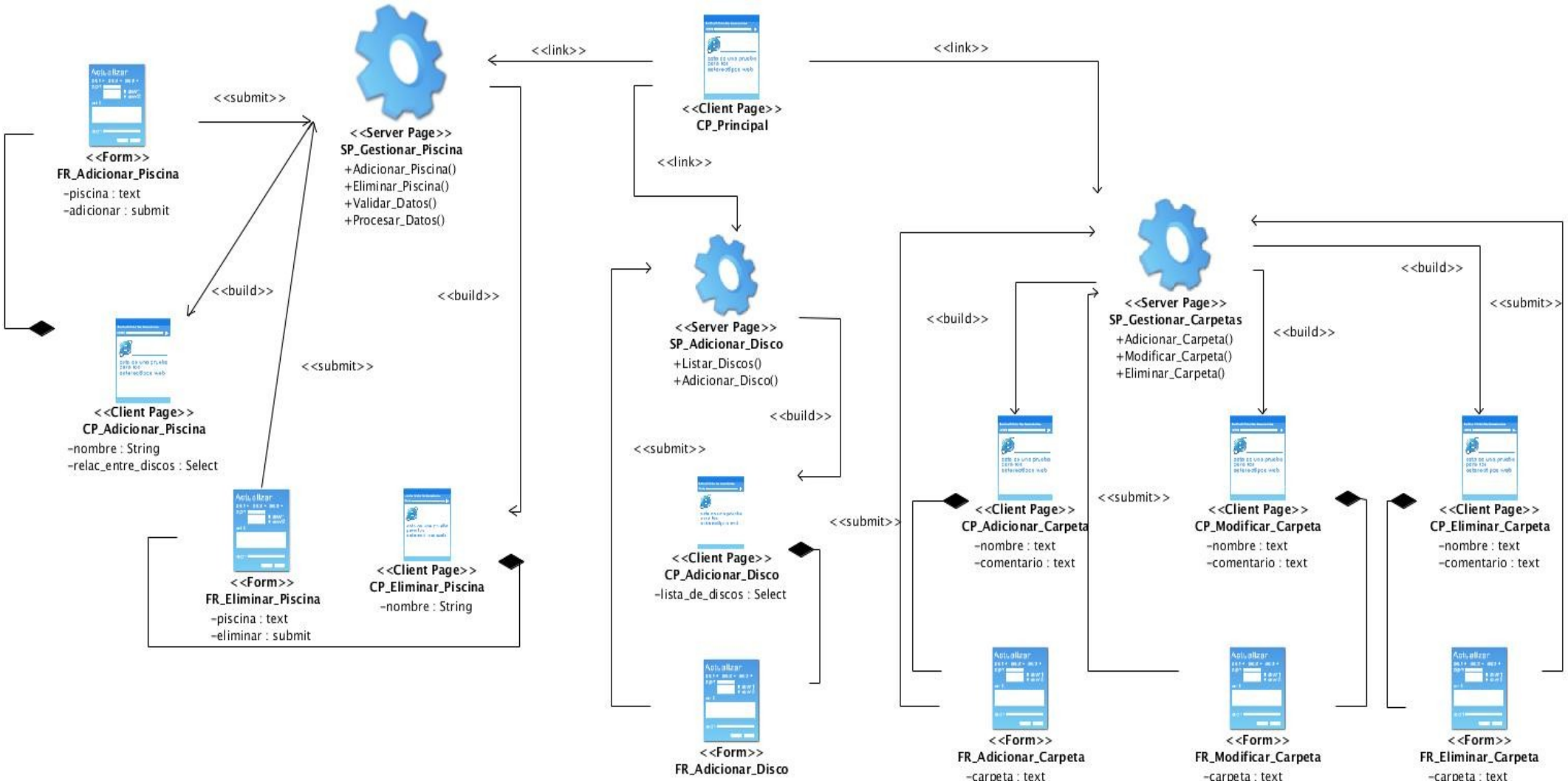


Figura 7. Diagrama de clases con estereotipos web.

2.5 Modelo de dominio

El **modelo de dominio** es una representación visual estática del entorno real objeto del proyecto. Es un diagrama con los objetos que existen (reales) relacionados con el proyecto que se va a llevar a cabo y las relaciones existentes entre ellos. Pero no son clases de software (aunque algunos objetos pueden terminar siéndolo). Se llama "de dominio" para distinguirlo del Modelo de Negocio (*Model of Business*) que en **RUP** (*Rational Unified Process*) es un concepto más amplio. El MOB de RUP incluye toda la organización, sus relaciones, sus procesos, todo. Sin embargo, el modelo de dominio se centra en una parte del negocio, la relacionada con el ámbito del proyecto, en otras palabras, en este contexto el término "dominio" representa una parte del "negocio". [10]

Para una mejor concepción de esta representación se muestra a continuación dicho diagrama. El administrador hace uso de la interfaz que proporciona la plataforma de desarrollo Zentyal. Esta cuenta con aplicaciones, en este caso contará con el módulo NAS, a través del cual se gestionarán piscinas, que tienen una relación de composición con discos y de agregación con carpetas. Los discos serán los que posee el ordenador (servidor) y las carpetas contendrán la información que será compartida en la red.

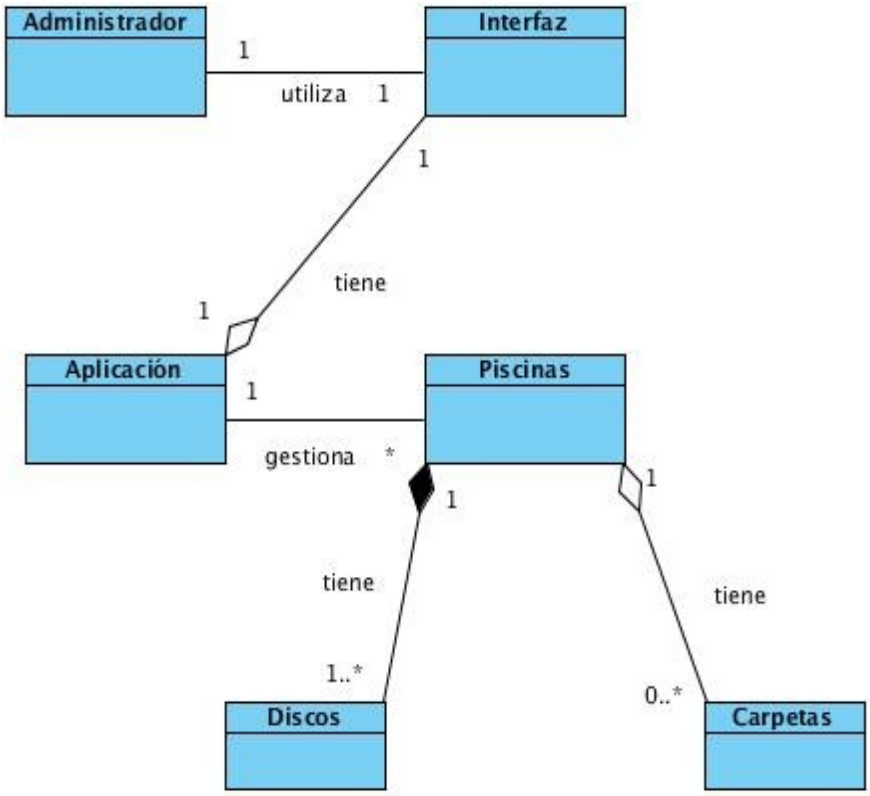


Figura 8. Diagrama de modelo de dominio

2.6 Arquitectura de la aplicación

Para la **arquitectura de la aplicación** se hará uso del patrón Modelo-Vista-Controlador. El mismo permite separar los datos de la aplicación, la interfaz y la lógica en tres componentes distintos. Este patrón, de llamada y retorno está presente en la interfaz web y el código que provee de datos dinámicos a la página.

Diagrama de componentes

La vista la proporciona la plataforma Zentyal que funcionará como un subsistema, que cuenta con el *plugin* desde el que se establece comunicación con el lente de augeas que funciona como modelo para permitir el intercambio de información entre la vista ya antes mencionada y el controlador en este caso el archivo de configuración necesario para realizar las operaciones requeridas. Este último es el responsable de recibir eventos de entrada de la vista. Todo esto se muestra mediante el siguiente diagrama.

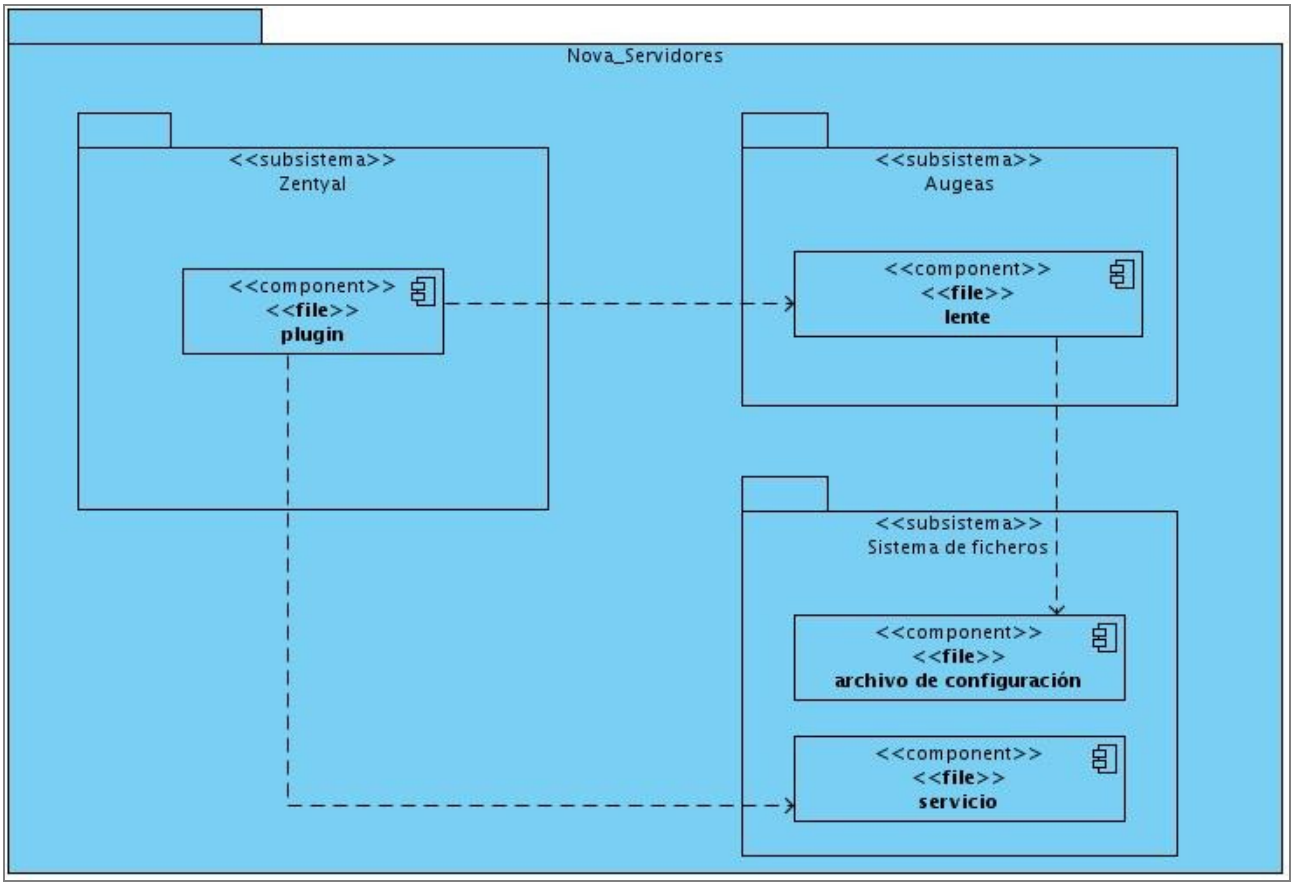


Figura 9. Diagrama de componentes

2.7 Diagramas de interacción

Un **diagrama de interacció**n muestra una cierta vista sobre los aspectos dinámicos de los sistemas modelados. Aunque es una representación gráfica se distingue fuertemente de los diagramas de secuencia y de comunicación, dos de los otros diagramas de interacción. Consiste en un conjunto de objetos y sus relaciones, incluyendo los
mensajes que se pueden enviar entre ellos. Son utilizados para modelar los aspectos dinámicos de un sistema.

2.7.1 Diagramas de secuencia

Un **diagrama de secuencia** m0uestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso. Mientras que el diagrama de casos de uso permite el modelado de una vista *business* del escenario, el diagrama de secuencia contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes intercambiados entre los objetos.

Típicamente se examina la descripción de un caso de uso para determinar qué objetos son necesarios para la implementación del escenario. Si se dispone de la descripción de cada caso de uso como una secuencia de varios pasos, entonces se puede "caminar sobre" ellos para descubrir qué objetos son necesarios para que se puedan seguir los pasos.

Un diagrama de secuencia muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales.

Existen dos tipos de mensajes: sincrónicos y asincrónicos. Los primeros se corresponden con llamadas a métodos del objeto que recibe el mensaje. El objeto que envía el mensaje queda bloqueado hasta que termina la llamada. Este tipo de mensajes se representan con flechas con la cabeza llena. Los mensajes asincrónicos terminan inmediatamente, y crean un nuevo hilo de ejecución dentro de la secuencia. Se representan con flechas con la cabeza abierta.

También se representa la respuesta a un mensaje con una flecha discontinua.

A continuación se muestran los diagramas de secuencia para los requisitos funcionales que componen el CU Gestionar piscina.

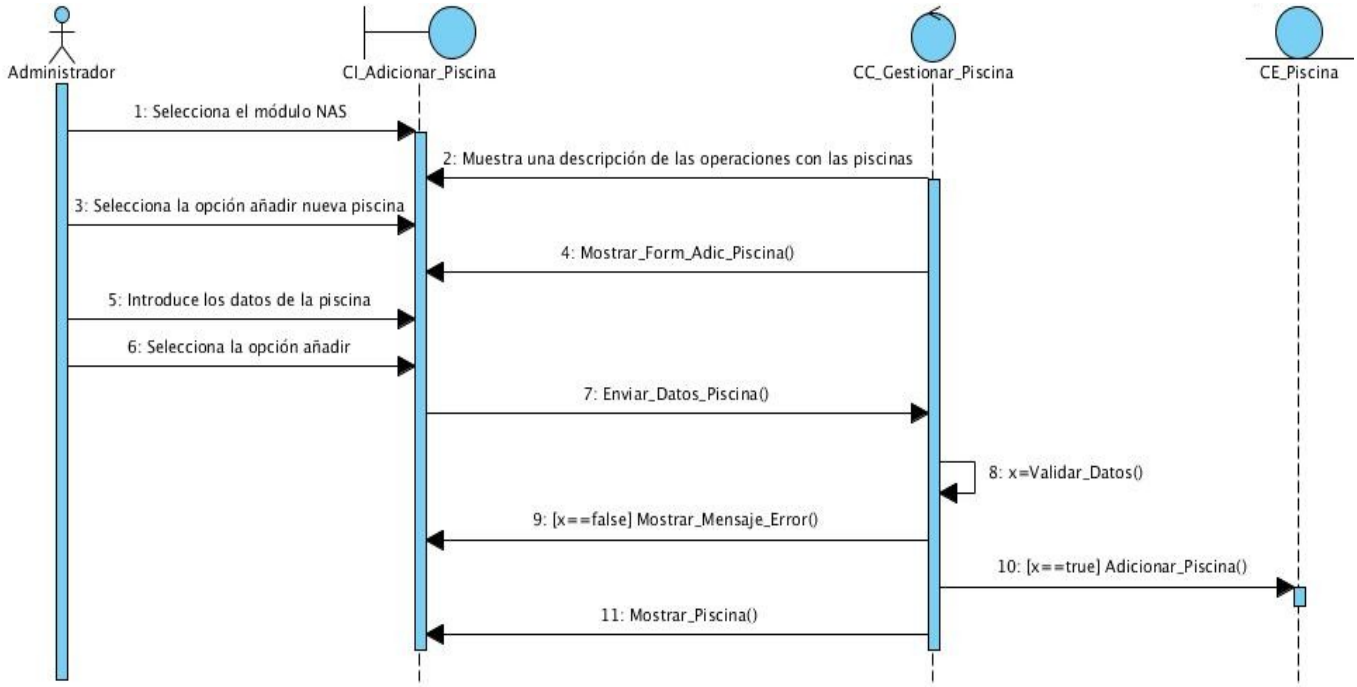


Figura 10. Diagrama de secuencia Adicionar Piscina

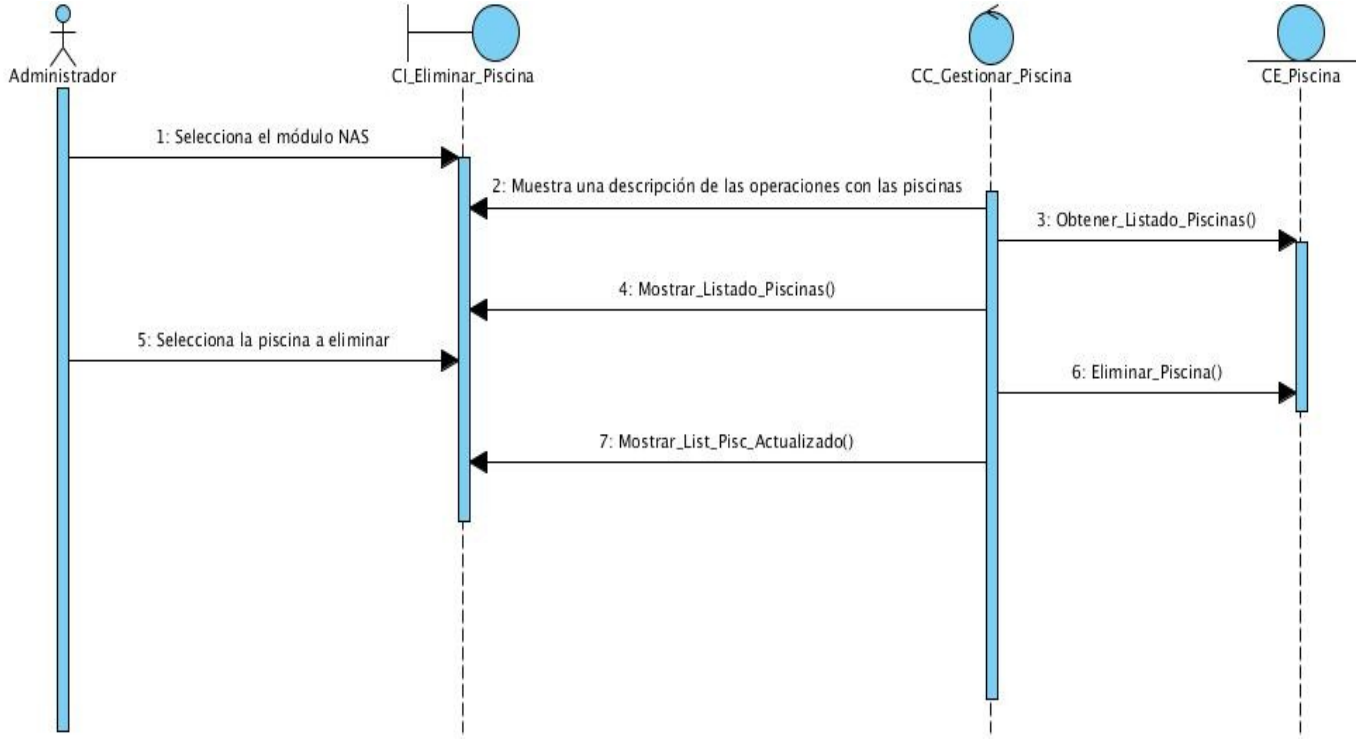


Figura 11. Diagrama de secuencia Eliminar Piscina

2.8 Patrones de diseño

Los **patrones de diseño** son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces, brindando una solución ya probada y documentada. Se deben tener presente los siguientes elementos de un patrón: su nombre, el problema (cuándo aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios).

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular. En otras palabras, un patrón de diseño no es más que una solución estándar para un problema común de programación. Una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios. [11]

2.8.1 Patrones GRASP

GRASP es un acrónimo de *General Responsibility Assignment Software Patterns o* Patrones Generales de Software para Asignar Responsabilidades). El nombre se eligió para sugerir la importancia de aprender estos principios para diseñar con éxito el software orientado a objetos. Los patrones GRASP son utilizados para describir los principios fundamentales del diseño y la asignación de responsabilidades. [11]

- **Experto en información**

El GRASP de experto en información es el principio básico de asignación de responsabilidades. Indica, por ejemplo, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada, posibilitando la disminución del acoplamiento.

- **Creador**

El patrón creador permite identificar quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases.

Una de las consecuencias de usar este patrón es la visibilidad entre la clase creada y la clase creador. Una ventaja es el bajo acoplamiento, lo cual supone facilidad de mantenimiento y reutilización.

La creación de instancias es una de las actividades más comunes en un sistema orientado a objetos. En consecuencia de esto resulta útil contar con un principio general para la asignación de

las responsabilidades de creación. Si se asignan bien el diseño puede soportar un bajo acoplamiento, mayor claridad, encapsulación y reutilización.

- **Alta cohesión**

La cohesión es la medida de la fuerza que une a las responsabilidades de una clase. Una clase con baja cohesión hace muchas cosas no relacionadas, o hace demasiado trabajo. Tales clases no son convenientes pues son difíciles de mantener, de reutilizar y de entender. Una clase con alta cohesión mejora la claridad y la facilidad de su uso, su mantenimiento se simplifica y es fácil de reutilizar.

Es decir, dice que la información que almacena una clase debe de ser coherente y debe estar (en la medida de lo posible) relacionada con la clase.

- **Bajo acoplamiento**

Bajo acoplamiento es la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

Los conceptos de cohesión y acoplamiento no están íntimamente relacionados, sin embargo se recomienda tener un mayor grado de cohesión con un menor grado de acoplamiento. De esta forma se tiene menor dependencia y se especifican los propósitos de cada objeto en el sistema.

2.8.2 Patrón DAO

Data Access Object (DAO, Objeto de Acceso a Datos) es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una base de datos o un archivo.

De esta forma se encapsula la forma de acceder a la fuente de datos. Este patrón surge históricamente de la necesidad de gestionar una diversidad de fuentes de datos, aunque su uso se extiende al problema de encapsular no sólo la fuente de datos, sino además ocultar la forma de acceder a los datos. Se trata de que el software cliente se centre en los datos que necesita y se olvide de cómo se realiza el acceso a los datos o de cuál es la fuente de almacenamiento. [12]

2.9 Conclusiones del capítulo

En este capítulo se definieron los aspectos relacionados con el análisis y el diseño de la aplicación. Se definieron, además, los requisitos funcionales de la aplicación, así como los no funcionales, en virtud de lograr el buen funcionamiento de la aplicación. Se modelaron algunos diagramas de consideración importante para favorecer una mejor comprensión de las funcionalidades con las que debe contar el sistema. También se aplicaron los patrones de diseño GRASP y DAO, para asignar responsabilidades a las clases y garantizar un acceso a datos orientado a objetos, respectivamente.

Capítulo 3: Implementación y realización de pruebas

3. Introducción

En el presente capítulo se plasman elementos como el Diagrama de Despliegue, mediante el que se representan los principales componentes y sus relaciones. También se realizó la implementación de la herramienta, para la cual se muestran el código fuente y los principales métodos. Asimismo, se realizan las pruebas que validan la solución propuesta.

3.1 Diagrama de Despliegue

El **diagrama de despliegue** es un tipo de diagrama del Lenguaje Unificado de Modelado que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes. Los elementos que usa este tipo de diagrama son nodos, componentes y asociaciones.

La mayoría de las veces el modelado de la vista de despliegue implica modelar la topología del hardware sobre el que se ejecuta el sistema. Aunque UML no es un lenguaje de especificación hardware de propósito general, se ha diseñado para modelar muchos de los aspectos hardware de un sistema a un nivel suficiente para que un ingeniero software pueda especificar la plataforma sobre la que se ejecuta el software del sistema.

Los elementos usados por este tipo de diagramas son:

Nodos: los elementos de procesamiento con al menos un procesador, memoria, y posiblemente otros dispositivos.

Dispositivos: los nodos son estereotipados sin capacidad de procesamiento en el nivel de abstracción que se modela.

Conectores: expresan el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo.

A continuación se muestra el diagrama de despliegue de la aplicación:

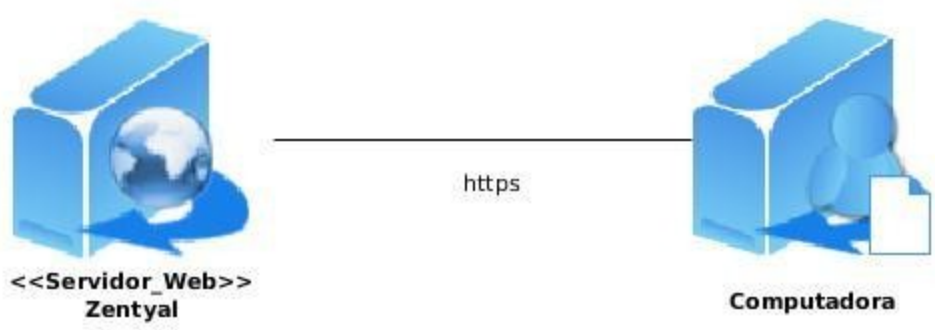


Figura 12. Diagrama de Despliegue

3.2 Resultados obtenidos

Para obtener la aplicación deseada se tomaron como requerimientos básicos, en primer lugar lograr la familiarización con el lenguaje de programación correspondiente (Perl) pues aunque los módulos están escritos precisamente en dicho lenguaje, resulta necesario el fácil entendimiento de las estructuras de datos y la sintaxis. También, y en segundo lugar, la elección para la configuración del entorno de construcción mediante el uso de una máquina virtual con Zentyal o usando la propia máquina.

3.2.1 Usando una máquina virtual con Zentyal

Este es el caso recomendado en virtud de no tener que pasar por situaciones no deseadas. Para el mismo se debe contar con los siguientes aspectos:

- Se puede utilizar VirtualBox[9] (aunque esa es una decisión muy personal) con una instalación limpia de Zentyal 2.2.
- Probablemente se quiera establecer una conexión *SSH* a la máquina virtual para trabajar más cómodamente.
- Hay que tener todos los paquetes necesarios de construcción que ya están en los repositorios o crear un repositorio en la propia máquina con los paquetes y dependencias necesarias.
- Siempre es una buena práctica utilizar las capacidades de instantáneas para mantener un ambiente limpio para volver cuando sea necesario.

[9] http://www.virtualbox.org

3.2.2 Usando la propia máquina

Esta opción es menos recomendable porque cualquier acción que se haga, sin tener conocimientos sólidos puede afectar el sistema. Pero si se siente seguro y hay pleno dominio sobre las herramientas y aplicaciones que serán utilizadas pues:

- Se tendrá que añadir los PPA de Zentyal con el fin de poder instalar las herramientas de construcción. Se recomienda consultar la Guía de Instalación[10] para obtener más información.
- Se necesita una distribución Linux Debian, Ubuntu, Nova para trabajar con el repositorio mencionado.
- Se puede construir los paquetes localmente y luego copiarlos a las máquinas virtuales utilizando *scp*.

Luego de hacer la elección se procede a la instalación de la herramienta de desarrollo del módulo de Zentyal (también se instalará el paquete **zbuildtools** necesario para construir módulos de Zentyal):

sudo apt-get install zmoddev

Para crear el módulo hay que utilizar zmoddev, una colección de *scripts* de conveniencia que ayudará a crear la estructura y los archivos básicos para un módulo de Zentyal.

3.2.3 Creando zentyal-nas

En primer lugar hay que crear todo el andamiaje de un nuevo módulo. Por lo general, esto se puede hacer copiando la carpeta completa de un módulo simple a otra parte, eliminar cosas, no es necesario el cambio de nombre y de espacios de nombres, pero se puede hacer utilizando zmoddev. Para ello hay que ejecutar el siguiente comando:

zentyal-moddev-create --module-name nas -main-class NAS --version 1.0

Lo anterior creará un directorio llamado nas con todos los archivos que componen un módulo de Zentyal.

El modelo de datos para gestionar los módulos es muy sencillo, se necesitará un campo para almacenar el nombre del módulo y otro campo para habilitar y deshabilitarlo.

Se utilizará *zentyal-moddev-model,* una herramienta que permite añadir fácilmente nuevos

[10] http://trac.zentyal.org/wiki/Documentation/Community/Installation/InstallationGuide

modelos a nuestro módulo de Zentyal. Luego de tener bien claros los modelos con que contará el módulo se procede a su creación mediante la ejecución del siguiente comando dentro del directorio nas:

zentyal-moddev-model --main-class NAS --name Pool --field pool_name:Text --field disk_relationship:Select –field disk:HasMany –field path:HasMany --model table

El comando anterior suma un nuevo modelo con el nombre deseado (en este caso el nombre es *Pool*) que se compone de un tipo *Text* llamado *pool_name,* un tipo *Select* llamado *disk_relationship* y del uso de del tipo *HasMany* para contener a los submodelos *Disk* y *Path.* El tipo de modelo es una tabla instanciada de un formulario como se hizo en la versión 1.0 y *main-class* es NAS como en la creación inicial del módulo.

Es importante que se sepa que existen diferentes tipos de modelos en Zentyal como Text, Int, Boolean, Select, Port, Password, PortRange, Service, HostIP, Host, IPAddr, entre otros, los cuales son utilizados en dependencia de las necesidades que se presenten para el desarrollador.

Construir e instalar el paquete

Por ahora no hay ninguna entrada en el menú para ver la vista asociada a este módulo, pero se puede fácilmente conocer la dirección URL al saber el nombre del modelo.

En la parte que van los modelos (*nas/src/EBox/NAS/Model*) se encuentran precisamente los modelos Disk, Path y Pool, este último es donde se maneja mayormente la administración del módulo.

En el método _**table** se definen los valores con los que va a contar la tabla con los datos de interés, con sus tipos de datos correspondientes y las acciones que se pueden hacer con los mismos. En la figura que se muestra a continuación se pueden observar los elementos por los que se compone la tabla correspondiente al modelo *Pool.*

```
sub _table
{
    my ($self) = @_;

    my @fields =
    (
        new EBox::Types::Text(
            'fieldName' => 'pool_name',
            'printableName' => __('Nombre'),
            'unique' => 1,
            'editable' => 1,
            'help' => 'Nombre de la piscina.',
        ),
        new EBox::Types::Select(
            'fieldName' => 'disks_relationship',
            'printableName' => __('Relación entre discos'),
                        'editable' => 1,
                        'populate' => \&populate_campo,
                'help' => __('La relaci&oacute; que existir&aacute; entre los discos (Ninguna, Mirror, Raidz).'),
        ),
        new EBox::Types::HasMany(
            'fieldName' => 'disk',
            'printableName' => __('Discos'),
                        'foreignModel' => 'nas/Disk',
                        'view' => '/NAS/View/Disk',
        ),
        new EBox::Types::HasMany(
                        'fieldName' => 'path',
                        'printableName' => __('Carpetas'),
                        'foreignModel' => 'Path',
                        'view' => '/NAS/View/Path',
                ),
    );
    my $dataTable =
    {
        'tableName' => 'Pool',
        'printableTableName' => __('Piscina'),
        'printableRowName' => __('Piscina'),
        'modelDomain' => 'NAS',
        'defaultActions' => ['add', 'del', 'editField', 'changeView' ],
        'tableDescription' => \@fields,
        'help' => '',
    };
    return $dataTable;
}
```

Figura 13. Método _table del modelo Pool.

El método **populate_partition** del modelo *Disk* permitirá que el administrador pueda seleccionar los discos o particiones con que cuenta la máquina a través del directorio /dev/. La importancia de este método es que permite seleccionar los discos de las piscinas que se crearán.

```
sub populate_partition
{
        my @result = ();

        my @res = EBox::Sudo::root("ls /dev");

        for my $part (@{$res[0]}) {
                if ($part =~ /^sd.\d+$/) {
                        push (@result, { value => $part, printableValue => $part });
                }
        }

    return \@result;
}
```

Figura 14. Método populate_partition del modelo Disk.

El método **menu** como precisamente el nombre lo indica es un menú opcional que se declara en la clase principal, a diferencia de los métodos anteriores. Este permite que cuando se le dé clic encima a NAS vaya directamente a la tabla con la información que porta la vista principal del módulo.

```
sub menu
{
    my ($self, $root) = @_;
    my $pools = new EBox::Menu::Item(
    'url' => 'NAS/View/Pool',
    'text' => __('NAS'));
    $root->add($pools);
}
```

Figura 15. Método menu.

Con lo que se ha hecho hasta el momento solamente se tiene una vista preliminar, sin embargo aún no está funcional. Para poder lograr el correcto funcionamiento del módulo hay que crear los métodos que harán posible la consistencia de la utilización del servicio en cuestión.

Luego de tenerlo todo listo se corren los siguientes comandos vía consola:

cd nas (esto es si no se está en este directorio) y **zentyal-package** para construir el paquete debian, que se encontrará en el directorio *nas/debs-ppa.* Entonces se procede a la instalación que puede hacerse dando doble clic sobre el *.deb* o usando *dpkg* mediante la línea de comandos **sudo dpkg -i debs-ppa/zentyal-nas_1.0_all.deb.**

Ya ha sido creado e instalado el módulo, ahora solamente hace falta ponerlo a prueba.

3.3 Validación funcional

Durante todo el ciclo de elaboración del software es preciso velar, controlar y garantizar su correcta calidad, haciendo posible el cumplimiento de los requerimientos que precisamente satisfacen las necesidades del cliente. Este aspecto debe estar presente de forma paralela desde la concepción del producto hasta la fase de producción del mismo. Para verificar lo antes mencionado se recurre a la realización de las **pruebas de software**.

3.3.1 Pruebas de software

Las pruebas de software es un concepto que a menudo es conocido como verificación y validación. Integra las técnicas de diseño de casos de prueba en una serie de pasos bien planificados que dan como resultado una correcta construcción del software. Entre algunas de las técnicas que se llevan a cabo para el proceso de prueba se encuentran las técnicas de caja negra y de caja blanca. [13]

Pruebas de caja negra

Prueba de caja negra es aquel elemento que se estudia desde el punto de vista de las entradas que recibe y las salidas o respuestas que produce, sin tener en cuenta su funcionamiento interno. En otras palabras, de una caja negra solamente interesará su forma de interactuar con el medio que le rodea (en ocasiones, otros elementos que también podrían ser cajas negras) entendiendo qué es lo que hace, pero sin dar importancia a cómo lo hace. Por tanto, de una caja negra deben estar muy bien definidas sus entradas y salidas, es decir, su interfaz; en cambio, no se precisa definir ni conocer los detalles internos de su funcionamiento.

Cuando se habla de un software, la prueba de caja negra se refiere a las pruebas que se llevan a cabo sobre la interfaz del mismo. Los métodos de prueba de la caja negra se centran en los requisitos funcionales del mismo e intentan encontrar errores de las siguientes categorías: 1) funciones incorrectas o ausentes; 2) errores de interfaz; 3) errores en estructuras de datos o en acceso a bases de datos externas; 4) errores de rendimiento y 5) errores de inicialización y terminación. [13]

A continuación se muestran los casos de pruebas para los CU fundamentales, representados a través de la matriz de datos, posibilitando de esta manera comprobar el correcto funcionamiento de la aplicación. Para la misma la V significa que es válido, la I que es invalido y NA indica que no es necesario proporcionar un valor del dato en este caso, ya que es irrelevante. Asimismo, se muestra la matriz parcial por los casos de uso en cuestión.

Matriz parcial del escenario para el caso de uso Gestionar piscinas

Nombre del Escenario	Flujo donde empieza
Escenario 1- Introducir una nueva piscina.	Flujo Básico
Escenario 2- No se puede introducir una nueva piscina, campos sin llenar.	Flujo Básico
Escenario 3- No se puede introducir una nueva piscina, conflicto con el nombre.	Flujo Básico
Escenario 4- No se puede introducir una nueva piscina, no se ha añadido al menos un disco.	Flujo Básico
Escenario 5- Eliminar una piscina.	Flujo Básico

Matriz de caso de prueba para el caso de uso Gestionar piscinas

ID del caso de prueba	Escenario/Condición	Nombre de la piscina	Relación entre discos	Discos	Carpetas	Resultado esperado
RC1	Escenario 1- Introducir una nueva piscina.	V	V	V	N/A	Visualiza una nueva piscina en la lista de piscinas.
RC2	Escenario 2- No se puede introducir una nueva piscina, campos sin llenar.	I	N/A	V	N/A	Muestra un mensaje de error, vuelta al paso 1.
RC3	Escenario 3- No se puede introducir una nueva piscina, conflicto con	I	N/A	N/A	N/A	Muestra un mensaje de error, vuelta al paso 1.

	el nombre.					
RC4	Escenario 4- No se puede introducir una nueva piscina, no se ha añadido al menos un disco.	N/A	N/A	I	N/A	Muestra un mensaje de error, vuelta al paso 1.
RC5	Escenario 5- Eliminar una piscina.	N/A	N/A	N/A	N/A	Elimina la piscina seleccionada.

Matriz de caso de prueba con los valores para los datos (CU Gestionar piscinas)

ID del caso de prueba	Escenario/ Condición	Nombre de la piscina	Relación entre discos	Discos	Carpetas	Resultado esperado
RC1	Escenario 1- Introducir una nueva piscina.	Yani	Ninguna.	sda3	N/A	Visualiza una nueva piscina en la lista de piscinas.
RC2	Escenario 2- No se puede introducir una nueva piscina, campos sin llenar.	Hay campos sin llenar.	Discos en espejos.	sda2 sda4	N/A	Muestra un mensaje de error: Falta argumento: Nombre.
RC3	Escenario 3- No se puede	Se introduce el	Discos en acoplamien	N/A	N/A	Muestra un mensaje

	introducir una nueva piscina, conflicto con el nombre.	nombre de una piscina existente: Yani.	to.			de error: Nombre ya existente: Yani.
RC4	Escenario 4- No se puede introducir una nueva piscina, no se ha añadido al menos un disco.	N/A	N/A	No se añaden discos a la piscina.	N/A	Muestra un mensaje de error: Debe insertar al menos un disco.
RC5	Escenario 5- Eliminar una piscina.	N/A	N/A	N/A	N/A	Piscina eliminada del sistema..

Matriz parcial del escenario para el caso de uso Adicionar discos

Nombre del Escenario	Flujo donde empieza
Escenario 1- Introducir un nuevo disco.	Flujo Básico
Escenario 2- No se puede introducir un nuevo disco, dispositivo ocupado o en uso.	Flujo Básico
Escenario 3- No se puede introducir un nuevo disco, no hay discos disponibles.	Flujo Básico

Matriz de caso de prueba para el caso de uso Adicionar discos

ID del caso de prueba	Escenario/Condición	Nombre de la partición/disco	Resultado esperado
RC1	Escenario 1- Introducir un nuevo disco.	V	Visualiza un nuevo disco en la lista de discos.

RC2	Escenario 2- No se puede introducir un nuevo disco, dispositivo ocupado o en uso.	I	Muestra un mensaje de error, vuelta al paso 1.
RC3	Escenario 3- No se puede introducir un nuevo disco, no hay discos disponibles.	I	Muestra un mensaje de error, vuelta al paso 1.

Matriz de caso de prueba con los valores para los datos (CU Adicionar discos)

ID del caso de prueba	Escenario/Condición	Nombre de la partición/disco	Resultado esperado
RC1	Escenario 1- Añadir un nuevo disco.	sda3	Visualiza un nuevo disco en la lista de discos.
RC2	Escenario 2- No se puede añadir un nuevo disco, dispositivo ocupado o en uso.	Se inserta un nuevo disco que está ocupado o se está usando.	Muestra un mensaje de error: Dispositivo ocupado o en uso.
RC3	Escenario 3- No se puede añadir un nuevo disco, no hay discos disponibles.	No hay discos disponibles.	No se pueden seleccionar discos, pues ya todos están ocupados.

Matriz parcial del escenario para el caso de uso Gestionar carpetas

Nombre del Escenario	Flujo donde empieza
Escenario 1- Introducir una nueva carpeta.	Flujo Básico
Escenario 2- No se puede introducir una nueva carpeta, conflicto de repetición con el nombre.	Flujo Básico
Escenario 3- Modificar una carpeta.	Flujo Básico
Escenario 4- No se puede modificar el nombre, conflictos de	Flujo Básico

repetición.	
Escenario 5- Eliminar una carpeta.	Flujo Básico

Matriz de caso de prueba para el caso de uso Gestionar carpetas

ID del caso de prueba	Escenario/Condición	Nombre de la carpeta	Comentario	Resultado esperado
RC1	Escenario 1- Introducir una nueva carpeta.	V	N/A	Visualiza una nueva carpeta en la lista de carpetas.
RC2	Escenario 2- No se puede introducir una nueva carpeta, conflicto de repetición con el nombre.	I	N/A	Muestra un mensaje de error, vuelta al paso 1.
RC3	Escenario 3- Modificar una carpeta.	V	V	Visualiza la carpeta modificada en la lista de carpetas o una carpeta de reemplazando la anterior.
RC4	Escenario 4- Eliminar una carpeta.	N/A	N/A	Elimina la carpeta seleccionada.

Matriz de caso de prueba con los valores para los datos (CU Gestionar carpetas)

ID del caso de prueba	Escenario/Condición	Nombre de la carpeta	Comentario	Resultado esperado
RC1	Escenario 1- Introducir una nueva carpeta.	Docencia	N/A	Visualiza una nueva carpeta en la lista de carpetas.

RC2	Escenario 2- No se puede introducir una nueva carpeta, conflicto de repetición con el nombre.	Introduce un nombre ya existente: Docencia.	N/A	Muestra un mensaje de error: Carpeta ya existente: Docencia.
RC3	Escenario 3- Modificar una carpeta.	Docencia	Carpeta destinada a materiales docentes.	Visualiza el comentario modificado o una carpeta de reemplazo por la anterior en la lista de carpetas.
RC4	Escenario 4- Eliminar una carpeta.	N/A	N/A	Elimina la carpeta seleccionada.

Luego de haber sido realizadas las pruebas, teniendo en cuenta el nivel de complejidad de la aplicación con respecto a los requisitos funcionales planteados para la misma, se obtuvo como resultado una aplicación donde no existen no conformidades.

3.4 Conclusiones del capítulo

En el presente capítulo se realizó la implementación del sistema, para ello se llevó a cabo el diseño de los principales componentes y sus relaciones, mediante la modelación del Diagrama de Despliegue. Posteriormente se mostró el código de algunos métodos de la aplicación. Para realizar la validación del sistema se definió usar la técnica de caja negra la cual fue aplicada en el diseño de pruebas, mostrando así las entradas y salidas válidas de la aplicación según los requerimientos, sin arrojar no conformidades, obteniendo como resultado que el sistema se comporte de la manera esperada.

Conclusiones

Al concluir el presente trabajo:

- Se realizó un estudio del estado del arte sobre herramientas para la implementación de servicio NAS.
- Se obtuvo una aplicación que facilita la implementación del servicio NAS, cumpliendo con los objetivos planteados.
- Se validó la solución propuesta mediante casos de prueba de caja negra, obteniéndose resultados satisfactorios.
- Se documentó el proceso de construcción de un módulo para la plataforma de desarrollo Zentyal, para las nuevas generaciones.

Dando así cumplimiento al objetivo de esta investigación. Además se estableció comunicación con la comunidad internacional de desarrolladores de Zentyal y ZFS, posibilitando mayor visibilidad y prestigio para el proyecto productivo Nova. El resultado de esta investigación no solo va a posibilitar que la empresa GEDEME comercialice servidores de almacenamiento, sino que comienza un camino, el cual los futuros desarrolladores del proyecto Nova, transitarán para darle un mayor valor agregado a la distribución maximizando sus niveles de Seguridad, Sostenibilidad, Soberanía y Socio-Adaptabilidad.

Recomendaciones

El análisis, diseño e implementación del módulo se efectuó de forma satisfactoria, sin embargo, con la finalidad de alcanzar mejores resultados se recomienda:

- Continuar con la implementación del módulo hasta llegar a una interfaz de administración que no dependa de Samba.
- Incluir como antivirus el cubano SAV (Unix)

Referencias Bibliográficas

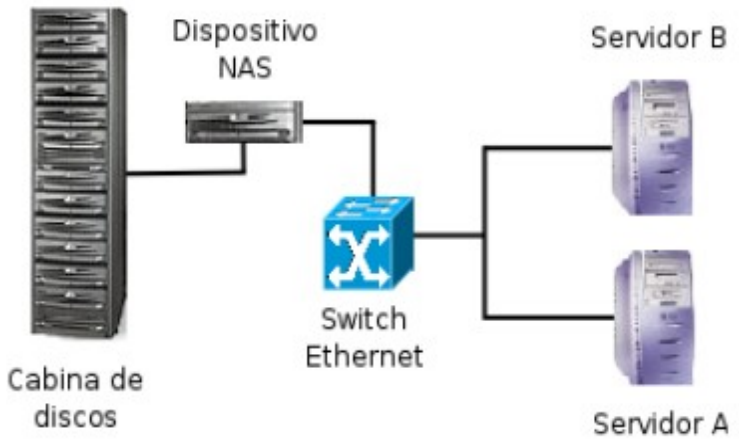
1. Red Hat Enterprise Linux 4: Introducción a la administración de sistemas. *Capítulo 5. Administración del Almacenamiento* [Citado el 25 de noviembre de 2011] http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-isa-es-4/s1-storage-adv.html
2. Sistemas de archivos. *Capítulo 6. Utilizando Discos y Otros Medios de Almacenamiento.* [Citado el 3 de marzo de 2012] http://www.ibiblio.org/pub/Linux/docs/LDP/system-admin-guide/translations/es/html/ch06s08.html
3. Zettabyte File System (ZFS) [Citado el 3 de marzo de 2012] http://ylvy.net/blog/?p=467
4. The Perl Programming Language [Citado el 30 de novimebre de 2011] http://www.perl.org/
5. Free Software Magazine. *gedit: a powerful, underrated text editor for everybody.* [Citado el 18 de mayo de 2012] http://www.freesoftwaremagazine.com/articles/gedit_powerful_underrated_text_editor_everybody
6. Importación de la Configuración [Citado el 18 de noviembre de 2011] http://doc.zentyal.org/es/develop.html
7. Augeas. *Lenses* [Citado el 16 de febrero de 2012] http://augeas.net/docs/lenses.html
8. Modelo de Dominio [Citado el 5 de marzo de 2012] http://migueljaque.com/index.php/tecnicas/tecnicasmodnegocio/37-modelado_negocio/46-modelo-de-dominio?tmpl=component&print=1&page=
9. **Egdamar.** [En línea] [Citado el 12 de mayo de 2012] http://egdamar877.blogspot.com/2009/05/expocicion.html
10. **García, Joaquín.** UML: Casos de Uso. Use case. *Desarrollo de Software Orientado a Objetos.* 27 de noviembre de 2003 [Citado el 5 de marzo de 2012] http://www.ingenierosoftware.com/analisisydiseno/casosdeuso.php
11. **Larman, Craig.** UML y Patrones. s.l. : Indiana, 2004.
12. **Lago, Ramiro.** Patrón "Data Access Object" abril de 2007 [Citado el 8 de marzo de 2012] http://www.proactiva-calidad.com/java/patrones/DAO.html
13. **Esperanza Fernández, Jorge Luis.** “Sistema para el Análisis de Sensibilidad en la plataforma BioSyS”. Universidad de las Ciencias Informáticas, 2011.

Bibliografía

1. eBox module development guide. *Release trunk.* July 08, 2009.
2. Zentyal Module Development Tutorial. [Citado el 1 de febrero de 2012] http://trac.zentyal.org/wiki/Documentation/Community/Document/Development/Tutorial
3. **Hernández Meléndrez, Edelsys**. *Cómo escribir una tesis*. Escuela Nacional de Salud Pública, 2006. [Citado el 30 de enero 2012].
4. **Arzuaga Ruíz, Yaniel y Martínez Casero, Adrian.** *Personalización de un sistema GNU/Linux para administración de Redes de Área de Almacenamiento.* Universidad de las Ciencias Informáticas, 2011.
5. **Esperanza Fernández, Jorge Luis.** *Sistema para el Análisis de Sensibilidad en la plataforma BioSyS.* Universidad de las Ciencias Informáticas, 2011.
6. **Larman, Craig.** UML y Patrones. s.l. : Indiana, 2004.
7. **Hernández León, Rolando Alfredo y Coello González, Sayda.** *El Proceso de Investigación Científica.* Ciudad de La Habana: Editorial Universitaria, 2011.
8. **Lam Díaz, Rosa María.** *Metodología para la confección de un proyecto de investigación.* Instituto de Hematología e Inmunología. Ciudad de La Habana, 2005.
9. **Galán Ortiz, David.** *ZFS Zettabyte.* Spain OpenSolaris Users Group. [Citado el 3 de junio de 2012]

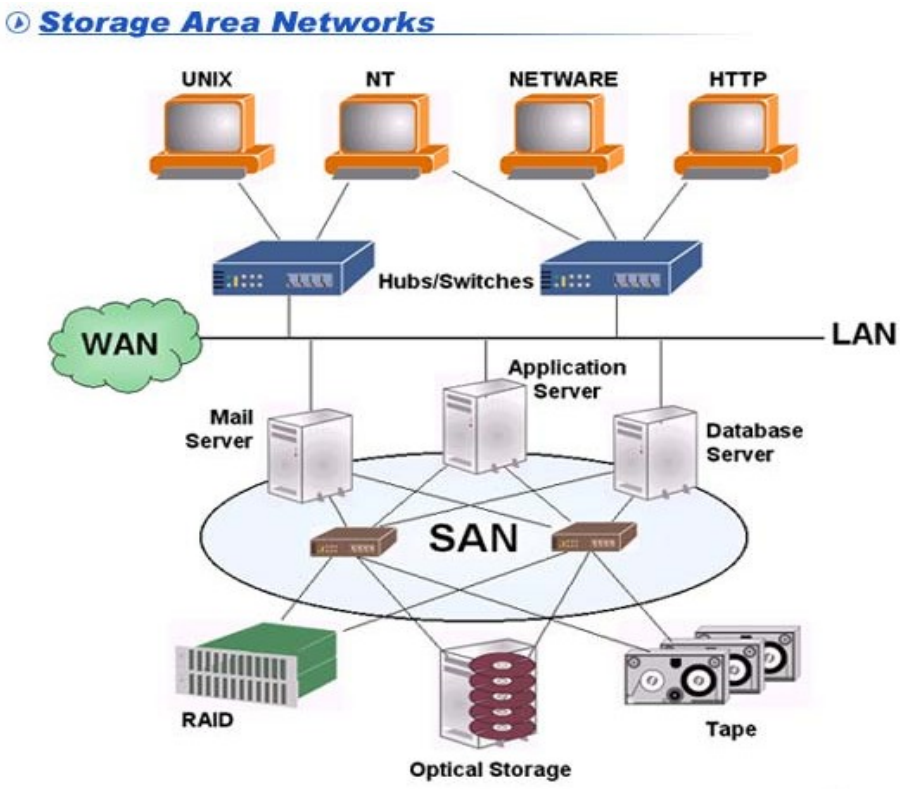
Anexos

Anexo #1



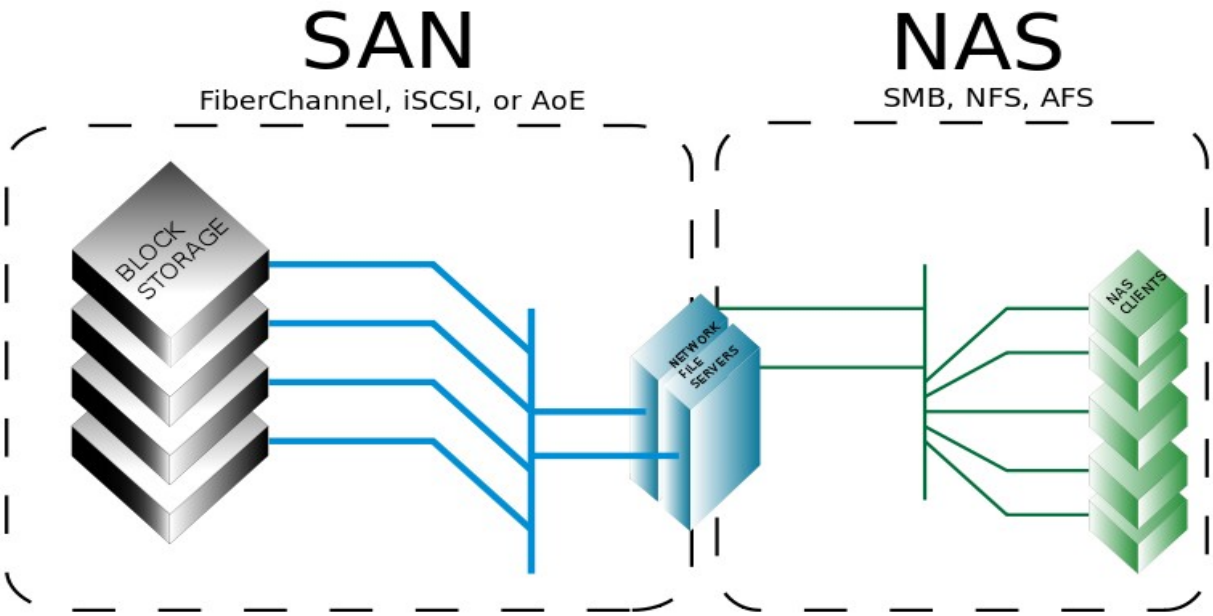
Anexo 1. Redes NAS.

Anexo #2



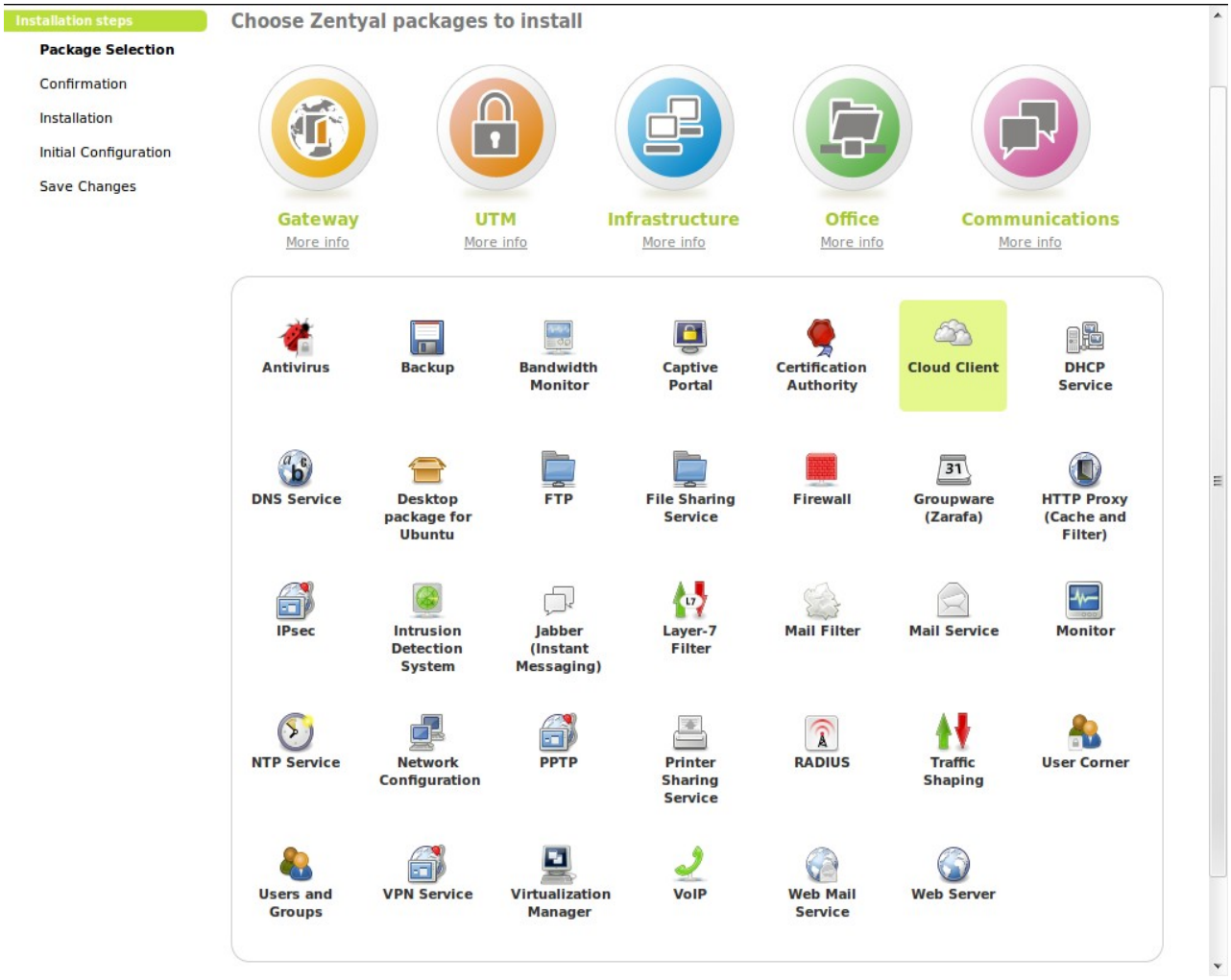
Anexo 2. Redes SAN.

Anexo #3



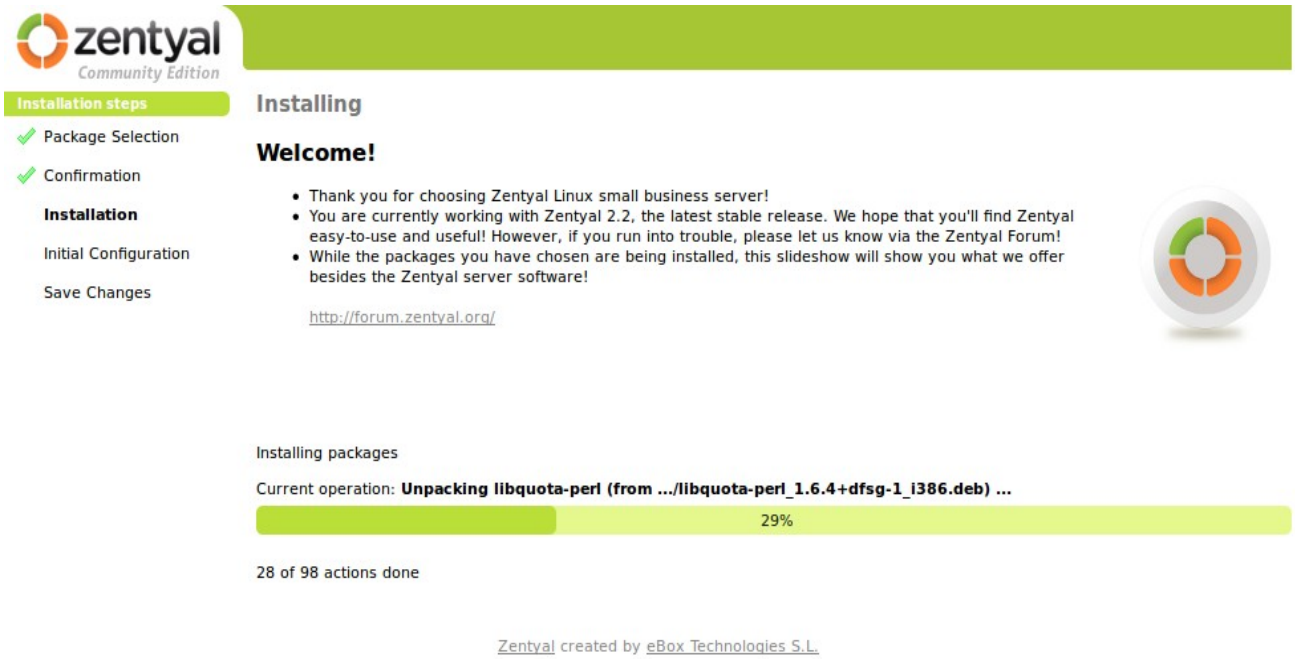
Anexo 3. Diferencias entre las redes SAN y NAS.

Anexo #4



Anexo 4. Selección de los paquetes de Zentyal para la instalación.

Anexo #5

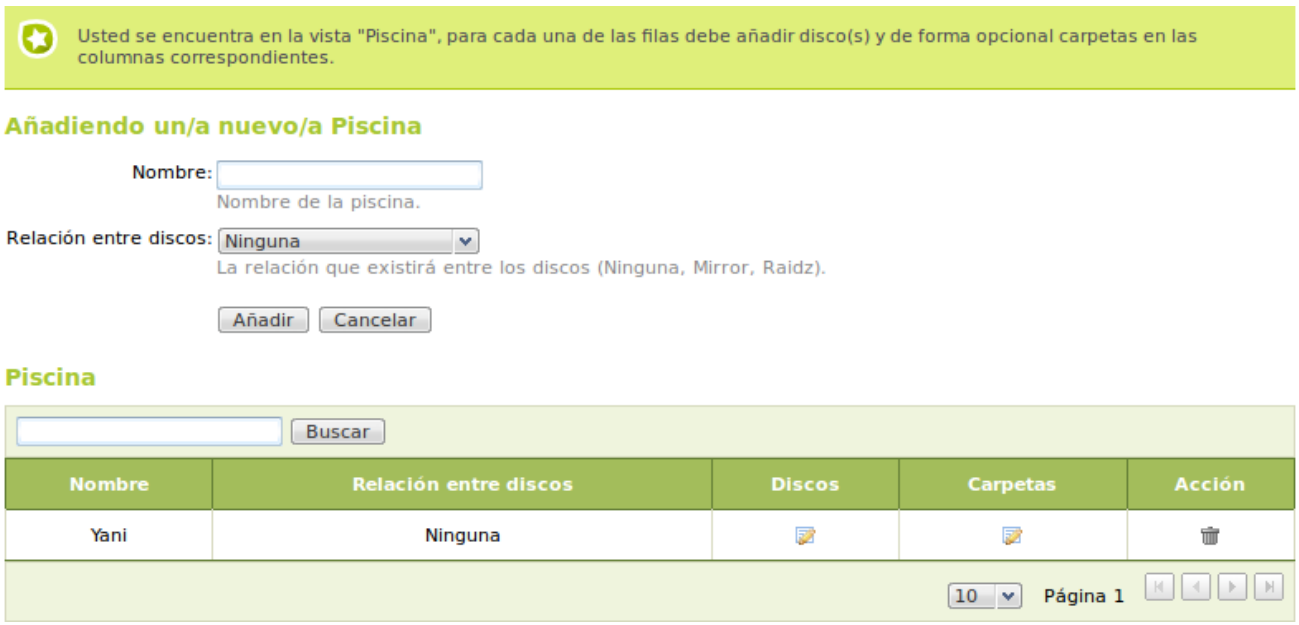


Anexo 5. Instalación de los paquetes seleccionados en Zentyal.

Anexo #6

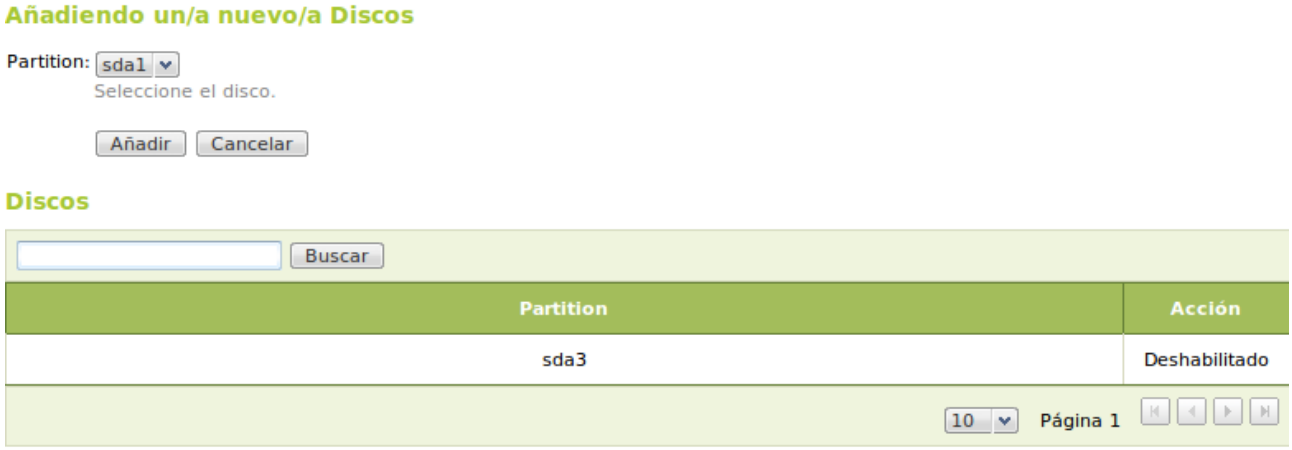
Anexo 6. Instalación del módulo NAS.

Anexo #7



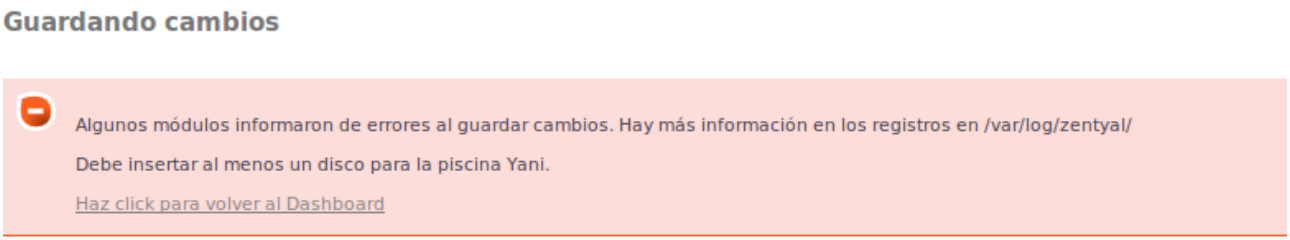
Anexo 7. Añadiendo una nueva piscina.

Anexo #8



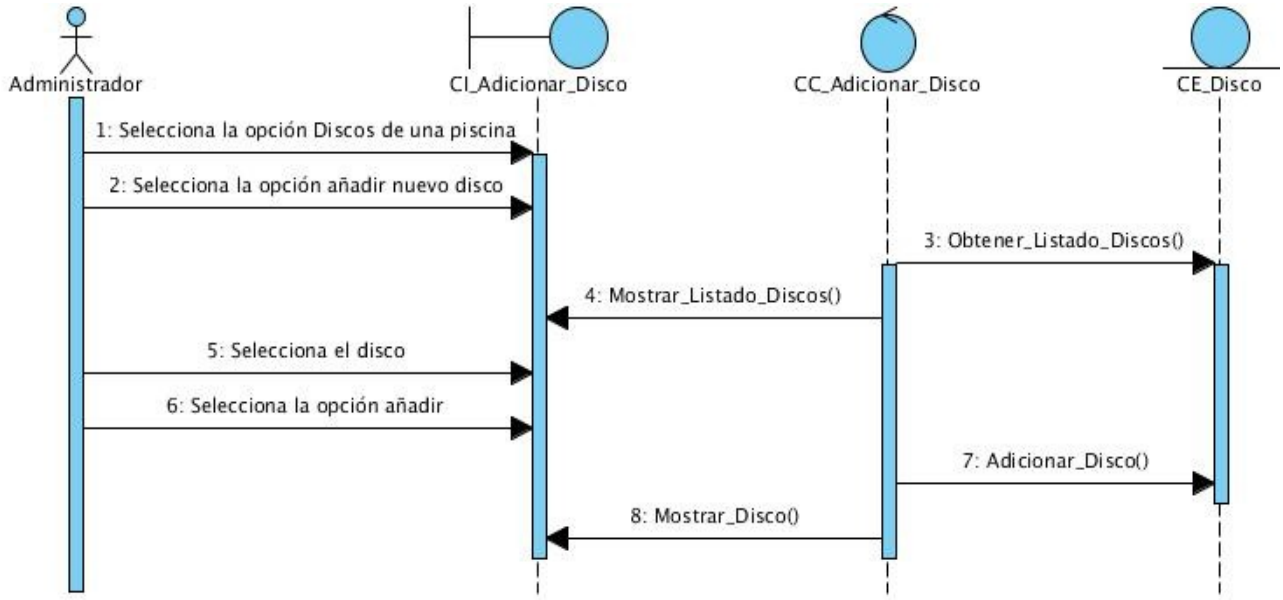
Anexo 8. Añadiendo discos a la piscina.

Anexo #9



Anexo 9. Mensaje de error al no escoger discos para las piscinas.

Anexo 10

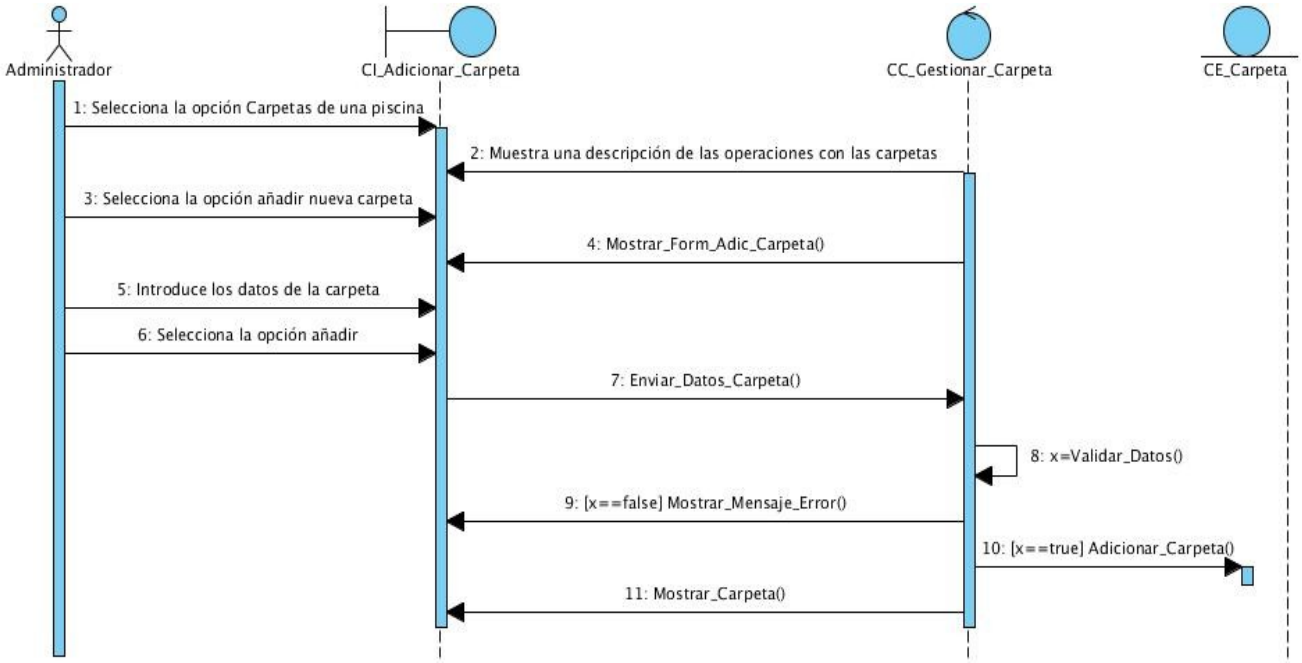


Anexo 10. Diagrama de secuencia Adicionar Disco.

Anexo #11

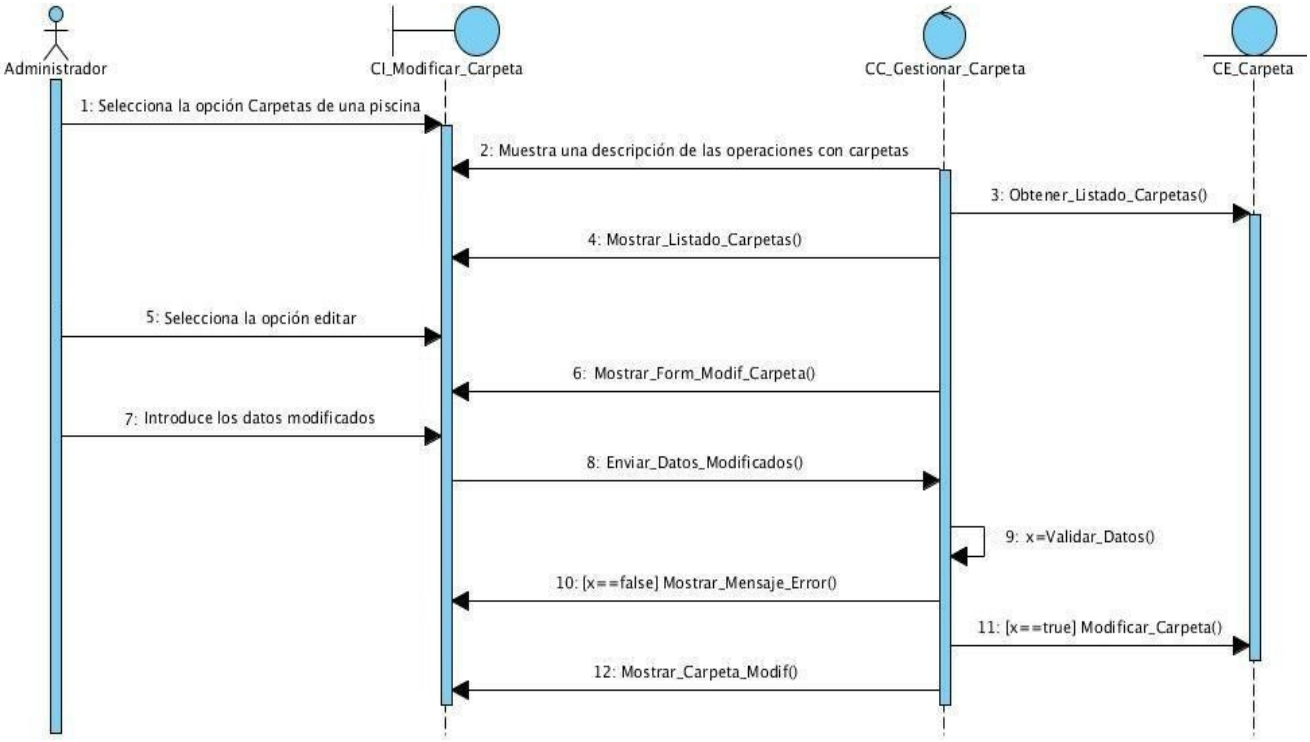
Anexo 11. Añadiendo carpeta a la piscina.

Anexo #12



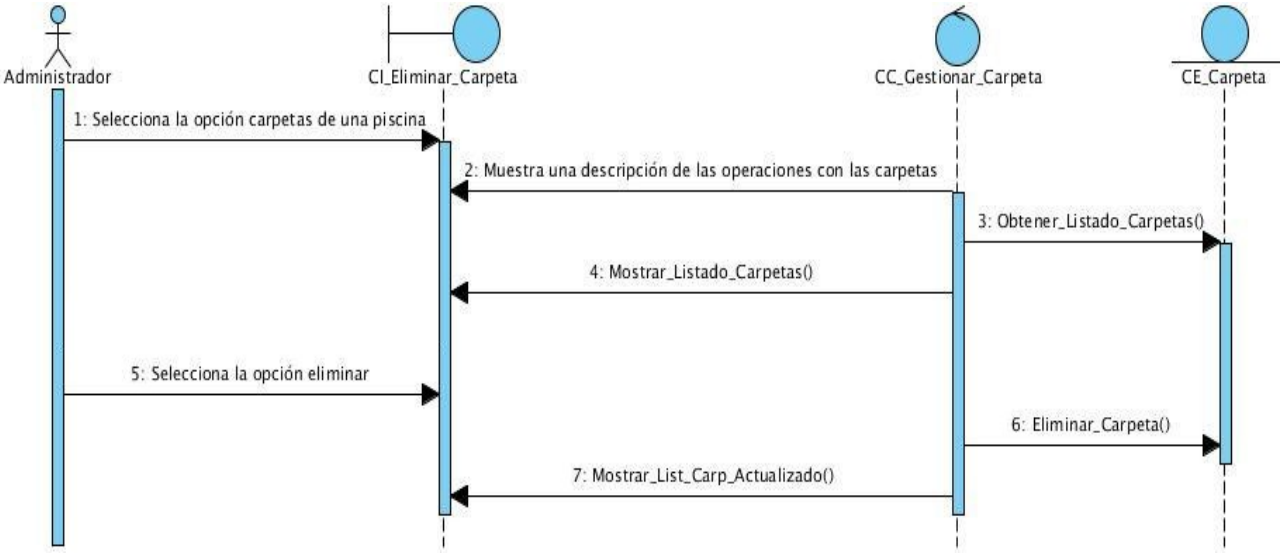
Anexo 12. Diagrama de secuencia Adicionar Carpeta.

Anexo #13



Anexo 13. Diagrama de secuencia Modificar Carpeta.

Anexo #14



Anexo 14. Diagrama de secuencia Eliminar Carpeta.