UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

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

MÓDULO DE HMAST PARA LA ADMINISTRACIÓN Y MIGRACIÓN HACIA SAMBA4 DEL SERVICIO DIRECTORIO ACTIVO

AUTORES

Felipe González Santiago

Yosel Lázaro Vera González

TUTORES

Mtr. Amaury Viera Hernández

Ing. Yadiel Pérez Villazón

Ing. Miriela Velázquez Arias

La Habana, junio de 2016

"Año 58 de la Revolución"

Declaración de autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de la 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 _____

Felipe González Santiago

_____

Firma del autor

Yosel Lázaro Vera González

_____

Firma del autor

Amaury Viera Hernández

_____

Firma del tutor

Yadiel Pérez Villazón

_____

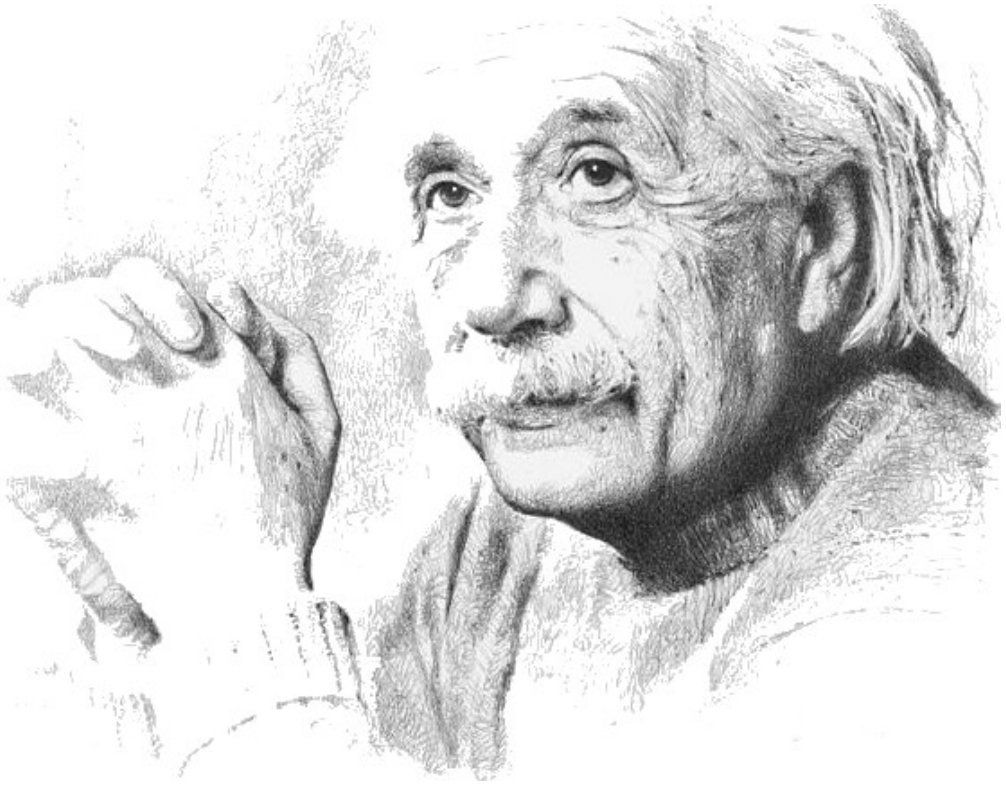
Firma del tutor

Miriela Velázquez Arias

_____

Firma del tutor

¿Por qué esta magnífica tecnología científica, que ahorra trabajo y tiempo y nos hace la vida más fácil, nos aporta tan poca felicidad?
La respuesta es esta, simplemente: porque aún no hemos aprendido a usarla con tino.



Albert Einstein

Dedicatoria

A Mamá (mi abuela) por la perspicacia con que me dejó beber de la sabiduría popular.
Felipe.

A mi mamá por ser la mujer más fuerte que conozco y por todo lo que me ha enseñado, y a mi familia que sin su apoyo nada de esto hubiera sido posible.
Yosel

Agradecimientos

A Amaury, Yadiel y Miriela, los mejores tutores y excelentes personas, por no aguantarse las ganas de ayudar.
A los linuxeros del Centro de Software Libre, por la constancia y la buena energía.
A los profes, que se tomaron en serio lo de educar.
Felipe y Yosel.

A mi familia, por allanar el camino.
A mis compañeros, por las lecciones de humildad.
A Yosel, por su consagración y por saber cuándo ceder y cuándo convencer.
A la familia RCD, por darme la oportunidad de recibir las recompensas que se obtienen cuando se hace lo que se ama.
Felipe.

A mi papá que me ayudó mucho a mí y a mi mamá para que pudiera lograr todas mis metas.
A toda mi familia que siempre me apoyó en todo, mi hermanos, mis primos, mis tíos, todos aportaron a mi formación como mejor persona y profesional.
A todas mis amistades que durante estos cinco años siempre nos ayudamos y nos mantuvimos unidos para rebasar los retos de la carrera.
A Carlos e Iguelit y a sus familias que me acogieron como uno más de su familia.
A los trabajadores del complejo CIMEX de Jovellanos que sin dudarlo siempre prestaron su ayuda a mi mamá.
A mi compañero de tesis por haber confiado en mí para realizar la tarea más importante de nuestra carrera.
Yosel.

RESUMEN

Actualmente se evidencia una tendencia hacia la migración a software libre y código abierto. Samba4 es una alternativa libre al servicio Directorio Activo que posee una alta compatibilidad puesto que ha sido desarrollada a partir de las especificaciones de los protocolos ofrecidas por el equipo de Microsoft. Actualmente migrar los datos y servicios del Directorio Activo hacia Samba4 se realiza de forma manual mediante una interfaz de línea de comandos por lo que es un proceso lento y complejo. En el Centro de Software Libre de la Universidad de las Ciencias Informáticas se desarrolla la Herramienta para la Migración y Administración de Servicios Telemáticos, la cual debe permitir administrar desde plataformas libres los servicios telemáticos en las empresas cubanas. Sin embargo, esta herramienta no garantiza la administración del Directorio Activo, ni su migración hacia Samba4. El presente trabajo de diploma tiene como objetivo desarrollar un módulo para HMAST que permita la administración y migración hacia Samba4 del servicio Directorio Activo, garantizando la integridad de los datos. El desarrollo del sistema permitió informatizar el proceso de migración hacia Samba4, así como realizar una administración avanzada del servicio a través de una interfaz web.

Palabras clave: administración, Directorio Activo, migración, Samba4, servicio telemático.

Índice de contenido

INTRODUCCIÓN

El auge de las Tecnologías de la Información y la Comunicación (TIC) ha favorecido su incorporación en gran parte de las esferas de la sociedad. En Cuba se lleva a cabo un proceso de informatización que evidencia el uso de soluciones informáticas dirigidas a los procesos de negocio en las instituciones. En los Organismos de la Administración Central del Estado (OACE) resalta el uso de dichas soluciones sobre sistemas operativos privativos, específicamente Microsoft Windows.

La migración hacia plataformas GNU/Linux es de suma importancia debido a la alta disponibilidad y tiempo efectivo de funcionamiento; al aseguramiento de una mayor estabilidad, fiabilidad y seguridad así como a la reducción de los costos por licencias, tanto por el sistema operativo instalado como por las aplicaciones que se ejecutan en él [1]. Las vulnerabilidades que contienen los programas informáticos pueden ser explotadas y comprometer la información confidencial de una organización. El proceso de corrección de errores identificados por el usuario en el software privativo suele ser extremadamente engorroso y cuando se logra la reparación del error, se debe esperar hasta la siguiente versión del sistema operativo. Tal y como ha sido denunciado por personalidades como Julian Assange[1] y Edward Snowden[2], algunos fabricantes de software privativo han colaborado con agencias gubernamentales para incluir accesos secretos al software y así poder acceder a datos confidenciales de otras empresas, hechos que comprometen la seguridad de la información.

Por otro lado, el empleo de soluciones de código abierto permite disponer del código fuente, por lo que cualquier programador con la preparación necesaria puede realizar sus propias auditorías para comprobar que no exista código malicioso. Además disminuye la dependencia del cliente del fabricante en cuanto a requerimientos de soporte. Debido a la libertad que tiene el usuario de modificar el programa, es posible también la personalización, el mantenimiento y la adaptación del software a las necesidades del usuario final.

En tal sentido, la máxima dirección del país identificó las necesidades existentes en el sector de las TIC, y en el año 2004 el Consejo de Ministros de la República de Cuba anuncia el acuerdo 084, donde se orienta una migración paulatina de los OACE hacia aplicaciones de código abierto [2].

Los servicios telemáticos desempeñan un rol decisivo dentro de la infraestructura tecnológica de las entidades, por lo cual se les debe dedicar gran parte del esfuerzo durante el proceso de migración a

1 Julian Assange: es un programador, ciberactivista y periodista australiano. Fundador, editor y portavoz del sitio web WikiLeaks.

2 Edward Snowden: es un consultor tecnológico estadounidense, informante, antiguo empleado de la CIA y de la NSA.

código abierto. Un servicio telemático permite transferir información entre los usuarios y equipos de una misma red de telecomunicaciones empleando protocolos establecidos para los sistemas de interconexión abiertos (OSI[3]). Existen diversos servicios telemáticos tales como correo electrónico, DNS[4], DHCP[5], SSH[6] y Directorio Activo.

El Directorio Activo es la implementación de Microsoft de un servicio de directorio que almacena información acerca de una variedad de objetos en la red como usuarios, equipos y grupos [3]. Desempeña un papel relevante para la gestión centralizada de los recursos en una red distribuida de computadoras. Además, constituye el eje principal de autenticación de todos los servicios de la empresa. En una encuesta realizada en el año 2013 a diez instituciones cubanas se resume que, del total de las empresas encuestadas, el 89% emplea el servicio Directorio Activo como tecnología para la administración centralizada de los recursos de red y solo el 11% tecnologías libres, específicamente OpenLDAP [4].

Existen varias soluciones de código abierto que implementan un servicio de directorio mediante el protocolo LDAP[7] como son OpenLDAP, Samba4, 389-DS y Apache Directory Server. Entre ellas se destaca la herramienta de software libre Samba4, la cual integra el conjunto de tecnologías del Directorio Activo. Las tecnologías asociadas son SMB[8], DNS, LDAP, NTP[9] y Kerberos. Samba4 posee una alta compatibilidad con el Directorio Activo puesto que ha sido desarrollada a partir de las especificaciones de los protocolos ofrecidas por el equipo de Microsoft.

El proceso de migrar el servicio de Directorio Activo hacia Samba4 se realiza manualmente a través de una interfaz de línea de comandos, por lo que se requiere una gran cantidad de operaciones. Esto provoca dificultades tales como lentitud durante la ejecución de las tareas, aumento de la complejidad en el proceso de migración, mayor cobertura a la introducción de errores humanos e incluso la pérdida de información. Estos incidentes atrasan determinados procesos provocando que no se ejecuten en el momento deseado y por consiguiente, atentan contra el éxito de la migración de los datos y servicios.

La Universidad de las Ciencias Informáticas (UCI) tiene la misión de producir aplicaciones y servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación y servir de soporte a la industria cubana de la informática [5]. Dentro de su infraestructura docente-productiva se encuentra el

3 OSI: Sistemas de interconexión abiertos, del inglés *Open System Interconnection*.

4 DNS: Sistema de nombres de dominio, del inglés *Domain Name System*.

5 DHCP: Protocolo de Configuración Dinámica de Host, del inglés *Dynamic Host Configuration Protocol*.

6 SSH: Intérprete de Órdenes Seguro, del inglés *Secure Shell*.

7 LDAP: Protocolo Ligero de Acceso a Directorio, del inglés *Lightweight Directory Access Protocol.*

8 SMB: De las siglas en inglés *Server Message Block*. Protocolo que permite compartir archivos.

9 NTP: Protocolo de Tiempo de Red, del inglés *Network Time Protocol*.

Centro de Software Libre (CESOL), líder en los procesos de migración hacia tecnologías libres y de código abierto. Además, realiza acciones para alcanzar la independencia y soberanía tecnológicas en el país, lo cual se caracteriza por la “posibilidad de desarrollarse de forma autónoma en el campo de las TIC, teniendo total capacidad de decisión sobre las tecnologías y la forma en que se desarrollan y usan las mismas” [6].

El departamento de Servicios Integrales en Migración, Asesoría y Soporte (SIMAYS), perteneciente a CESOL, se encuentra proyectado hacia las siguientes líneas de trabajo: el desarrollo de estrategias, guías y planes de migración; las herramientas de soporte al proceso de migración; y la migración de los servicios telemáticos y aplicaciones de escritorio. Para dar cumplimiento a estas líneas se desarrolla la Herramienta para la Migración y Administración de Servicios Telemáticos (HMAST). Esta solución está dirigida a la administración y migración de los servicios telemáticos hacia plataformas libres de forma remota, adaptándose a las condiciones y especificidades de las empresas nacionales. HMAST cuenta con módulos para administrar los servicios DHCP, SSH, MySQL, Apache y Bacula; sin embargo, no garantiza la administración del servicio de Directorio Activo ni su migración hacia Samba4.

Teniendo en cuenta los siguientes elementos:

- La importancia del servicio de Directorio Activo para la administración centralizada de los recursos de la red en una institución.
- El amplio uso del Directorio Activo en las instituciones cubanas.
- La existencia de una alternativa libre (Samba4) con una gran compatibilidad con el Directorio Activo.
- La necesidad de investigar el mecanismo de migración del Directorio Activo que ofrece Samba4.
- La importancia de informatizar el mecanismo de migración de Samba4 para disminuir tanto el tiempo como la introducción de errores durante el proceso de migración de los datos y los servicios.
- La necesidad de investigar las herramientas existentes para gestionar el servicio Samba4 sobre GNU/Linux que puedan ser reutilizadas en HMAST.

Se define como **problema científico**: ¿Cómo transferir los datos almacenados en un servicio de Directorio Activo hacia Samba4 y administrarlos desde HMAST?

Se determina como **objeto de estudio** la administración y migración hacia plataformas libres del servicio Directorio Activo; enmarcándose el **campo de acción** en la administración y migración hacia Samba4 del servicio Directorio Activo.

Para dar solución al problema científico se establece como **objetivo general** del presente trabajo desarrollar un módulo para HMAST que permita la administración y migración hacia Samba4 del servicio Directorio Activo, garantizando la integridad de los datos. Luego se derivan los siguientes **objetivos específicos**:

- Caracterizar las herramientas de administración del servicio Directorio Activo.
- Establecer el mecanismo a utilizar en el proceso de migración del servicio Directorio Activo hacia Samba4.
- Diseñar un módulo para HMAST que permita la administración y migración hacia Samba4 del servicio Directorio Activo.
- Implementar un módulo para HMAST que permita la administración y migración hacia Samba4 del servicio Directorio Activo.
- Realizar pruebas internas y de aceptación al módulo.

Se formula la siguiente **idea a defender**: el desarrollo de un módulo de HMAST para la administración y migración hacia Samba4 del servicio Directorio Activo permitirá la gestión de los recursos de red garantizando la integridad de los datos.

Las **tareas de investigación** planteadas son:

- Caracterización de los aspectos teóricos asociados a la administración y migración hacia Samba4 del servicio Directorio Activo para adquirir el basamento teórico de la investigación.
- Análisis de sistemas informáticos que realizan la administración de servicios de directorio.
- Descripción del mecanismo de migración a emplear.
- Sistematización de la arquitectura de HMAST.
- Definición de los requisitos funcionales y no funcionales.
- Diseño e implementación de las clases y métodos que den solución a los requisitos definidos.
- Diseño y ejecución de casos de prueba a las funcionalidades implementadas.

Para facilitar el cumplimiento del objetivo propuesto y de las tareas de investigación se emplean métodos teóricos y empíricos de la investigación científica. Los métodos teóricos que se emplean son el Histórico-lógico para el estudio, en trabajos anteriores, de los antecedentes de situaciones semejantes al problema

de la investigación y sus posibles variantes de solución. El método Analítico-sintético se pone en práctica para el estudio de los elementos del problema planteado, profundizando en las características de cada uno, para luego sintetizarlos y elaborar la propuesta de solución de la investigación. El método de Modelación se emplea para elaborar abstracciones a través de diagramas, como el diagrama de clases y diagrama de despliegue. Se emplea el método empírico experimental para comprobar la integridad de los datos una vez realizada la migración del servicio Directorio Activo hacia Samba4.

El presente documento está estructurado en introducción, tres capítulos, conclusiones generales, referencias bibliográficas, anexos y glosario de términos. En el primer capítulo “Fundamentación Teórica” se presenta el basamento teórico de la presente investigación. Se realiza un análisis bibliográfico basándose en los métodos teóricos antes expuestos, que permite caracterizar el servicio Directorio Activo y Samba4 como alternativa libre. También se caracteriza la herramienta HMAST y su arquitectura, así como la metodología de desarrollo a emplear.

En el segundo capítulo “Diseño del módulo” se especifican las características principales de la propuesta de solución y se definen cuáles son las funcionalidades que se implementarán. Se presentan los prototipos de interfaz de usuario, las historias de usuario correspondientes a las funcionalidades definidas y los patrones de diseño que se utilizarán durante la implementación.

En el tercer capítulo “Implementación y pruebas al módulo” se documentan los artefactos asociados a la implementación de la propuesta de solución y se llevan a cabo los casos de prueba que se aplicarán sobre el sistema.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

En el presente capítulo se realiza un estudio de los servicios de directorio, centrándose en el servicio Directorio Activo. Se analizan diversas herramientas que permiten administrar un servicio de directorio tanto de la parte del servidor como del cliente, para valorar su posible integración en el módulo a desarrollar. Se detallan las herramientas y tecnologías que serán empleadas para el desarrollo del módulo en cuestión, como son el lenguaje de programación y la metodología de desarrollo de software. También se caracteriza HMAST así como su arquitectura y se exponen algunas consideraciones a tener en cuenta para integrar el módulo a esta herramienta.

1.1 Directorio Activo

El Directorio Activo es un servicio de directorio de red de propósito general. Constituye la implementación de Microsoft de un servicio de directorio, que almacena información acerca de una variedad de objetos en la red como usuarios, equipos, grupos e impresoras [3]. El Directorio Activo es un medio de organizar, controlar y administrar centralizadamente el acceso a los recursos de la red por parte de los usuarios y las aplicaciones, de manera que se convierte en el eje principal de autenticación de todos los servicios de la empresa. Además permite una administración eficiente de estos recursos y ayuda a monitorizar y localizar los servicios.

Este servicio posee un conjunto de características que brindan varias facilidades para la administración de los recursos de una organización de pequeño, mediano o gran tamaño, entre las cuales destacan [7]:

- Escalabilidad. Permite el crecimiento de los recursos de la organización y el soporte de un elevado número de objetos.
- Extensibilidad. Permite personalizar las clases y objetos que están definidas según las necesidades propias a través de la extensión del esquema.
- Flexibilidad. Permite reflejar la organización lógica y física de la empresa donde se instala y que varios dominios se conecten en una estructura de árbol o de bosque.
- Multimaestro. No distingue entre controladores de dominios primarios o secundarios, de modo que cualquier controlador de dominio puede procesar cambios del directorio.
- Seguridad. Cada dominio representa un límite de la administración y por tanto de la seguridad. El acceso a los objetos dentro de cada dominio se controla mediante Entradas de Control de Acceso

(ACE, del inglés *Access Control Entries*) contenidas en Listas de Control de Acceso (ACL, del inglés *Access Control Lists*).

1.1.1 Tecnologías asociadas al Directorio Activo

El Directorio Activo integra un conjunto de tecnologías informáticas tales como LDAP, Kerberos, DNS, SMB y NTP. A continuación se hace referencia a cada una de ellas y a su dependencia con el Directorio Activo.

- **Protocolo Ligero de Acceso a Directorio**

El Protocolo Ligero de Acceso a Directorio (LDAP, del inglés *Lightweight Directory Access Protocol*) es un protocolo a nivel de aplicación que concede el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red. Normalmente se utiliza para acceder a la información almacenada de usuarios, grupos y equipos de una organización [8].

Es un protocolo estándar de la industria, establecido por el Grupo de Trabajo de Ingeniería de Internet (IETF, del inglés *Internet Engineering Task Force*), que permite a los usuarios consultar y actualizar la información en un servicio de directorio [3].

LDAP estructura la información en forma de un árbol, de forma análoga a la estructura de directorio de UNIX. La definición detallada y las especificaciones técnicas de LDAP están disponibles en las RFC[10] 2251 y 3377 respectivamente.

LDAP es el protocolo de acceso principal para Directorio Activo y especifica que un objeto sea representado por una serie de componentes de dominio, unidades organizativas y nombres comunes, creando una ruta de nomenclatura LDAP en el Directorio Activo. Existen varias nomenclaturas para identificar los objetos dentro del Directorio Activo.

Nombre Completo (DN, del inglés *Distinguished Name*). Es un identificador único de un objeto dentro del Directorio Activo. Identifica el dominio donde se encuentra el objeto y la ruta de acceso completa por la que se llega hasta él. A continuación se muestra un ejemplo.

dn: CN=Bertha Doe, CN=Users, DC=cesol, DC=cu

Nombre Completo Relativo (RDN, del inglés *Relative Distinguished Name*). Es un par atributo-valor usado en el DN de un objeto. Representa el nombre único de un elemento hijo en relación con su padre.

10 RFC: Solicitud de comentarios, del inglés *Request For Comments*. Documento que contiene una propuesta para una nueva tecnología, información acerca del uso de tecnologías y/o recursos existentes.

El RDN de un elemento hijo en combinación con el nombre completo (FQDN) de los padres constituye el FQDN del hijo [3].

Componente de Dominio (DC, del inglés *Domain Component*). Es un componente del nombre DNS del dominio. El nombre de dominio DNS cesol.cu, por ejemplo, quedaría representado como DC=cesol, DC=cu.

Identificador Global Único (GUID, del inglés *Globally Unique Identifier*). Genera una cadena de 128 bits que es usada por el Directorio Activo para buscar y replicar información. Es único para cada objeto en el bosque de dominio, y no cambia nunca. Está formado por el identificador de seguridad del dominio (prefijo) y un identificador relativo único. Se almacena en el atributo nombrado objectGUID.

Nombre de Usuario Principal (UPN, del inglés *User Principal Name*). Permite el ingreso abreviado a un recurso o un directorio de la red. Su sintaxis es objetodered@dominio.

- **Kerberos**

Kerberos es un sistema de autenticación que permite a dos partes de un sistema intercambiar información privada a través de una red mediante la asignación de una clave única, llamada *ticket*, a cada usuario que inicia sesión en la red [3]. Es el protocolo utilizado por el Directorio Activo para la autenticación de usuarios y máquinas en el dominio.

Para permitir la autenticación de los usuarios en el dominio, Kerberos v5 exige que los equipos clientes estén sincronizados en tiempo con el servidor de Directorio Activo, con una tolerancia máxima por defecto de hasta cinco minutos. Por esta razón todos los clientes deben estar sincronizados con el controlador de dominio, para lo cual se recomienda el uso del protocolo NTP.

Kerberos impide que las claves sean enviadas a través de la red, con el consiguiente riesgo de su divulgación. Además centraliza la autenticación de usuarios, manteniendo una única base de datos de usuarios para toda la red.

- **Sistema de nombres de dominio**

El sistema de nombres de dominio (DNS) permite la resolución de los nombres de los equipos a direcciones IP y viceversa. El correcto funcionamiento de este servicio es esencial para el Directorio Activo teniendo en cuenta los siguientes elementos [9]:

- Conocer la lista de Controladores de Dominio que pueden ser usados para iniciar sesión en el dominio.

- Los clientes buscan a los Controladores de Dominio a través de DNS para tareas de administración e inicios de sesión.
- Los Controladores de Dominio se encuentran entre sí a través del servicio DNS. Esto permite la replicación del Directorio Activo entre ellos.
- Los Controladores de Dominio encuentran a los servidores que tienen la funcionalidad de Catálogo Global usando el servicio DNS.

➢ **Protocolo de tiempo de red**

Debido a que el Directorio Activo es un servicio que emplea el protocolo Kerberos, requiere una correcta sincronización de tiempo entre los equipos servidores y los clientes. El controlador de dominio puede funcionar como servidor de tiempo empleando el protocolo NTP. El servidor con el rol emulador de PDC es considerado como el proveedor de tiempo predeterminado en un dominio [10].

1.1.2 Conceptos asociados al Directorio Activo

➢ **Estructura lógica**

El Directorio Activo organiza los recursos mediante una estructura lógica que permite localizar un recurso por su nombre y no por su localización física. Esto brinda una mayor transparencia y flexibilidad a los usuarios. A continuación se describen los componentes lógicos de este servicio de directorio.

El **dominio** constituye la unidad central de la estructura lógica del Directorio Activo. Está formado por un conjunto de objetos, dígase usuarios y equipos que comparten un espacio de nombres común. El dominio representa un límite de administración para la autenticación, la replicación, las políticas de grupo y la delegación de tareas administrativas [3].

La **unidad organizativa** es un contenedor que permite organizar los objetos del directorio, agrupándolos de forma coherente y creando vistas más pequeñas y manejables. A nivel de administración permite agrupar objetos con los mismos requerimientos y delegar sobre ellos tareas administrativas. A nivel de políticas de grupo permite establecer distintos detalles de seguridad a cada unidad organizativa.

La agrupación de uno o más dominios de forma jerárquica que comparten un espacio de nombres continuo se conoce como **árbol de dominio** [3]. Los administradores de un dominio padre no son automáticamente administradores del dominio hijo y el conjunto de políticas de un dominio padre no se

aplica automáticamente a los dominios hijos.

Un **bosque** es un conjunto de uno o varios árboles de dominio que comparten información del directorio común. Un bosque establece un límite de seguridad y administrativo para todos los objetos que se encuentran dentro de los dominios que pertenecen a ese bosque [3].

- **Estructura física**

La estructura física del Directorio Activo se utiliza para configurar y administrar el tráfico de red y define dónde y cuándo se produce el tráfico de inicio de sesión. Los dos componentes físicos son los controladores de dominio y los sitios.

Un **controlador de dominio** es un equipo servidor donde se ejecuta el servicio Directorio Activo y que almacena una copia del directorio del dominio. Además, administra los procesos de inicio de sesión, autenticación y búsquedas en directorios de usuarios. Un controlador de dominio solo puede admitir un dominio, por otro lado un dominio debe disponer de más de un controlador para proporcionar la disponibilidad y la tolerancia a fallos adecuadas [11].

Un **sitio** es un conjunto de una o más subredes TCP/IP[11] que están conectadas por un vínculo de alta velocidad. Permite configurar la topología de replicación y acceso al Directorio Activo de forma que se utilicen los vínculos y programas más efectivos para el tráfico de inicio de sesión y replicación. Además permite que clientes de una subred puedan localizar servicios en subredes más próximas o con mejor ancho de banda [11].

- **Roles de Maestro de Operaciones (FSMO)**

A diferencia de un dominio NT4[12] en el que existe solo un maestro responsable de procesar los cambios, conocido como controlador de dominio primario (PDC, del inglés *Primary Domain Controller*) y este replica los cambios al resto de los controladores secundarios (BDC, del inglés *Backup Domain Controller*), el Directorio Activo opera como un sistema multi-maestro. La principal ventaja es que todos los controladores de dominio pueden escribir los cambios realizados. Este enfoque puede traer varios conflictos respecto al manejo de los cambios en el directorio. Una posible solución en caso de que se hayan realizado modificaciones distintas sobre un mismo objeto en varios controladores, es mantener la última que se haya realizado. Sin embargo, existen situaciones en que esta medida no es suficiente para la resolución de los conflictos durante el manejo de los cambios. Por ello, el Directorio Activo asigna roles a los

11 TCP/IP: Protocolo de Control de Transmisión/Protocolo de Internet, del inglés *Transmission Control Protocol/Internet Protocol*.

12 Dominio NT4: Implementación de un dominio desde un servidor que ejecuta el sistema operativo Windows NT 4.0 Server.

controladores de dominio para algunas tareas específicas, de manera que opera parcialmente como PDC, siendo exclusivamente un controlador de dominio el responsable de procesar cambios específicos. Los roles que implementa el Directorio Activo son [11]:

- Maestro de esquema.
- Maestro de nombres de dominio.
- Emulador de PDC.
- Maestro de identificadores relativos.
- Maestro de infraestructura.

Los dos primeros roles son únicos a nivel de bosque y el resto a nivel de dominio. A continuación se describen cada uno de estos roles.

Maestro de esquema. El maestro de esquema controla las actualizaciones y modificaciones del esquema del directorio. Una vez que las actualizaciones son completadas, los cambios son replicados a todos los controladores en el bosque. Debe existir uno por bosque [11].

Maestro de nombres de dominio. Existe solo un maestro de nombres de dominio en el bosque. Este es el único que puede adicionar o eliminar un dominio en el directorio. La información de nombres de dominios es almacenada en la partición *Configuration Naming Context* en CN=Partitions, CN=Configuration, DC=domain. Esta partición existe en todos los controladores, pero solo es actualizada en el controlador que posee este rol [11].

Emulador de PDC. El servidor propietario del rol emulador de PDC controla las actualizaciones y modificaciones del esquema del directorio. Una vez que las actualizaciones son completadas, los cambios son replicados a todos los controladores. Debe haber uno por cada dominio. Además, se encarga de ofrecer servicios de sincronización de tiempo entre los controladores de dominio y los clientes, así como procesar los bloqueos de cuentas. Los cambios de contraseñas son replicados preferentemente a este controlador. De ser posible, debe estar disponible todo el tiempo dado que Kerberos exige una correcta sincronización en todas las computadoras para poder ejecutar los inicios de sesión [11].

Maestro de identificadores relativos. El controlador de dominio que posee este rol es responsable de responder a las solicitudes de identificadores relativos (RID) de todos los controladores en un dominio [11]. También es responsable de quitar un objeto de su dominio y ponerlo en otro dominio durante el

traslado de un objeto. Cuando un controlador de dominio crea un objeto principal de seguridad como un usuario o grupo, adjunta al objeto un identificador de seguridad único (SID). Dicho SID está compuesto por el SID del dominio y un RID que es único para cada objeto creado en el dominio.

Maestro de infraestructura. Existe solo un maestro de infraestructura por cada dominio. Se encarga de actualizar las referencias a objetos comparando sus datos de directorio con el catálogo global, replicando los cambios si es necesario. Su función consiste en actualizar los atributos Identificador de Seguridad (SID, del inglés *Security Identifier*) y DN de un objeto cuando se hace una referencia entre objetos de diferentes dominios. Es usado, por ejemplo, si un usuario de un dominio es añadido a un grupo de seguridad de otro dominio [11].

- **Niveles funcionales**

Los niveles funcionales determinan las funciones disponibles en el dominio y el bosque del Directorio Activo, a medida que se elevan sus valores se habilitan nuevas capacidades. Además, establecen los sistemas operativos Windows Server que pueden ejecutarse en los controladores de dominio. El nivel funcional del bosque limita al del dominio, puesto que ningún dominio puede poseer un nivel funcional inferior al del bosque [7].

Los niveles funcionales de dominio son Windows Server 2000 Mixto, Windows Server 2000 Nativo, Windows Server 2003 provisional, Windows Server 2003, Windows Server 2008, Windows Server 2008 R2, Windows Server 2012 y Windows Server 2012 R2. El nivel Windows Server 2003 provisional permite servidores NT 4.0 y Windows Server 2003, pero no Windows Server 2000. Este nivel se establece cuando se migra un PDC NT 4.0 a Windows Server 2003 [7].

Los controladores de dominio no pueden ejecutar un sistema operativo inferior al establecido por el nivel funcional del dominio. Por ejemplo, un dominio con nivel funcional Windows Server 2008 no permite controladores de dominio con el sistema operativo Windows Server 2000, ni Windows Server 2003 [7].

Además, existen dos modos de niveles funcionales: mixto y nativo. El modo mixto permite controladores de dominio que ejecutan el sistema operativo Windows Server NT 4.0 y solo está disponible para Windows Server 2000 y Windows Server 2003. Este modo no permite hacer uso de funciones como el empleo de grupos universales. Por el contrario, el modo nativo requiere que todos los dominios usen Directorio Activo, o sea, que los controladores de dominio ejecuten un sistema operativo igual o superior a Windows Server 2000 [7].

- **Grupos**

Un grupo es un conjunto de cuentas de usuario y de equipo, y de otros grupos que pueden ser administrados como una sola unidad [3]. Los grupos se distinguen por un ámbito y un tipo.

Existen dos tipos de grupos: los de seguridad y los de distribución. Los grupos de seguridad permiten asignar permisos para el acceso a los recursos compartidos en el dominio. Su función es controlar quién puede usar qué recursos. Los grupos de distribución se usan para listas de distribución de correo electrónico. A estos grupos no se les puede asignar permisos para el acceso a los recursos.

El ámbito de un grupo determina y comprende si el grupo pertenece a uno o varios dominios. Los ámbitos definidos en el Directorio Activo así como sus reglas de pertenencia se describen a continuación.

El grupo local de dominio se usa para garantizar permisos a recursos situados en el mismo dominio. Solo es visible en el dominio en que se crea [3].

- Miembros. Puede contener usuarios, grupos globales y otros grupos universales de cualquier dominio del bosque. En un dominio de modo nativo[13], también puede contener grupos locales de dominio de su propio dominio.
- Miembro de. En modo nativo, puede ser miembro de otro grupo local desde su propio dominio.
- Permisos. Cualquier recurso del dominio en donde reside el grupo local de dominio.

El grupo global organiza objetos de usuario de dominio entre dominios en función de las labores que realizan. Es visible en todos los dominios del bosque [3].

- Miembros. Puede contener cuentas de usuario del dominio en donde existe el grupo. En un dominio de modo nativo, los grupos globales también pueden contener otros grupos globales desde el dominio en donde existe.
- Miembro de. Grupos locales de dominio de cualquier dominio del bosque. En un dominio de modo nativo, también puede ser miembro de otro global en su propio dominio.
- Permisos. Todos los dominios del bosque.

El grupo universal se utiliza para agrupar usuarios y conceder permisos en todo el bosque. Es visible en todo el bosque [3].

13 Modo nativo: Modo que solo permite en el dominio servidores ejecutando versiones igual o superior a Windows Server 2000. No soporta controladores de dominio NT.

- Miembros. Puede contener cuentas de usuario, grupos globales y otros grupos universales de cualquier dominio del bosque.
- Miembro de. Grupos locales de dominio y otros grupos universales de cualquier dominio del bosque.
- Permisos. Todos los dominios del bosque.

El empleo de grupos dentro del Directorio Activo permite simplificar la administración, al poder asignar los permisos para los recursos compartidos a un conjunto de objetos en lugar de hacerlo usuario por usuario.

1.2 Samba4

Samba es una herramienta de software libre y código abierto que se distribuye bajo la licencia GNU GPL[14]. El objetivo que persigue este proyecto es eliminar las barreras de interoperabilidad entre servidores Linux/Unix y clientes Windows [12]. Desde el año 1992 provee servicios de archivos e impresión seguros y estables para todos los clientes usando el protocolo SMB/CIFS. Además, ofrece a los administradores de red flexibilidad y libertad en términos de configuración y elección de sistemas y equipamiento de red [13].

El desarrollo de la versión 4 de Samba se realizó a partir de la reescritura de Samba3, en busca de una herramienta modular y el reemplazo de un Directorio Activo. Su desarrollo se ha realizado completamente desde cero respecto a la versión anterior [14].

Andrew Bartlett, miembro del equipo Samba, en una conferencia impartida en la empresa Hobart, Estados Unidos, manifestó que Samba4 está destinado a reemplazar el Directorio Activo proporcionando una implementación del software libre utilizando los protocolos de Microsoft, Samba4 no es sólo el desarrollo de la personalización de los protocolos de Microsoft, es también la posibilidad de migrar el proyecto proporcionando un dominio compatible con el dominio del NT4 [15].

Samba4 es un proyecto desarrollado sobre GNU/Linux en conjunto con el equipo de la empresa Microsoft, por lo cual, una de sus características principales es la casi total compatibilidad con las implementaciones de Directorio Activo existentes. Samba4 utiliza las tecnologías y protocolos que soporta el Directorio Activo, entre las que se pueden mencionar: SMB, LDAP, DNS, Kerberos y NTP.

14 GPL: Licencia Pública General de GNU o más conocida por su nombre en inglés GNU *General Public License.*

1.3 Migración del servicio Directorio Activo hacia Samba4

El proceso de migrar los datos desde un Directorio Activo en Windows Server hacia Samba4, garantizando su integridad de los mismos, presenta gran relevancia para la migración a código abierto en las instituciones cubanas. Para realizar dicho proceso Samba4 provee su propio mecanismo de migración. Este proceso de migración es equivalente a unir el controlador de dominio Samba4 a un Directorio Activo existente, lo cual garantiza que ambos servidores continúen replicando todos los datos en la red durante el periodo en que estén conectados [16].

A pesar de la alta compatibilidad de Samba4 con el Directorio Activo, existen algunas prestaciones que Samba4 aún no ofrece, puesto que no han sido implementadas por el equipo de desarrollo. Ejemplo de ello es la migración de las políticas de grupo en Windows Server.

- **Preparación del entorno en el servidor GNU/Linux**

Antes de proceder a ejecutar el mecanismo de migración se deben realizar varias acciones en el sistema operativo. Se debe tener en cuenta el nombre del servidor y su vinculación con el dominio desde el cual se realizará la migración de los datos. Además, en el sistema operativo base existen aplicaciones de seguridad que deben ser configuradas para evitar que se bloqueen determinadas acciones durante el despliegue del servicio, tal es el caso de *apparmor*. Se debe comprobar que la dirección IP del servidor sea estática en lugar de haber sido asignada por el protocolo DHCP. También se debe eliminar cualquier versión previa de Samba que pueda existir en el ordenador. Es importante no obviar las opciones con que deben montarse las particiones que utilizarán la base de datos del servicio para tener acceso a las características avanzadas de Samba4, las cuales se especifican a continuación:

- Sistema de archivos ext3: user_xattr, acl, barrier=1
- Sistema de archivos ext4: defaults, barrier=1
- XFS: No las necesita

El empleo de un controlador de dominio con Samba4 en una red requiere instalar en el servidor un conjunto de dependencias para la ejecución de las tecnologías utilizadas, así como DNS, Kerberos, NTP y el servicio principal, Samba4, que internamente utiliza un servicio LDAP. El orden para la realización de estas tareas debe ser el siguiente:

1. Instalación de dependencias.
2. Adición de las tecnologías necesarias.

3. Adición del servicio principal (Samba4).

Según la documentación oficial que ofrece el equipo de Samba, existen algunos requisitos que deben cumplirse para realizar la migración.

- El nivel funcional del bosque debe ser al menos 2003 nativo.
- La versión del esquema del bosque debe ser como máximo 47. En los servidores Windows se puede comprobar la versión accediendo al parámetro *Schema Version* en la llave de registro: *HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\NTDS\Parameter*; o ejecutando el comando: *dsquery * cn=schema,cn=configuration,dc=domainname,dc=local -scope base -attr objectVersion.*

➢ **Migración de los datos**

El mecanismo de migración que ofrece Samba4 mediante la utilidad *domain join*[15] garantiza la migración de la totalidad de objetos contenidos en el Directorio Activo como son las cuentas de usuarios y equipos, los grupos y las unidades organizativas. Además se mantiene el Árbol de Información de Directorio (DIT) con su estructura original.

Para emplear este mecanismo se deben especificar el dominio y las credenciales de una cuenta con permisos de administración sobre el dominio. Además se puede precisar el reino de Kerberos, el sistema de nombres de dominio a utilizar, la interfaz de red a la cual se va a asociar el servicio en caso de que el controlador de dominio tenga más de una tarjeta de red. A continuación se muestra un ejemplo de la utilidad *domain join* de la herramienta samba-tool de Samba4: *samba-tool domain join dominio.pdc.cu DC -U Administrator --realm=dominio.pdc.cu –dns-backend=BIND9_DLZ*.

➢ **Migración de las políticas de grupo**

Samba4 aún no implementa la replicación de las políticas de grupo. Los desarrolladores de Samba4 recomiendan una solución que permite automatizar este proceso. Consiste en programar la ejecución de una tarea del sistema que copie todo el contenido de la carpeta compartida sysvol[16] desde el servidor Windows hacia el controlador de dominio Samba4. Para ello, se propone el empleo del software rsync[17] debido a que soporta la lectura de las ACL extendidas presentes en los sistemas Windows.

15 Join: Utilidad que ofrece la herramienta de administración samba-tool para la unión a un Controlador de Dominio.

16 sysvol: Volumen del sistema, del inglés *system volume*. Recurso compartido en el Controlador de Dominio donde se almacenan los contenedores y las configuraciones de las políticas de grupo. A esta carpeta se accede a través de la dirección: smb://ip_del_servidor/sysvol

17 rsync: Es una aplicación libre para sistemas de tipo Unix y Microsoft Windows que ofrece transmisión eficiente de datos incrementales.

- **Acciones posteriores a la migración del Directorio Activo**

Se deben tener en cuenta varias consideraciones relativas a los roles del maestro de operaciones y el correcto funcionamiento de las diversas tecnologías empleadas tales como Kerberos y DNS.

Si la migración de los datos culminó de forma exitosa, el servidor Windows puede ser degradado para que deje de brindar el servicio de directorio. Previamente todos los roles que posee el servidor Windows deben ser transferidos hacia el controlador de dominio Samba4.

Una de las tecnologías cuyo correcto funcionamiento es esencial para el servicio Directorio Activo es DNS, por lo que se deben comprobar la existencia de zonas y sus respectivos registros.

1.4 Administración del servicio Directorio Activo

En la presente sección se analizan diferentes aplicaciones tanto privativas como libres que permiten la administración de un servicio de directorio LDAP de forma parcial o total.

1.4.1 Herramientas de Administración Remota del Servidor (RSAT)

Las herramientas RSAT[18] permiten la administración remota de los roles y las características de los servidores de Windows desde una computadora con cualquier versión de Windows. Dentro de estas herramientas destacan aquellas que se vinculan directamente con la administración del servicio Directorio Activo. Del conjunto de herramientas que ofrece RSAT, las que tributan al objetivo de la investigación, debido a que se especializan en la administración avanzada del servicio son:

- Usuarios y computadoras del Directorio Activo (ADUC, del inglés *Active Directory Users and Computers*): es la herramienta principal para la administración avanzada de objetos del Directorio Activo. Establece una conexión con un dominio o un controlador de dominio y permite realizar la gestión avanzada de usuarios, grupos, computadoras, dominios, unidades organizativas y Contenedores de Directivas de Grupo (GPC, del inglés *Group Policy Containter*) desde una interfaz sencilla e intuitiva.
- Administrador de DNS (*DNS Manager*): es la aplicación principal para la administración de DNS en los servidores de Windows que permite la gestión completa de los tipos de zonas y registros.

18 RSAT: del inglés *Remote Server Administration Tools.*

- Editor de políticas de grupo (GPOE del inglés *Group Policy Object Editor*): es una herramienta que se utiliza para la configuración avanzada de los objetos de directivas de grupo. Dentro de las configuraciones destacan plantillas administrativas, *scripts*[19], opciones de seguridad, instalación de software, redirección de carpetas, instalación de servicios remotos y mantenimiento de Internet Explorer.

1.4.2 Clientes LDAP

Los clientes LDAP son herramientas que se utilizan para la administración del servicio LDAP. Estas aplicaciones se encargan de establecer una conexión con un controlador de domino, el cual puede tener implementado un Directorio Activo o alguna otra tecnología que implemente un servicio de directorio como OpenLdap.

- **PhpLDAPadmin (PLA)**

Es un cliente LDAP multiplataforma basado en la web y está diseñado para ser usado por administradores de LDAP con el propósito de gestionar la base de datos LDAP. Permite la navegación por el árbol LDAP y la administración de entradas incluyendo la búsqueda, eliminación, creación y modificación de estas. Los usuarios finales del sistema deben tener conocimientos básicos sobre el servicio en cuestión. También permite exportar información a archivos Formato de intercambio de Datos LDAP (LDIF[20]) y Lenguaje de Marcado de Servicios de Directorio (DSML[21]). El usuario puede acceder al esquema del servicio y sus configuraciones, y realizar lecturas sobre atributos binarios [17].

- **JXplorer (JX)**

Es una herramienta multiplataforma de propósito general. Ha sido escrita en Java y se puede utilizar para buscar, leer y editar cualquier directorio estándar, o cualquier servicio de directorio LDAP o con una interfaz DSML. Permite realizar búsquedas complejas sobre el directorio a través de una interfaz que posibilita la construcción de filtros de búsquedas. JX proporciona la navegación por el directorio LDAP al igual que copiar, mover o eliminar de forma recursiva una porción de este [18].

- **LDAP Administration Tool (LAT)**

19 Script: archivo de órdenes, archivo de procesamiento por lotes o, también conocido en círculos profesionales y académicos como guión.

20 LDIF: del inglés *Data Interchange Format*.

21 DSML: del inglés *Directory Services Markup Language.*

LAT es un software de administración de LDAP para sistemas GNU/LINUX. Permite navegar a través de los directorios y añadir, eliminar y mover entradas. Es capaz de almacenar perfiles para realizar conexiones rápidas a diferentes servidores. Proporciona soporte para exportar e importar archivos LDIF y se integra con el entorno de escritorio GNOME [19].

- **LDAP Admin**

Es un software libre y de código abierto para el sistema operativo Windows que se utiliza para la administración del servicio LDAP. La aplicación soporta atributos binarios y exportar e importar archivos LDIF. Permite la gestión de los grupos y cuentas POSIX[22]. Incluye también la gestión de cuentas de Samba y la navegación por el esquema LDAP [20].

1.4.3 Herramienta samba-tool

Samba-tool es la herramienta principal de administración de Samba4 pero la interacción con el usuario final es a través de una interfaz de línea de comandos. Permite administrar solamente los objetos usuarios y grupos. Permite crear, eliminar y actualizar zonas de DNS, también adicionar y eliminar registros de zonas. En el caso de la administración del dominio facilita la gestión de las políticas de las contraseñas y la configuración del nivel funcional del bosque y del dominio. En cuanto a la gestión de los GPO[23] solamente se permite crear y eliminar este tipo de objetos [21].

Se decide basar el diseño del módulo en las herramientas privativas ADUC, Administrador de DNS y Editor de GPO debido a que estas aplicaciones se especializan específicamente en la administración del servicio Directorio Activo. Son herramientas con interfaces sencillas que, a pesar de no liberar el código fuente, aportan a la definición de las funcionalidades y el diseño del sistema a implementar. Al contrario de estas aplicaciones, los clientes LDAP analizados no se especializan en el servicio Directorio Activo, solo comprenden la administración de este protocolo. No obstante, el estudio del código fuente contribuye a la investigación referente a la administración de los atributos mediante LDAP. La herramienta samba-tool, a pesar de ser una aplicación basada en una interfaz de línea de comandos, ofrece funcionalidades esenciales para la administración que mediante LDAP no son posibles realizar. Sin embargo, dicha herramienta que provee Samba4 no permite realizar una gestión avanzada de los datos almacenados en el Directorio Activo.

22 POSIX: Interfaz de Sistema Operativo Portable, del inglés *Portable Operating System Interface.*

23 GPO: Objeto de Directivas de Grupo, del inglés *Group Policy Object.*

1.5 Herramienta de Migración y Administración de Servicios Telemáticos

La herramienta HMAST es una aplicación web que permite la administración de máquinas servidoras. La administración se puede realizar tanto de forma local como remota, esta última a través de conexiones seguras mediante el uso del protocolo SSH. Tiene las funcionalidades necesarias para la gestión de usuarios, tareas programadas y servicios telemáticos [22].

- **Arquitectura de HMAST**

La arquitectura de HMAST presenta un estilo N-Capas orientada al Dominio (ver Figura 2). Su estructura se basa en cinco capas las cuales interactúan a través de interfaces y utilizando inyección de dependencias. Las características y principales responsabilidades de cada una de estas capas se describen a continuación.

- **Capa Presentación** (*presentation*). Presenta al usuario los conceptos del negocio mediante una interfaz de usuario. Además, facilita la explotación de dichos procesos, informa sobre la situación de los procesos de negocio e implementación de las reglas de validación de dicha interfaz. En esta capa se realiza el tratamiento a las excepciones lanzadas desde capas inferiores.
- **Capa Aplicación** (*application*). Realiza las llamadas a servicios de la capa inferior y tiene la responsabilidad de adaptar la información que recibe a los requisitos de los servicios de dominio. En esta capa también se ubican operaciones de trazas, seguridad, envío de correos electrónicos, cuando no forman parte estricta del negocio. Está compuesta por los servicios de aplicación y los objetos para la transferencia de datos (DTO, del inglés *Data Object Transfer*).
- **Capa Dominio** (*domain*). Se encarga de implementar la lógica de dominio, o sea, las reglas del negocio. Define las interfaces de persistencia a datos, conocidos como contratos a repositorios, pero no los implementa. Es responsable de realizar las validaciones y sus componentes solo dependen de la capa Infraestructura Transversal. Esta capa está compuesta por las entidades del dominio, los servicios de dominio y los contratos de repositorio. Las entidades representan objetos del dominio y están definidas fundamentalmente por su identidad y continuidad en el tiempo. Los servicios de dominio contienen la lógica que trata a las entidades como un todo y los contratos de repositorios son interfaces que especifican las operaciones que deben implementar los repositorios.

- **Capa Persistencia** (*persistence*). Es responsable de contener el código necesario para persistir los datos. Se compone fundamentalmente por los repositorios, que son clases que implementan los contratos de repositorios definidos en la **capa de dominio**.
- **Capa Infraestructura Transversal** (*infrastructure crosscutting*). Es responsable de promover la reutilización de código. Contiene las operaciones de seguridad, inicios de sesión, monitoreo del sistema, mecanismos de persistencia reutilizables, validadores genéricos y todas aquellas operaciones que se puedan utilizar desde otras capas.

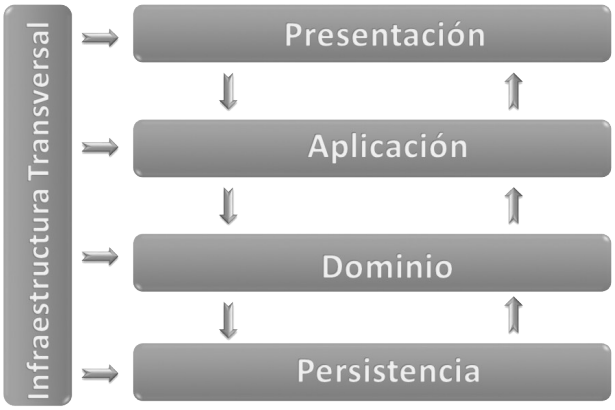


Figura 1. Arquitectura de HMAST.

- **Consideraciones para implementar un módulo para HMAST**

Para integrar un módulo a HMAST se deben tener en cuenta un conjunto de consideraciones las cuales se especifican a continuación.

- La lógica de Aplicación no deberá incluir ninguna lógica del Dominio, solo tareas de coordinación relativas a requerimientos técnicos de la aplicación, como conversiones de formatos de datos de entrada a entidades del Dominio y llamadas a componentes de Infraestructura para que realicen tareas complementarias.
- La capa de Presentación no debe tener como entrada o salida, objetos de dominio, sino objetos para la transferencia de datos (DTO).
- Las entidades solo pueden tener dependencias de componentes de la capa de dominio.
- Las clases de servicios deben ser las únicas responsables de acceder a los repositorios, no se puede implementar código de persistencia a datos en la capa de Dominio.

- Solo se puede acceder a la información almacenada en los servidores haciendo uso de los repositorios.
- El código reutilizable por más de un repositorio debe estar disponible en la capa de Infraestructura Transversal.

1.6 Metodología, herramientas y tecnologías

Teniendo en cuenta que el módulo que se va a desarrollar debe ser integrado a una herramienta (HMAST), la metodología, las herramientas y las tecnologías empleadas se corresponden con las mismas de la herramienta principal.

➢ Metodología de desarrollo de software

La metodología para el desarrollo de software es un modo sistemático de realizar, gestionar y administrar un proyecto para llevarlo a cabo con altas posibilidades de éxito. Una metodología para el desarrollo de software comprende los procesos a seguir sistemáticamente para idear, implementar y mantener un producto de software desde que surge la necesidad del producto hasta que se cumple el objetivo por el cual fue creado. Estas se clasifican en dos tipos: ágiles y tradicionales [23].

Para el desarrollo del módulo de administración y migración de Directorio Activo hacia plataforma libre se emplea la metodología UCI, resultante de una variación de la metodología ágil AUP[24].

AUP-UCI. Esta metodología define 3 fases, 7 disciplinas y 11 roles. Las fases de AUP-UCI son: inicio, ejecución y cierre. Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación. En la segunda fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. En la fase de cierre se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Los roles definidos por esta metodología son: jefe de proyecto, planificador, analista, arquitecto de información (opcional), desarrollador, administrador de la configuración, cliente/proveedor de requisitos, administrador de calidad, probador, arquitecto de sistema y administrador de base de datos. Además AUP-UCI define cuatro escenarios para modelar el sistema. Se emplea el escenario número 4, el cual indica que proyectos que no modelen negocio solo pueden modelar el sistema con HU [24].

24 AUP: Proceso Unificado Ágil, del inglés *Agile Unified Process*.

- **Lenguaje de programación**

Los lenguajes de programación son un conjunto de reglas, herramientas y condiciones que permiten crear programas o aplicaciones dentro de un ordenador. Permiten además ordenar distintas acciones a la computadora en un lenguaje entendible por esta. Cada lenguaje posee su parte sintáctica y semántica, que no son más que el conjunto de reglas acerca de cómo se deben escribir las instrucciones y de qué forma [25].

Java. Lenguaje de programación creado por Sun Microsystems basado en C y C++ aunque es un lenguaje mucho más sencillo y simple. El concepto de objeto es parte integral de Java convirtiéndolo en un lenguaje orientado a objetos. Estos combinan los valores con las funciones que operan sobre ellos. Posee una arquitectura neutra y funciona con uniformidad en diferentes sistemas operativos. La precisión y los formatos para los tipos de datos primitivos se especifican por completo, para asegurar que los programas de Java funcionen igual en todas las implantaciones [26].

- **Framework**

Un *framework* (marco de trabajo) es un conjunto de bibliotecas, módulos o artefactos reutilizables que establecen una estructura que brinda soporte a los desarrolladores. Provee las piezas necesarias, las cuales deben ser personalizadas y conectadas para desarrollar la aplicación. Representa un conjunto de consideraciones especiales que garantizan la calidad del diseño del mismo [27].

Spring. Es un marco de trabajo de código abierto para Java que proporciona muchas funcionalidades útiles como la inversión de control, la inyección de dependencias y el soporte para la programación orientada a aspectos. Permite el desarrollo de todo tipo de aplicaciones de manera rápida aplicando conceptos de modularización y facilita la realización de pruebas al código. Abstrae al desarrollador de los conceptos de “solicitudes” y “respuestas” y garantiza la reutilización de código dentro del software.

La Inversión de Control (IoC del inglés *Inversion of Control*) y la Inyección de Dependencias (DI del inglés *Dependency Injection*) se encuentran entre las principales características de Spring y están estrechamente relacionadas entre ellas. IoC es un patrón de diseño de software en el cual un ensamblador, en este caso Spring Framework, realiza el acoplamiento entre objetos en tiempo de ejecución en vez de en tiempo de compilación. Permite que los desarrolladores programen un conjunto de interfaces, las cuales puedan ser intercambiadas en diferentes ambientes y sin que sea necesario volver a compilar el código. La DI es la técnica más común para realizar este proceso. Usando DI, una porción del

programa, es decir una clase, se declara que depende de una interfaz y en tiempo de ejecución, Spring Framework es el encargado de realizar la inyección de la instancia de esa dependencia [28].

- **Biblioteca LDAP. Spring LDAP**

Las bibliotecas para la conexión LDAP permiten acceder a directorios LDAP, en este caso a través del lenguaje Java. Spring LDAP es un proyecto de Spring Source que simplifica considerablemente la programación utilizando este servicio en Java. Se basa en los principios de plantillas de Base de Datos en Java (JDBC[25]) de Spring. Reduce la realización de tareas rutinarias, el desarrollador no necesita preocuparse por establecer o cerrar la conexión en cada operación que se realice. Elimina las iteraciones sobre los resultados de *NamingEnumeration*. Proporciona facilidades para la creación de consultas, generación dinámica de filtros y nombres de dominio, además de ofrecer ventajas al programador para la gestión de los atributos LDAP. Permite la configuración de una agrupación de conexiones con el objetivo de reutilizar las conexiones físicas que se encuentren abiertas. Provee al programador una jerarquía de excepciones basada en el *DataAccessException* de Spring [29].

- **Entorno de Desarrollo Integrado**

Un entorno de desarrollo integrado (IDE[26]) es un programa que permite a los desarrolladores escribir y ejecutar código a través de un lenguaje de programación y un conjunto de herramientas. Estos pueden soportar más de un lenguaje de programación o uno específicamente.

IntelliJ IDEA. Es un IDE para el lenguaje de programación Java. Es ligero en el diseño y ofrece características útiles como pruebas de Junit, depuración, inspección de código y soporte para múltiple refactorización. IntelliJ Idea se encuentra enfocado en elevar la productividad de los desarrolladores por lo que ofrece soporte avanzado para los frameworks y estándares más importantes en el desarrollo web como son Spring Framework, Vaadin, Play, Grails, Web Services y Struts. Además incluye la asistencia de código para lenguajes como HTML, CSS, JavaScript, Node.js y TypeScript. Favorece la integración que tiene el IDE con herramientas para la construcción de proyectos como Maven, Ant y Gradle; y para el control de versiones como Git, SVN y Mercurial. Incorpora también un editor de base de datos SQL con soporte completo SQL para Oracle, PostgreSQL, MySQL y SQL Server.

- **Herramienta de Ingeniería de Software Asistida por Computadora**

25 JDBC: Conectividad de Base de Datos Java, del inglés *Java Database Connectivity.*

26 IDE: Entorno de Desarrollo Integrado, del inglés *Integrated Development Environment*.

Una herramienta CASE[27] ofrece facilidades al ingeniero de software durante el proceso de desarrollo y mantenimiento del sistema. Es además una herramienta individual para ayudar al desarrollador de software o administrador de proyecto durante una o más fases del desarrollo de software (o mantenimiento) [30].

Visual Paradigm para UML. Es una herramienta CASE diseñada para una amplia gama de usuarios, incluyendo ingenieros de software, analistas de sistemas, analistas de negocios y arquitectos de sistemas, o para cualquier persona que esté interesada en la construcción de forma fiable de los sistemas de software a gran escala con un enfoque orientado a objetos. Ofrece soporte para varias versiones de UML y para operaciones de ingeniería inversa de código a modelo o diagrama y de modelo o diagrama a código. Permite el trabajo colaborativo, es decir, varias personas pueden trabajar sobre un mismo proyecto.

- **Lenguaje Unificado de Modelado**

UML[28] es un lenguaje estándar utilizado para escribir planos de software. Puede utilizarse para visualizar, especificar, construir y documentar los artefactos y las relaciones entre ellos, que se crean durante el desarrollo del software. Este lenguaje se emplea para el diseño del módulo, específicamente para la construcción de los diagramas de clases por paquetes y el diagrama de despliegue.

- **Maven**

Maven es una herramienta de software de código abierto para la gestión y construcción de proyectos basada en estándares. Permite gestionar el ciclo de vida de un proyecto desde su creación hasta la generación de un binario que pueda distribuirse con este. Ofrece facilidades a los desarrolladores como la sencilla y ágil creación de proyectos o módulos, además de aplicar una estandarización de la estructura y organización del proyecto. También dispone de un robusto mecanismo de gestión de dependencias de un proyecto sobre las bibliotecas propias o de terceros y mantiene disponible para los desarrolladores un repositorio de bibliotecas de código abierto en constante actualización.

- **Sistema de control de versiones**

Un sistema de control de versiones es una herramienta que hará un seguimiento de los cambios en los archivos y permitirá la coordinación de diferentes desarrolladores que trabajan en partes de su sistema al

27 CASE: Herramienta de Ingeniería de Software Asistida por Computadora, del inglés *Computer Aided Software Engineering.*

28 UML: Lenguaje Unificado de Modelado, del inglés *Unified Modeling Language.*

mismo tiempo [31]. En el proyecto en el cual se realiza el módulo se utiliza Subversion, un sistema de control de versiones libre y de código fuente abierto que permite gestionar los cambios realizados en el proyecto.

Conclusiones parciales

La sistematización en los conceptos, características y funciones del servicio Directorio Activo permitió una mejor comprensión de su estructura y funcionamiento. Se caracterizó la aplicación de código abierto Samba4, lo que permitió identificar las funcionalidades que ofrece como alternativa libre al Directorio Activo. Se estableció como mecanismo de migración a emplear, la utilidad *domain join* que brinda la herramienta samba-tool lo cual permite ejecutar este proceso garantizando la integridad de los datos. Se estudiaron diversas aplicaciones que permiten la administración del protocolo LDAP, concluyendo que se tendrán en cuenta las herramientas RSAT para el diseño del módulo. Se caracterizó la herramienta HMAST para identificar los aspectos a tener en cuenta para integrar el módulo a esta herramienta. La selección de la arquitectura, la metodología, las herramientas y tecnologías a emplear en el diseño e implementación del módulo, demostró las potencialidades de todas estas tecnologías, en aras de lograr la creación del módulo propuesto en la presente investigación.

CAPÍTULO 2. DISEÑO DEL MÓDULO

Tomando como punto de partida las características del sistema base HMAST y los elementos planteados por la metodología AUP-UCI, se describe la propuesta de diseño del módulo a desarrollar. Para ello, se especifican las funcionalidades del sistema mediante los requisitos funcionales y las historias de usuario. Además, se describe la arquitectura y los patrones de diseño que se emplean.

2.1 Propuesta de solución

En la presente investigación se propone el desarrollo de una aplicación con las funcionalidades que permitan, desde HMAST, administrar el Directorio Activo y migrar los datos de este servicio hacia Samba4. El módulo permitirá la administración del servicio en diversas máquinas servidoras de forma remota, mediante conexiones seguras empleando el protocolo SSH.

En la sección 1.3 se explicó el proceso de migración del Directorio Activo hacia Samba4 y se ofrecen los elementos técnicos de dicho proceso. El módulo permite, a través de una interfaz gráfica de usuario, migrar los datos y servicios hacia Samba4. Para ello, se debe especificar el nombre del dominio DNS, el tipo de servidor DNS, las credenciales del usuario administrador del dominio y en caso de existir más de una interfaz de red, cuál se va a usar para el controlador de dominio. Las zonas y registros DNS pueden ser gestionados con el DNS interno que ofrece Samba4 o con el servidor Bind. El DNS interno permite una configuración rápida y sencilla. Bind se recomienda para configuraciones DNS complejas o funciones especiales como la transferencia de zonas que actualmente no son soportadas por el DNS interno de Samba4. El servidor Bind debe ser ejecutado en el mismo servidor Samba4 y para la gestión DNS se emplea el controlador BIND_DLZ[29]. En cualquier caso, el servidor DNS seleccionado puede ser cambiado en correspondencia con los requerimientos del entorno en que se trabaja.

El módulo proporciona las funcionalidades de administración del servicio Directorio Activo. Permite la gestión de las principales clases de objetos de este servicio, tales como cuentas de usuario y de equipos, grupos y unidades organizativas. Brinda la posibilidad de gestionar los recursos DNS, específicamente zonas y registros. Permite además, mostrar y elevar los niveles funcionales del bosque y del dominio.

29 DLZ: del inglés *Dynamically Loadable Zones*.

2.2 Artefactos generados

El proceso de desarrollo es guiado por la metodología AUP-UCI. Teniendo en cuenta que no se modela el negocio y se ajusta por tanto, al escenario 4 que establece esta metodología, las funcionalidades se describen en un documento de Especificación de Requisitos de Software. Estos requisitos son encapsulados mediante Historias de Usuario, lo cual constituye el principal artefacto generado durante el diseño del módulo.

2.2.1 Especificación de requisitos

El Glosario de Terminología Estándar de Ingeniería de Software[30] define al requisito como una condición que necesita un usuario para resolver un problema o lograr un objetivo. También como una capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente [32]. Estos se clasifican en funcionales y no funcionales.

Los requisitos funcionales describen las funciones que el software debe ejecutar, con el objetivo de establecer un entendimiento común entre el usuario y el proyecto de software. La siguiente tabla muestra el listado de los requisitos funcionales del módulo clasificados según su prioridad en alta, media y baja.

Tabla 1. Listado de requisitos funcionales.

No.	Nombre del requisito funcional	Descripción
Prioridad		**Alta**
RF1	Instalar el servicio Directorio Activo	Permite instalar el servicio Directorio Activo a partir de la herramienta Samba4.
RF2	Desinstalar el servicio Directorio Activo	Permite desinstalar el servicio Directorio Activo a partir de la herramienta Samba4.
RF3	Mostrar objetos del Directorio Activo	Permite mostrar los objetos del Directorio Activo en una estructura jerárquica de árbol de directorio.
RF4	Crear usuario	Permite crear un usuario en el directorio.
RF5	Eliminar usuario	Permite eliminar un usuario del directorio.
RF6	Habilitar usuario	Permite habilitar un usuario, de modo que este pueda

30 IEEE: *Standard Glossary of Software Engineering Terminology*.

		autenticarse en el dominio.
RF7	Deshabilitar usuario	Permite deshabilitar un usuario, de modo que este no pueda autenticarse en el dominio.
RF8	Mostrar atributos de un usuario	Permite mostrar los atributos de un usuario dentro del directorio.
RF9	Mostrar grupo(s) a los que pertenece un usuario	Permite mostrar una lista con todos los grupos a los que pertenece un usuario.
RF10	Adicionar un usuario a grupo(s)	Permite adicionar un usuario a un grupo.
RF11	Establecer en cuáles computadoras puede autenticarse un usuario	Permite establecer en cuáles computadoras del dominio, el usuario puede autenticarse.
RF12	Editar atributos de usuario	Permite editar los atributos de un usuario.
RF13	Crear grupo	Permite crear un grupo en el directorio.
RF14	Eliminar grupo	Permite eliminar un grupo.
RF15	Adicionar miembro(s) a un grupo	Permite adicionar cuentas de usuario y de equipo, y grupos como miembros de un grupo.
RF16	Eliminar miembro(s) de un grupo	Permite eliminar cuentas de usuario y de equipo, y grupos como miembros de un grupo.
RF17	Mostrar miembro(s) de un grupo	Permite mostrar una lista con los miembros de un grupo.
RF18	Mostrar atributos de un grupo	Permite mostrar los atributos de un grupo dentro del directorio.
RF19	Mostrar administrador del grupo	Permite mostrar el usuario que es administrador de un grupo.
RF20	Establecer/Cambiar administrador del grupo	Permite establecer, o cambiar en caso que ya lo tenga, un administrador a un grupo.
RF21	Mostrar grupo(s) a los que pertenece un grupo	Permite mostrar una lista con los grupos a los que pertenece un grupo.
RF22	Adicionar grupo a grupo(s)	Permite adicionar un grupo como miembro de otro grupo.
RF23	Editar grupo	Permite editar los atributos de un grupo.

RF24	Crear unidad organizativa	Permite crear una unidad organizativa en el directorio.
RF25	Eliminar unidad organizativa	Permite eliminar una unidad organizativa del directorio.
RF26	Mostrar atributos de una unidad organizativa	Permite mostrar los atributos de una unidad organizativa dentro del directorio.
RF27	Editar unidad organizativa	Permite editar los atributos de una unidad organizativa.
RF28	Mostrar atributos de una computadora	Permite mostrar una computadora dentro del directorio.
RF29	Eliminar computadora	Permite eliminar una computadora del directorio.
RF30	Mostrar los atributos de una computadora	Permite mostrar los atributos de una computadora.
RF31	Mostrar grupo(s) a los que pertenece una computadora	Permite mostrar una lista con los grupos a los que pertenece una computadora.
RF32	Adicionar computadora a grupo(s)	Permite adicionar una computadora como miembro de un grupo.
RF33	Establecer administrador de la computadora	Permite establecer a un usuario como administrador de una computadora.
RF34	Mostrar políticas de las contraseñas del dominio	Permite mostrar los valores de los parámetros de las políticas de las contraseñas.
RF35	Editar políticas de las contraseñas del dominio	Permite editar los valores de los parámetros de las políticas de las contraseñas.
Prioridad		**Media**
RF36	Adicionar zona DNS	Permite adicionar una zona DNS al directorio.
RF37	Mostrar zona DNS	Permite mostrar una zona DNS.
RF38	Editar zona DNS	Permite editar una zona DNS.
RF39	Eliminar zona DNS	Permite eliminar una zona DNS del directorio.
RF40	Adicionar registro a zona DNS	Permite adicionar un registro a una zona DNS.
RF41	Mostrar registro de zona DNS	Permite mostrar un registro existente dentro de una zona DNS.
RF42	Editar registro de zona DNS	Permite editar los atributos de un registro de zona DNS.

RF43	Eliminar registro de zona DNS	Permite eliminar un registro de zona DNS.
RF44	Iniciar el servicio Samba4	Permite que el administrador de la herramienta pueda iniciar el servicio Samba4.
RF45	Detener el servicio Samba4	Permite que el administrador de la herramienta pueda detener el servicio Samba4.
RF46	Reiniciar el servicio Samba4	Permite que el administrador de la herramienta pueda reiniciar el servicio Samba4.
RF47	Iniciar el servicio DNS de bind	Permite que el administrador de la herramienta pueda iniciar el servicio DNS de bind9.
RF48	Detener el servicio DNS de bind	Permite que el administrador de la herramienta pueda detener el servicio DNS de bind9.
RF49	Reiniciar el servicio DNS de bind	Permite que el administrador de la herramienta pueda reiniciar el servicio DNS de bind9.
Prioridad		**Baja**
RF50	Mostrar información general del servicio Samba4	Permite mostrar los parámetros generales del servicio, como nombre y dirección IP del servidor, nombre del dominio, cantidad de usuarios.
RF51	Replicar el controlador de dominio	Permite replicar el directorio entre varios controladores de dominio.
RF52	Mostrar atributos de múltiples objetos	Permite mostrar los atributos comunes para diversos objetos del directorio.
RF53	Editar atributos de múltiples objetos	Permite editar los atributos comunes para diversos objetos del directorio.
RF54	Mostrar el nivel funcional del dominio	Permite mostrar el nivel funcional del controlador de dominio.
RF55	Elevar el nivel funcional del dominio	Permite elevar el nivel funcional del controlador de dominio.
RF56	Buscar objeto	Permite buscar un objeto dentro del directorio.
RF57	Realizar salva de la base de datos del controlador de dominio	Permite realizar una copia de seguridad de la base de datos del controlador de dominio.
RF58	Restaurar el controlador de dominio	Permite restaurar un controlador de dominio a partir de una copia de seguridad existente.

RF59	Mostrar roles del maestro de operaciones	Permite mostrar los roles del maestro de operaciones que posee un controlador de dominio.
RF60	Transferir roles del maestro de operaciones	Permite transferir roles del maestro de operaciones hacia un controlador de dominio.
RF61	Mover objeto	Permite mover un objeto de un contenedor hacia otro.

Los requisitos no funcionales se refieren a cualidades que imponen restricciones en el diseño y la implementación [33]. La metodología AUP-UCI propone una taxonomía partiendo de la ISO 25010 donde se asocian estos requisitos a atributos de calidad (ver Anexo 27). Diversos requisitos no funcionales son heredados de la herramienta HMAST, puesto que condicionan el funcionamiento del módulo para lograr una correcta integración con el sistema base. A continuación se muestra el listado de los requisitos no funcionales.

Tabla 2.Listado de requisitos no funcionales.

No	Nombre del requisito no funcional	Atributo de calidad	Descripción
1	Emplear como lenguaje de programación Java.	Funcionalidad	Estas son restricciones heredadas de HMAST, el módulo debe cumplirlas para poder integrarse correctamente.
2	El módulo se ejecutará sobre el sistema operativo GNU/Linux Nova Server.		
3	Disponer en el sistema operativo GNU/Linux de los siguientes paquetes: *augeas, augeas-tools, libjna-java, openjdk-7-jdk*.		
4	Proteger información y los datos, para que personas o sistemas desautorizados no puedan leer o modificar los mismos, y las personas o sistemas autorizados tenga el acceso a ellos.	Seguridad (Acceso restringido)	Todos los datos manejados por el módulo estarán encriptados; tanto los que se transfieren entre la estación cliente y el servidor HMAST como los que se almacenan temporalmente en el servidor.
5	Permitir al usuario aprender su aplicación.	Usabilidad	Internacionalizar la información que se muestra, en los idiomas español e inglés.
6	El módulo debe mantener un nivel de ejecución o desempeño especificado en caso de fallos del software o de infracción de su interfaz especificada.	Confiabilidad (Tolerancia a fallos)	Ante el fallo de una funcionalidad del sistema el resto de las funcionalidades que no dependen de esta, deberán seguir funcionando.

2.2.2 Historias de Usuario

Las Historias de Usuario especifican las tareas que debe realizar el sistema, lo que equivale a los casos de uso en el proceso unificado. Son escritas en lenguaje natural, sin un formato predeterminado, no excediendo su tamaño de unas pocas líneas de texto. Además, guían la construcción de las pruebas de aceptación y son utilizadas para estimar tiempos de desarrollo [34].

A continuación se muestran las siguientes Historias de Usuario:

1. Instalar servicio samba4 en modo Directorio Activo.
2. Desinstalar servicio samba4 en modo Directorio Activo.
3. Adicionar usuario.
4. Adicionar grupo.
5. Editar configuraciones de contraseñas del dominio.

<table>
<tr><td>Número: 1</td><td colspan="2">Nombre del requisito: Instalar servicio samba4 en modo Directorio Activo.</td></tr>
<tr><td colspan="2">Programador: Felipe González Santiago</td><td>Iteración Asignada: 1</td></tr>
<tr><td colspan="2">Prioridad: Alta</td><td>Tiempo Estimado: 20</td></tr>
<tr><td colspan="2">Riesgo en Desarrollo: Alto</td><td>Tiempo Real: 20</td></tr>
<tr><td colspan="3">Descripción: Permite instalar samba4 en modo Directorio Activo como controlador de dominio.
• Instalar servicio
La funcionalidad comienza cuando el usuario selecciona la opción Activar módulo sobre el módulo Samba4. Luego se muestra una interfaz con el logotipo, una pequeña descripción del servicio y los datos principales del servicio como son: nombre del servicio, fichero principal y directorio principal. Luego el usuario selecciona el botón Siguiente y se muestra una interfaz donde se deben especificar los datos iniciales con los que será instalado el servicio. Los campos son:
• Dominio*: especifica el dominio DNS que se empleará para la gestión del servicio. Es un campo de texto.
• Interfaz de red: se despliega una lista con la(s) interfaces de red del servidor. En caso de no seleccionar ninguna interfaz, entonces la herramienta samba-tool selecciona automáticamente una de las existentes.
Contraseña administrador*
• Contraseña: Es un campo de texto, pero no se muestra el texto, sino asteriscos.</td></tr>
</table>

- Confirmar contraseña: Es un campo de texto, pero no se muestra el texto, sino asteriscos.

La contraseña debe tener complejidad, o sea, debe cumplir con 3 de las siguientes 5 categorías:

1. Caracteres en mayúscula. (A-Z, con acento diacrítico)
2. Caracteres en minúscula. (a-z, con acento diacrítico)
3. Dígitos base 10. (0-9)
4. Caracteres no-alfanuméricos. (~!@#$%^&*_-+=`|\(){}[]:;'"<>,.?/)
5. Cualquier carácter Unicode que se clasifica como un carácter alfabético, pero no es mayúscula o minúscula. Esto incluye caracteres Unicode de idiomas asiáticos.

DNS Backend: muestra las opciones a través de radiobuttons.

- Bind_DLZ
- DNS interno

Tipo de instalación: muestra las opciones a través de radiobuttons.

- Controlador de dominio
- Réplica de controlador de dominio

En caso de seleccionar como tipo de instalación Réplica de controlador de dominio, es suficiente con los parámetros anteriores.

Si el tipo de instalación seleccionado es Controlador de dominio, se muestran otras opciones adicionales necesarias antes de instalar el servicio, a través de checkbox:

- Nivel funcional: especifica las capacidades de dominio o bosque de servicios de dominio de directorio activo (AD DS) que están disponibles y los sistemas operativos Windows Server que se pueden ejecutar en los controladores de dominio del dominio o del bosque.
- Usar xattr: permite interactuar con los atributos de un archivo y emplear las ACLs desde Windows.
- Usar ntvfs: especifica el sistema de archivos. De no habilitarla, por defecto toma la opción s3fs.
- Usar rfc2307: permite almacenar información de usuarios y grupos en el directorio a través de atributos Unix.

Luego debe seleccionar el botón *Instalar*. Entonces se procede a validar cada parámetro y se ejecuta el

comando *samba-tool domain provision*, si se seleccionó Controlador de dominio. Si se seleccionó Réplica de controlador de dominio, se validan los parámetros y se ejecuta el comando *samba-tool domain join*. Cuando culmina exitosamente la instalación, automáticamente se inicia el módulo y se muestra la interfaz inicial del mismo.
Observaciones:
Prototipo elemental de interfaz gráfica de usuario: (ver Anexos del 1 al 3)

Número: 2	**Nombre del requisito:** Desinstalar servicio samba4 en modo Directorio Activo.
Programador: Felipe González Santiago	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 20
Riesgo en Desarrollo: Alto	**Tiempo Real:** 20

Descripción: Permite desinstalar samba4 en modo Directorio Activo como controlador de dominio.

• **Desinstalar servicio**

La funcionalidad comienza cuando el usuario selecciona la opción *Desinstalar* en la interfaz Servidores lógicos en el módulo de samba4. Entonces se realizan una serie de comprobaciones:

-Si el controlador de dominio está en una red en replicación y no posee ningún rol de maestro de operaciones: se muestra un mensaje con el texto: "¿*Está seguro que desea desactivar módulo del servidor*?" Si selecciona el botón Aceptar el sistema procede a desinstalar el servicio a través del comando *samba-tool domain demote*, una vez concluido se muestra un mensaje con el texto: "Módulo desinstalado". Si selecciona el botón Cancelar regresa a la interfaz anterior.

-Si el controlador de dominio está en una red en replicación y posee algún rol de maestro de operaciones: se muestra un mensaje con el texto: "Este controlador de dominio posee los siguientes roles: [roles]. Se sugiere que transfiera estos roles a otro controlador de dominio. ¿*Está seguro que desea desactivar el módulo del servidor*?" Si selecciona el botón Aceptar el sistema procede a desinstalar los servicios *sernet-samba-ad, ntp* y *bind9* a través de los comandos *apt-get remove [servicio]*. Si selecciona el botón *Cancelar* regresa a la interfaz anterior.

-Si el controlador de dominio no está en una red en replicación y posee algún rol de maestro de operaciones: se muestra un mensaje con el texto: "Este controlador de dominio es el último en el dominio y posee los siguientes roles: [roles] ¿Está seguro que desea desinstalarlo?". Si selecciona el botón Aceptar el sistema

procede a desinstalar los servicios *sernet-samba-ad, ntp* y *bind9* a través de los comandos *apt-get remove [servicio]*, una vez concluido se muestra un mensaje con el texto: "*Servicios desinstalados*". Si selecciona el botón *Cancelar* regresa a la interfaz anterior.
Observaciones:
Prototipo elemental de interfaz gráfica de usuario: (ver Anexos del 4 al 6)

Número: 3	**Nombre del requisito:** Adicionar usuario
Programador: Felipe González Santiago	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 40
Riesgo en Desarrollo: Alto	**Tiempo Real:** 38

Descripción: Permite adicionar a un usuario al Directorio Activo.

• **Adicionar usuario**

La funcionalidad comienza cuando el usuario selecciona la opción *Usuario* en el botón *Adicionar* que se encuentra en la barra de gestión de usuarios y computadoras, habiendo previamente seleccionado un contenedor dentro del árbol de directorio. Luego se muestra una interfaz con un conjunto de pestañas, cada una con atributos referentes a un usuario. Las pestañas y sus atributos son:

Pestaña General

- ✔ Usuario*****: es el usuario que identificará a la persona en el Directorio Activo y con el cual podrá autenticarse en el dominio. El atributo LDAP que se modifica es samaccountname. No puede contener exclusivamente puntos (.) o espacios en blanco. Puede tener hasta 20 caracteres, mayúsculas y minúsculas. Excepto: **" / \ [ ] : ; | = , + * ? < >**
- ✔ Nombre del usuario: puede tener hasta 28 caracteres. Es un campo de texto.
- ✔ Apellidos del usuario: puede tener hasta 28 caracteres. Es un campo de texto.
- ✔ Correo: debe ser una dirección de correo válida.
- ✔ Iniciales del usuario: puede tener hasta 6 caracteres. Es un campo de texto.
- ✔ Descripción del usuario: puede tener hasta 1037 caracteres. Es un campo de texto.
- ✔ Establecer contraseña*****: para que el usuario pueda ser creado, antes se le debe establecer una contraseña. Para ello se muestran los campos de texto Nueva contraseña y Confirmar contraseña, donde la contraseña en ambos campos deben coincidir y además cumplir las políticas de complejidad

establecidas en el dominio en el que está siendo creado este usuario.

Pestaña Organización (todos son campos de texto)

- ✔ Cargo del usuario: permite 64 caracteres.
- ✔ Oficina del usuario: permite 128 caracteres.
- ✔ Departamento del usuario: permite 64 caracteres.
- ✔ Compañía: Permite 64 caracteres.
- ✔ Teléfono: Permite 64 caracteres. Solo permite valores numéricos.
- ✔ Otros teléfonos: Permite hasta 5 nuevos teléfonos.

Pestaña Datos Personales

- ✔ Dirección particular del usuario: permite 1024 caracteres.
- ✔ Ciudad del usuario: permite 128 caracteres.
- ✔ Provincia del usuario: permite 128 caracteres.
- ✔ Código postal del usuario: permite 40 caracteres.
- ✔ País/región del usuario: predeterminado. Listado de países.
- ✔ Número(s) de teléfono de la casa del usuario: permite 64 caracteres.
- ✔ Número(s) de teléfono celular del usuario: permite 64 caracteres.

Pestaña Cuenta

- ✔ Ruta del perfil móvil del usuario: especifica una carpeta en el controlador de dominio donde será almacenado el perfil del usuario. Ejemplo de entrada: \\s1.pdc.cu\profiles\%USERNAME%
- ✔ Ruta del directorio home del usuario: especifica el directorio home del usuario. Ej. /home/user
- ✔ Iniciar sesión en: en este campo se especifica las computadoras del dominio en las que el usuario puede iniciar sesión. Por defecto aparecen todas las computadoras del dominio. En el momento que especifica una o varias computadoras, se excluyen todas y solo se habilitan las computadoras especificadas.
- ✔ Deshabilitar cuenta: cuando se deshabilita una cuenta de usuario queda inactiva temporalmente. Lo cual impide que el usuario puede autenticarse en el dominio. Es un checkbox.

- ✔ Cambiar la contraseña al iniciar sesión: fuerza al usuario a cambiar su contraseña la siguiente ocasión que inicie sesión en el dominio. Es un checkbox.
- ✔ No puede cambiar la contraseña: no permite al usuario cambiar su contraseña. Es un checkbox.
- ✔ Contraseña nunca expira. Es un checkbox.
- ✔ Almacenar la contraseña con codificación reversible. Es un checkbox.

Pestaña Grupos

Especifica los grupos a los que pertenece el usuario en cuestión. Se muestran dos listas. Una lista a la izquierda con todos los grupos existentes en el directorio activo, y con la opción *Buscar* que permite filtrar la búsqueda. En la lista a la derecha se muestran los grupos a los que pertenece el usuario. Para adicionar otros grupos o eliminar algunos existentes, se emplean los dos botones: *Añadir* y *Eliminar* que se encuentran entre ambas listas.

Pestaña Atributos Unix (todos son campos de texto)

- ✔ NIS domain: especifica el dominio NIS (*Network Information Service*) al que pertenece el usuario.
- ✔ Uid: un número entero que identifica únicamente a un usuario en un dominio administrativo.
- ✔ Primary group name/Gid: un número entero que identifica únicamente a un grupo en un dominio administrativo.
- ✔ Login shell: especifica la ruta del shell de inicio.
- ✔ Directorio home: especifica la ruta al directorio home.

Observaciones:

Prototipo elemental de interfaz gráfica de usuario: (ver Anexos del 7 al 12)

Número: 4	**Nombre del requisito:** Adicionar grupo
Programador: Yosel Vera González	**Iteración Asignada:** 1
Prioridad: Alta	**Tiempo Estimado:** 30
Riesgo en Desarrollo: Alto	**Tiempo Real:** 30
Descripción: Permite adicionar un grupo al Directorio Activo.	

• Adicionar grupo

La funcionalidad comienza cuando el usuario selecciona la opción G*rupo* en el botón *Adicionar* que se encuentra en la parte superior derecha del árbol de dominio, habiendo previamente seleccionado un contenedor en el árbol de directorio, o sea el contenedor donde será ubicado el grupo. Luego se muestra una interfaz con un conjunto de pestañas, cada una con atributos a llenar referente a un grupo. Las pestañas y sus atributos son:

Pestaña General

- ✔ Nombre del grupo*****: especifica el nombre que identifica al grupo dentro del directorio. Permite 255 caracteres.
- ✔ Tipo de grupo*****: se muestran dos opciones, Seguridad y Distribución.
- ✔ Ámbito del grupo*****: se muestran tres opciones, Local, Global y Universal.
- ✔ Descripción: permite 1037 caracteres.
- ✔ Correo: válida
- ✔ Notas/Información: permite 1037 caracteres.

Pestaña Miembros: muestra 2 listados. El primero a la izquierda muestra todos los usuarios y grupos contenidos en el directorio, excepto el propio grupo en cuestión puesto que un grupo no puede tenerse a sí mismo como miembro; el segundo ubicado a la derecha, contiene los miembros del grupo. Para añadir miembros al grupo, se selecciona(n) el/los usuario(s) y grupo(s) y luego la opción *Añadir* que coloca el/los usuario(s) y grupo(s) en el segundo listado como miembro(s) del grupo en cuestión. Para eliminar miembro(s) del grupo, se selecciona(n) el/los miembro(s) y luego la opción *Eliminar*.

Pestaña Miembro de: Especifica los grupos a los que pertenece el grupo en cuestión. Muestra 2 listados. El primer listado a la izquierda muestra todos los grupos contenidos en el directorio, excepto el propio grupo en cuestión puesto que un grupo no puede tenerse a sí mismo como miembro; el segundo ubicado a la derecha contiene los grupos a los que pertenece el grupo en cuestión. Para añadir el grupo en cuestión como miembro de otros grupos, se selecciona(n) el/los grupo(s) y luego la opción *Añadir* que coloca el/los grupo(s) en el segundo listado. Para eliminar grupos a los que pertenece el grupo en cuestión, se selecciona(n) el/los grupo(s) y luego la opción *Eliminar*.

Pestaña Administrador

Se muestra una lista con todos los usuarios del directorio. A la izquierda de cada usuario se muestra un radiobutton, que permite seleccionar el usuario que será el administrador del grupo en cuestión.

Pestaña Atributos Unix ✔ NIS domain: especifica el dominio NIS (*Network Information Service*) al que pertenece el usuario. ✔ GID. Group ID: un número entero que identifica únicamente a un grupo en un dominio administrativo.
Observaciones:
Prototipo elemental de interfaz gráfica de usuario: (ver Anexos del 14 al 18)

Número: 5	**Nombre del requisito:** Editar configuraciones de contraseñas del dominio.
Programador: Yosel Lázaro Vera González	**Iteración Asignada:** 1
Prioridad: Media	**Tiempo Estimado:** 15
Riesgo en Desarrollo: Medio	**Tiempo Real:** 14

Descripción: Permite mostrar y editar las configuraciones de las contraseñas del dominio.

La funcionalidad comienza cuando el usuario selecciona en el menú lateral izquierdo la opción Configuraciones del dominio, posteriormente se despliega un submenú con tres opciones las cuales se describen a continuación.

Contraseñas

La funcionalidad comienza cuando en el submenú desplegado se selecciona la opción Contraseñas. En el panel central se muestra una tabla con los valores relacionados con las configuraciones de las contraseñas del dominio. A continuación se describen cada uno de estos valores.

- **Complejidad**: es un valor booleano que establece si los requisitos de complejidad son aplicados a las contraseñas del dominio.
- **Longitud del historial**: es un valor numérico que determina la cantidad de contraseñas antiguas que deben guardarse, un usuario del dominio no puede establecerse una contraseña igual a las anteriores guardadas. Puede tener valores entre 0 y 24.
- **Tiempo mínimo de cambio**: es un valor numérico entre 0 y 998 que especifica la cantidad de días en que la contraseña puede volver a cambiarse.
- **Longitud mínima**: valor numérico que especifica la cantidad mínima de caracteres que deben tener las contraseñas del dominio. Admite valores en el rango de 0 a 14.
- **Tiempo de expiración**: valor numérico entre 0 y 999 que determina la cantidad de días de validez que

posee una contraseña.

- **Encriptar contraseña**: valor booleano que especifica si las contraseñas son almacenadas en texto plano o son encriptadas.
- **Duración de bloqueo**: valor numérico entre 0 y 99999 que especifica la cantidad de minutos que las cuentas de usuario son bloqueadas después de exceder el límite de intentos de autenticación.
- **Reiniciar cuenta después**: valor numérico entre 0 y 99999 que especifica el tiempo en minutos que debe transcurrir para que la cantidad de intentos fallidos de autenticación se reinicie a 0.
- **Límite de intentos**: valor numérico entre 0 y 65535 que determina la cantidad de intentos fallidos de autenticación que puede realizar un usuario antes de que se bloquee la cuenta.

Al finalizar las modificaciones presiona en el botón guardar para realizar los cambios en el servidor.

Observaciones:

Prototipo elemental de interfaz gráfica de usuario: (ver Anexo 13)

2.3 Arquitectura del módulo

El proceso de diseño de la arquitectura debe decidir cuál funcionalidad es la más importante a desarrollar. Además, define cuáles son los componentes más básicos del sistema y cómo se relacionan entre ellos para implementar la funcionalidad [35].

Se determina como arquitectura del módulo, la establecida para la herramienta base HMAST (ver Figura 2), para lograr la consistencia con los componentes del sistema. Se emplea una arquitectura N-Capas orientada al Dominio, la cual tiene como objetivo estructurar de forma clara la complejidad de una aplicación empresarial basada en las diferentes capas de la arquitectura siguiendo el patrón N-Capas y las tendencias de arquitecturas orientadas al dominio [35].

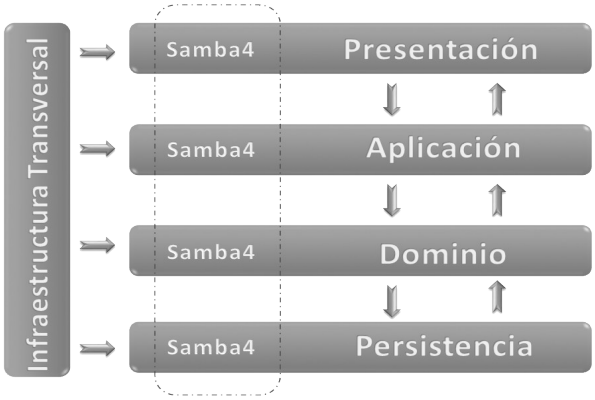


Figura 2. Arquitectura del módulo.

2.4 Diagrama de clases por paquete

El diagrama de paquetes muestra las agrupaciones lógicas en que está dividido el sistema, así como las dependencias entre dichas agrupaciones. Para el diseño del diagrama de paquetes se toma como referencia la arquitectura anteriormente propuesta, de modo que existe un paquete por cada capa, y dentro de cada uno de estos paquetes otro, llamado *samba4* que contiene todas las clases pertenecientes al módulo.

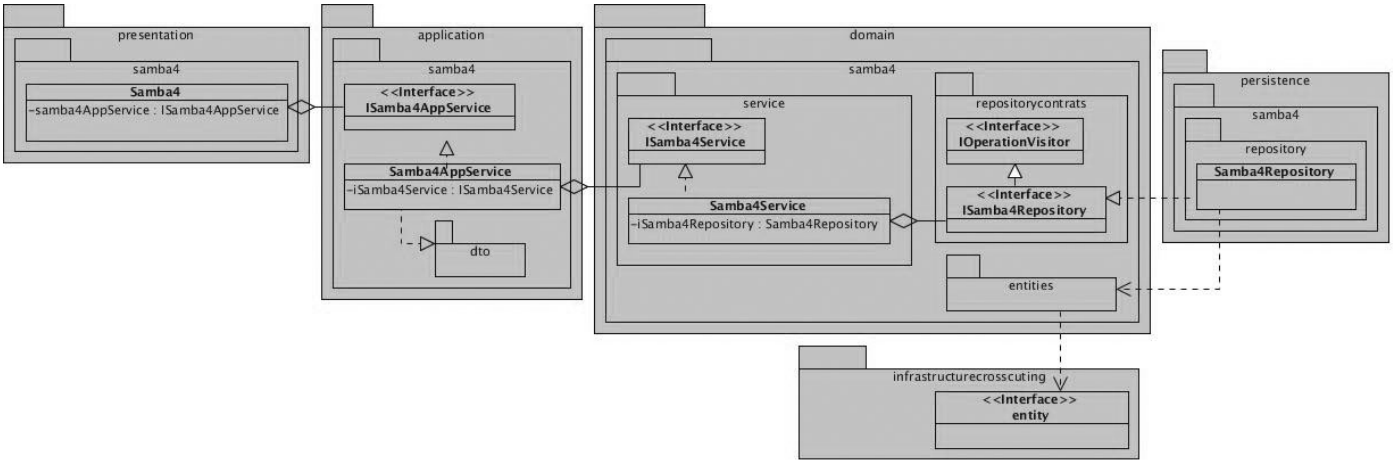


Figura 3. Diagrama de clases por paquete.

A continuación se realiza una descripción del módulo teniendo en cuenta los paquetes que contiene y la organización de sus elementos. En las siguientes figuras se representan los elementos del diagrama de paquetes que intervienen en la historia de usuario *Mostrar objetos del Directorio Activo* y otros que son relevantes en el desarrollo de diferentes funcionalidades del módulo.

En la Figura 4 se representa la capa Aplicación, esta contiene el paquete *samba4* donde se encuentra la interfaz *ISamba4AppService* que define los métodos que serán invocados en el futuro desde la capa Presentación. También se encuentra la clase *Samba4AppService,* que se encarga de realizar la implementación de los métodos de la interfaz *ISamba4AppService*. En *samba4* se encuentra también el paquete *dto*, que contiene los objetos de transferencia de datos, en este caso en particular el objeto *ObjectActiveDirectoryTreeDTO* que es utilizado para mostrar los objetos en la representación de la estructura lógica del Directorio Activo. Los objetos de transferencia de datos son enviados desde y hacia la capa de Presentación, la cual depende de *ISamba4AppService.* La clase encargada de realizar las conversiones de entidades a DTO y viceversa es la clase *Samba4AppService.*

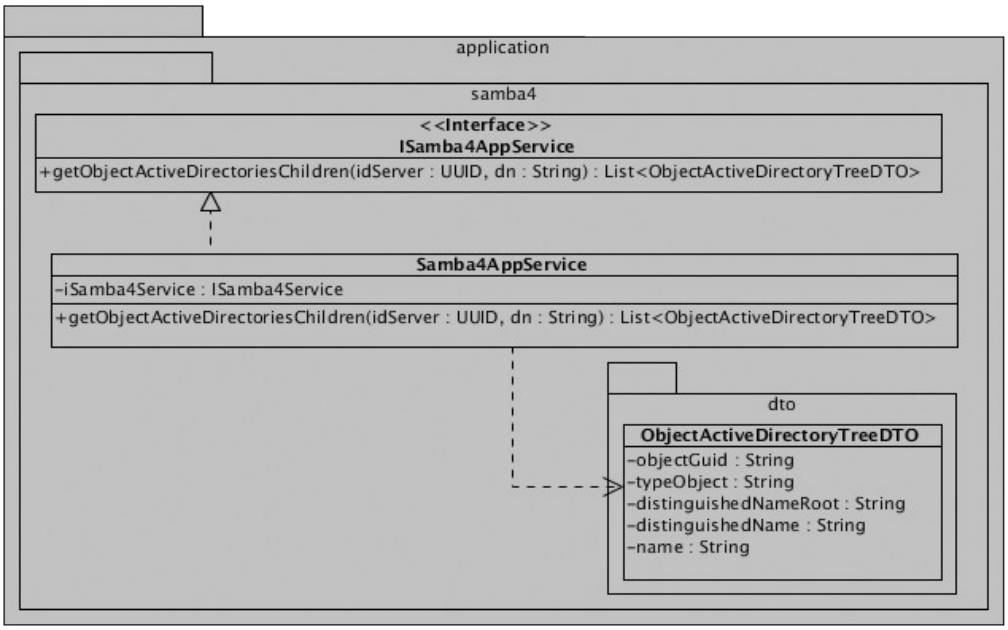


Figura 4. Diagrama de clases de la capa Aplicación.

La Figura 5 representa la capa de Dominio que se relaciona con la capa de Aplicación a través de una relación de agregación con la interfaz *ISamba4Service* mediante la inyección de dependencia. Esta capa contiene dentro de *samba4,* tres paquetes denominados *entities, services y repositorycontrats*, los cuales se describen a continuación.

- *entities*: se definen las entidades del módulo, que representan los objetos del dominio. En la Figura 5 se representan las entidades *Samba4* y *ObjectActiveDirectory* pues son las que intervienen en la historia de usuario Mostrar objetos del Directorio Activo. Las entidades restantes que aparecen son las que intervienen en el patrón *Visitor*, implementado en el módulo, con el objetivo de almacenar y ejecutar las operaciones LDAP. En cada entidad se realizan validaciones atómicas para cada atributo.
- *services*: se define la interfaz *ISamba4Service* y la clase *Samba4Service* que es donde se implementan los métodos definidos en esta interfaz. El objetivo de *Samba4Service* es realizar las validaciones que no se implementen en las entidades. Además, tiene una relación de agregación con *ISamba4Repository* y se realiza mediante una inyección de dependencias.
- *repositorycontrats*: define los contratos de repositorios en *ISamba4Repository* pero no realiza su implementación.

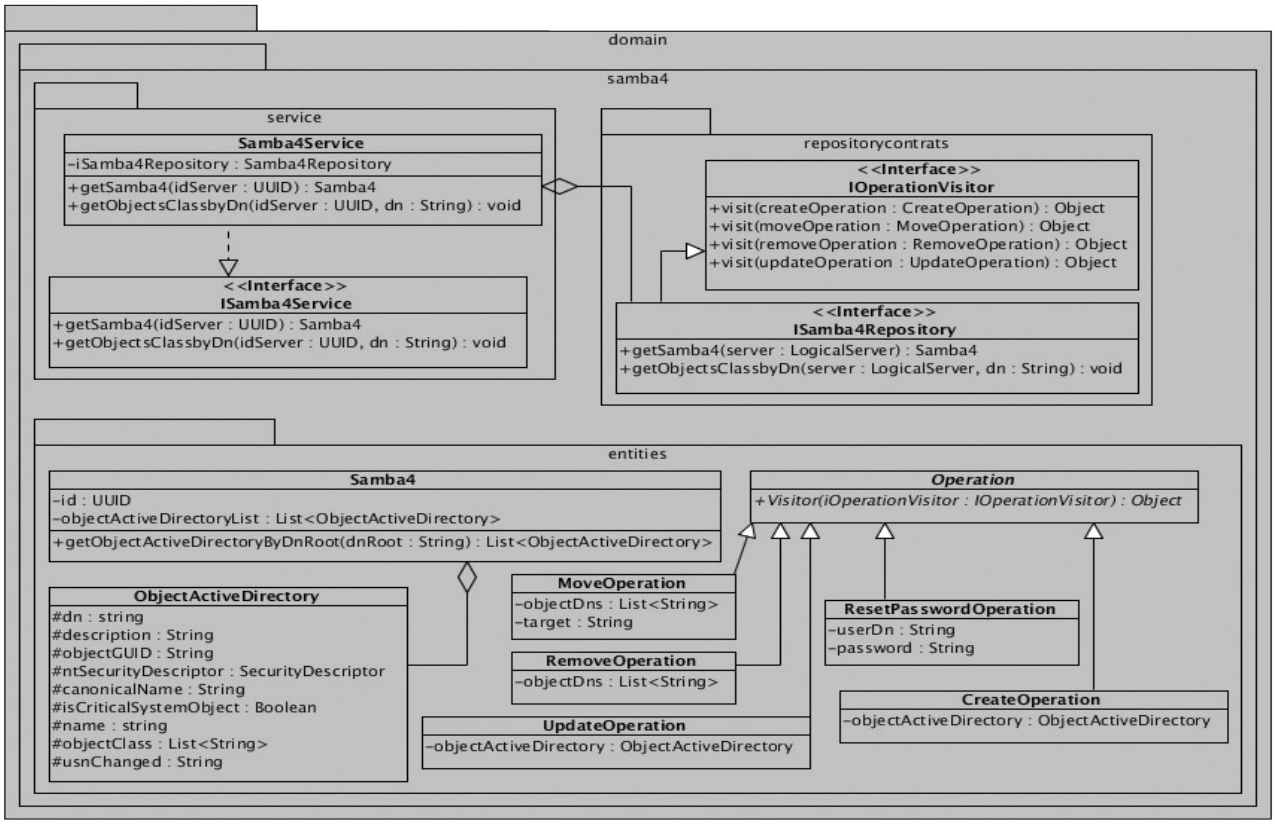


Figura 5. Diagrama de clases de la capa Dominio.

En la Figura 6 se representa el paquete *samba4* correspondiente a la capa Persistencia. Contiene el paquete *repository* en el cual se encuentra la clase *Samba4Repository*, que se encarga de implementar todos los métodos establecidos en la interfaz *ISamba4Repository* con el objetivo de lograr la persistencia de los datos que se manejan en el módulo. Para lograr su objetivo, *Samba4Repository* emplea dos clases definidas en el paquete *mappers.* Una es *LoadMappers* que se utiliza para realizar la lectura de los objetos del Directorio Activo y transformarlos en entidades del dominio. La otra clase es *WritterMappers* encargada de la transformación de las entidades del dominio en contextos LDAP reconocidos por el Directorio Activo. Estas clases fueron creadas debido a la cantidad de objetos y atributos que se definen en el servicio y la complejidad de su almacenamiento. También emplean otro conjunto de clases definidas en el paquete *utils,* las cuales realizan el tratamiento de atributos binarios, que representan estructuras de datos, y atributos numéricos que representan banderas para manejar el comportamiento de un objeto determinado.

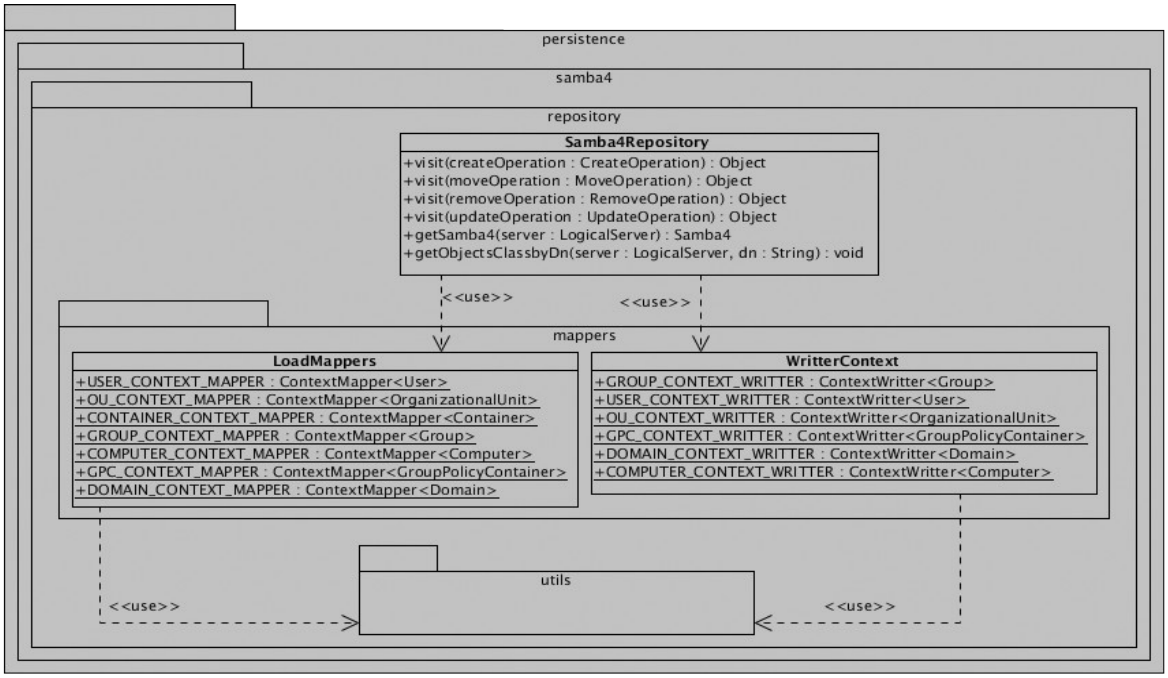


Figura 6. Diagrama de clases de la capa Persistencia.

2.5 Patrones de diseño

Los patrones de diseño son principios generales de soluciones que aplican ciertos estilos que ayudan a la creación de software. Brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares [36]. En el modelo de diseño del módulo se aplican patrones de asignación de responsabilidades (GRASP[31]) y patrones GoF[32], los cuales favorecen un mejor ordenamiento, estructuración y entendimiento del diseño.

2.5.1 Patrones GRASP

Los patrones GRASP describen los principios fundamentales del diseño de objetos y la asignación de responsabilidades, expresados como patrones. A continuación se describen los utilizados en el desarrollo del módulo [37].

- **Experto en información o experto:** se utiliza para la asignación de responsabilidades relacionadas con la obtención de información. Conduce a diseños donde los objetos del software

31 GRASP: Patrones Generales de Software para Asignar Responsabilidades, del inglés *General Responsibility Assignment Software Patterns.*

32 GoF: Patrones de diseño, del inglés *Gang of Four,* nombre con el que se conoce comúnmente a los autores del libro *Desing Patterns*.

realizan aquellas operaciones que normalmente se hacen a los objetos inanimados del mundo real que representan. Se mantiene el encapsulamiento de la información logrando un bajo acoplamiento entre los objetos y se distribuye el comportamiento entre las clases, lo que estimula las definiciones de clases más cohesivas.

- **Creador**: guía la asignación de responsabilidades relacionadas con la creación de objetos. La intención básica es encontrar un creador que necesite conectarse al objeto creado en alguna situación. Eligiéndolo como el creador se favorece el bajo acoplamiento. Su utilización se evidencia cuando en la clase *Samba4AppService* de la capa de Aplicación se realiza la creación de los DTO utilizando para ello las entidades necesarias del dominio. Esta clase es la encargada de crear un DTO a partir de una entidad o crear la entidad a partir de un DTO.
- **Bajo acoplamiento**: consiste en asignar responsabilidades de manera que el acoplamiento permanezca bajo. El acoplamiento es una medida de la fuerza con que un elemento está conectado a, tiene conocimiento de, confía en, otros elementos. El uso de este patrón permite la reutilización de las clases y que no se afecten por cambios que se realicen en otros componentes. Este patrón se emplea en las distintas capas del módulo mediante el uso de interfaces que relacionan una capa con otra de forma que dichas relaciones no se establezcan directamente hacia las clases. Las conexiones se realizan a través del mecanismo de inyección de dependencias. Esto se evidencia en el paquete *repositorycontrats*, donde se define la interfaz *ISamba4Repository,* y la implementación de sus métodos es realizada por la clase *Samba4Repository* en la capa de Persistencia.
- **Alta Cohesión:** la cohesión es una medida de la fuerza con la que se relacionan y del grado de focalización de las responsabilidades de un elemento. Las clases tienen una responsabilidad moderada en un área funcional y colaboran con otras clases para llevar a cabo una tarea determinada. Permite que las clases sean fáciles de entender, mantener y reutilizar. Se emplea en la clase *Samba4Repository*, que tiene la responsabilidad de interactuar con los objetos del Directorio Activo y realizar operaciones de creación, lectura, actualización y eliminación. Para el cumplimiento de esta tarea utiliza las clases *LoadMappers* y *WritterMappers*, delegando en estas, las operaciones de conversión de los atributos.

2.5.2 Patrones GoF

Los patrones GoF son patrones de asignación de responsabilidades y se pueden categorizar en tres grupos teniendo en cuenta su propósito: creacionales, estructurales y de comportamiento [37].

- **Patrón Visitor**: es un patrón de comportamiento que permite modelar distintas operaciones a llevar a cabo sobre una misma estructura de datos permitiendo definir nuevas operaciones sin modificar la estructura del diseño. Es usado cuando muchas operaciones distintas y no relacionadas se quieren aplicar a un objeto en una estructura de datos y se desea evitar la contaminación de las clases con esas operaciones. El patrón *Visitor* se utiliza en la entidad *Operation* y en todas las clases que extienden de esta, las cuales agrupan las operaciones asociadas a la gestión de los objetos del directorio. La clase *Samba4Repository* es la encargada de implementar las operaciones definidas en *IOperationVisitor.* Para ello, *ISamba4Repository* debe extender de *IOperationVisitor.*
- **Patrón Solitario (Singleton)**: es un patrón de tipo creacional con el objetivo de garantizar la existencia de una única instancia para una clase y posibilitar el acceso global a dicha instancia. Su empleo se evidencia cuando se realiza una conexión LDAP o SSH a un servidor, pues con una única instancia del objeto conexión se realizan las operaciones sobre este. Tal es el caso de la clase *LdapConnection*, donde con una única instancia se realizan todas las operaciones LDAP en el servidor.

Conclusiones parciales

A partir del estudio realizado en el Capítulo 1 sobre el servicio Directorio Activo y sobre Samba4 como alternativa libre, se definieron las características y funcionalidades del módulo propuesto, obteniendo 61 requisitos funcionales y 6 requisitos no funcionales. Se asumió como arquitectura a emplear la N-Capas orientada al Dominio, coincidiendo con la empleada en la herramienta base HMAST lo cual permitió mantener la homogeneidad del módulo con la herramienta base. Dicha arquitectura se tomó como base para el diseño del diagrama de clases por paquete. Además, se tuvieron en cuenta varios patrones de diseño que favorecen la reutilización y mantenibilidad de código.

CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBAS AL MÓDULO

En el presente capítulo se presentan los estándares de codificación empleados para el desarrollo del módulo, para mantener la uniformidad con la codificación de los restantes módulos. Además se establece la estrategia de pruebas y se documenta la realización de las pruebas seleccionadas, entre ellas el desarrollo de una prueba experimental con el objetivo de asegurar la correcta ejecución de la migración de los datos y servicios hacia Samba4. Para comprender la distribución física de los componentes que conforman el sistema, también se modela el diagrama de despliegue.

3.1 Estándar de codificación

Las técnicas de codificación contribuyen a una mejor compresión del código fuente [38]. Los estándares que se emplean en el módulo se ajustan a las pautas definidas en el expediente de proyecto de HMAST.

- **Asignación de nombres**. Emplear descriptores en inglés. Evitar nombres largos y que difieran en una letra o en el uso de mayúsculas. Para nombrar las funciones y variables se escribe con la primera palabra en minúscula, en caso de que sea un nombre compuesto se utiliza la notación *CamelCase*[33].
- **Ficheros de código fuente**. Cada fichero contiene una única clase o interfaz. Si hay una clase privada o una interfaz asociada a una clase pública se puede poner en el mismo fichero. La clase pública debe ser la primera.
- **Identación**. La unidad de indentación de bloques de sentencias son 4 espacios.
- **Comentarios**. Los comentarios deben añadir claridad al código. Deben contar el por qué y no el cómo. Deben ser concisos.
- **Declaraciones**. Se debe declarar cada variable en una línea distinta, de esta forma cada variable se puede comentar por separado.
- **Continuidad de las líneas largas**. Cuando una sentencia no quepa en una única línea se debe fraccionar después de una coma, después de un operador y alinear la nueva línea con el comienzo de la expresión al mismo nivel de la linea anterior.
- **Longitud de la línea**. Limitar todas las líneas a un máximo de 120 caracteres.

33 CamelCase es la práctica de escribir frases o palabras compuestas eliminando los espacios y poniendo en mayúscula la primera letra de cada palabra.

- **Nombres de componentes**. Todos los paquetes comienzan con cu.uci.hmast.xxx.yyy.zzz.kkk

  xxx →presentation, application, domain, persistence.

  yyy →nombre del módulo (samba4).

  zzz→elementos que pueden contener los componentes verticales (entitys, repositorys).

  kkk → clases o subpaquetes.

3.2 Estrategia de pruebas

La prueba de software es una actividad realizada para evaluar la calidad de un producto y mejorarlo a través de la identificación de problemas. Consiste en la verificación dinámica del comportamiento de un programa con un conjunto finito de casos de prueba. Las pruebas no aseguran la ausencia de errores, sino que están orientadas a demostrar que existen defectos en el software [39].

La metodología AUP-UCI desagrega las pruebas en tres disciplinas: internas, liberación y aceptación. Las pruebas de liberación son diseñadas y ejecutadas por una entidad certificadora externa, por lo que en el presente trabajo no son analizadas. La estrategia de pruebas de software que se emplea propone pruebas de alto y bajo nivel que incluyen todos los componentes del módulo. Esta estrategia tiene un enfoque incremental y se representa mediante una espiral que analiza el código, el diseño, los requisitos y la ingeniería del sistema para lo cual se emplean pruebas de unidad, de integración, de validación y de sistema, respectivamente. En la Figura 7 se representa la estrategia de pruebas descrita anteriormente.

Figura 7. Estrategia de pruebas de software.

La prueba de unidad analiza cada módulo individualmente, asegurando que funciona adecuadamente como una unidad, haciendo un uso intensivo del método de prueba de caja blanca y ejercitando caminos específicos de la estructura de control del módulo para asegurar un alcance completo y una detección máxima de errores. La prueba de integración se enfoca al diseño y la construcción de la arquitectura del software. La prueba de validación permite validar los requisitos establecidos durante el análisis de

requisitos, comparándolos con el sistema que ha sido construido. Finalmente se prueba el software como un todo empleando la prueba del sistema [40].

3.2.1 Prueba de unidad

La prueba de unidad que se realiza hace uso del método de caja blanca y de la técnica del camino básico. El método del camino básico permite al diseñador de casos de prueba derivar una medida de complejidad lógica de un diseño procedural y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución. Los casos de prueba derivados del conjunto básico garantizan que durante la prueba se ejecute por lo menos una vez cada sentencia del programa [40].

Las unidades de prueba más pequeñas son las operaciones dentro de la clase. A continuación se realiza la técnica del camino mínimo a la operación `getMembersListOfGroup` de la clase `Samba4Repository` que permite obtener la lista de miembros de un grupo específico.

(1)	`public List <ObjectActiveDirectory> getMembersListOfGroup(LogicalServer logicalServer, UUID uuidGroup) throws EEntityNotExist, InvalidNameException {`
(2)	`String base = this.getBase(logicalServer);`
(3)	`List<ObjectActiveDirectory> groupList = entityManager.createQuery("select o from ObjectActiveDirectory o where o.idServer=:idServer and o.uuid=:uuid" , ObjectActiveDirectory.class).setParameter("idServer" , logicalServer.getId()).setParameter("uuid", uuidGroup).getResultList();`
(4)	`List<ObjectActiveDirectory> members = new LinkedList<>();`
(5)	`if(!groupList.isEmpty()){`
(6)	`Group group = (Group)groupList.get(0);`
(7)	`if(group.getMembers().isEmpty()) {`
(8)	`if(this.existsObjectInActiveDirectoryByGuid(logicalServer, group.getObjectGUID())) {`
(9)	`List<String> membersDn = logicalServer.getLdapPersistens().getLdapTemplate().search(LdapQueryBuilder.query().base(base).where("objectGuid").is(group.getObjectGUID()) , LoadMappers.MEMBERS_LIST_MAPPER).get(0);`
(10)	`String objectType = "";`
(11)	`ObjectActiveDirectory objectActiveDirectory = null;`
(12)	`for(String dn : membersDn) {`
(13)	`objectType = logicalServer.getLdapPersistens().getLdapTemplate().lookup(dn, LoadMappers.OBJECT_CLASS_MAPPER);`
(14)	`objectActiveDirectory = this.getValidObjectActiveDirectory(objectType,`

	logicalServer, dn);
(15)	**if**(!**this**.existsObjectInDataBaseByObjectGuid(logicalServer, objectActiveDirectory.getObjectGUID())) {
(16)	objectActiveDirectory.setUuid(UUID.*randomUUID*());
(17)	objectActiveDirectory.setIdServer(logicalServer.getId());
(18)	**this**.saveObjectActiveDirectory(objectActiveDirectory , logicalServer);
(19)	members.add(objectActiveDirectory);
(20)	group.getMembers().add(objectActiveDirectory.getObjectGUID());
(21)	}
(22)	**else**{
(23)	members.add(**this**.getObjectByGuid(logicalServer, objectActiveDirectory.getObjectGUID()));
(24)	}
(25)	}
(26)	**entityManager**.merge(group);
(27)	**return** members;
(28)	}
(29)	**else**
(30)	**throw new** EEntityNotExist("**No existe el grupo**");
(31)	}
(32)	**else** {
(33)	**for**(String guid: group.getMembers())
(34)	members.add(**this**.getObjectByGuid(logicalServer , guid));
(35)	**return** members;
(36)	}
(37)	}
(38)	**throw new** EEntityNotExist("**No existe el grupo**");
(39)	}

Luego de numerar las líneas de código, se diseña la gráfica del programa que describe el flujo de control lógico empleando nodos y aristas.

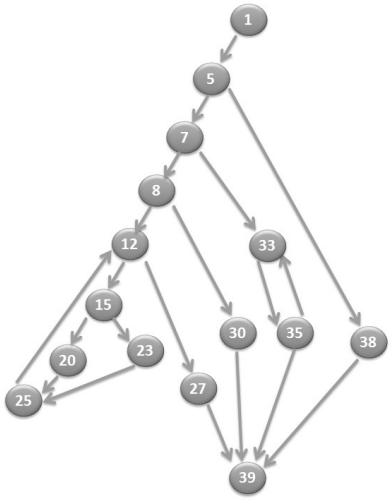


Figura 8.Grafo de flujo.

A partir del grafo obtenido con 15 nodos y 20 aristas se calcula la complejidad ciclomática *V(G)*, la cual constituye una métrica de software que proporciona una medida cuantitativa de la complejidad lógica del programa [40].

V(G) = cantidad_aristas – cantidad_nodos + 2

V(G) = 20 – 15 + 2 = 7

Una ruta independiente es cualquier ruta del programa que ingrese al menos un nuevo conjunto de instrucciones de procesamiento o una nueva condición [40]. La cantidad de rutas independientes son establecidas por la complejidad ciclomática, por tanto se identifican siete rutas, tal y como se muestra en la siguiente tabla.

Tabla 3. Listado de caminos básicos.

No. Ruta	Camino
1.	1-5-38-39
2.	1-5-7-33-35-39
3.	1-5-7-33-35-33-35-39
4.	1-5-7-8-30-39
5.	1-5-7-8-12-27-39
6.	1-5-7-8-12-15-20-25-12-27-39

7.	1-5-7-8-12-15-23-25-12-27-39

El valor de V(G) ofrece además un límite superior del número de pruebas de debe diseñarse y ejecutarse para garantizar la cobertura de todas las instrucciones [40]. Por tanto, se diseñan casos de pruebas para ser aplicados a cada ruta independiente.

Caso de Prueba de Unidad	
No. ruta: 1	**Ruta:** 1-5-38-39
Nombre de la persona que realiza la prueba: Felipe González Santiago	
Descripción de la prueba: Obtener la lista de miembros de un grupo.	
Entrada: Se envía como parámetro el uuid de un grupo que no existe.	
Resultado esperado: El sistema debe mostrar una excepción con el mensaje "*No existe el grupo*"	
Evaluación de la Prueba: Satisfactoria. Se obtuvo el mensaje esperado.	

Caso de Prueba de Unidad	
No. ruta: 2	**Ruta:** 1-5-7-33-35-39
Nombre de la persona que realiza la prueba: Felipe González Santiago	
Descripción de la prueba: Obtener la lista de miembros de un grupo.	
Entrada: Se envía como parámetro el uuid de un grupo que existe en la base de datos, además ya se encuentra almacenada su lista de miembros en base de datos.	
Resultado esperado: El sistema debe recorrer la lista de miembros del grupo, y retornar una de lista de *ObjectActiveDirectory*.	
Evaluación de la Prueba: Satisfactoria. Se obtuvo la lista de miembros del grupo.	

Caso de Prueba de Unidad	
No. ruta: 3	**Ruta:** 1-5-7-33-35-33-35-39
Nombre de la persona que realiza la prueba: Felipe González Santiago	
Descripción de la prueba: Obtener la lista de miembros de un grupo.	
Entrada: Se envía como parámetro el uuid de un grupo que existe en la base de datos, además ya se encuentra almacenada su lista de miembros en base de datos y esta lista tiene una longitud mayor que uno.	

Resultado esperado: El sistema debe recorrer la lista de miembros del grupo, y retornar una de lista de *ObjectActiveDirectory.*
Evaluación de la Prueba: Satisfactoria. Se obtuvo la lista de miembros del grupo.

Caso de Prueba de Unidad	
No. ruta: 4	**Ruta:** 1-5-7-8-30-39
Nombre de la persona que realiza la prueba: Felipe González Santiago	
Descripción de la prueba: Obtener la lista de miembros de un grupo.	
Entrada: Se envía como parámetro el uuid de un grupo que existe en la base de datos, pero que no existe en el Directorio Activo (ha sido borrado de forma externa).	
Resultado esperado: El sistema debe mostrar una excepción con el mensaje *"No existe el grupo"*	
Evaluación de la Prueba: Satisfactoria. Se obtuvo el mensaje esperado.	

Caso de Prueba de Unidad	
No. ruta: 5	**Ruta:** 1-5-7-8-12-27-39
Nombre de la persona que realiza la prueba: Felipe González Santiago	
Descripción de la prueba: Obtener la lista de miembros de un grupo.	
Entrada: Se envía como parámetro el uuid de un grupo que existe en la base de datos, que no posee miembros, o sea su lista de miembros está vacía.	
Resultado esperado: El sistema debe retornar una lista vacía.	
Evaluación de la Prueba: Satisfactoria. Se obtuvo la lista vacía de miembros.	

Caso de Prueba de Unidad	
No. ruta: 6	**Ruta:** 1-5-7-8-12-15-20-25-12-27-39
Nombre de la persona que realiza la prueba: Felipe González Santiago	
Descripción de la prueba: Obtener la lista de miembros de un grupo.	
Entrada: Se envía como parámetro el uuid de un grupo que existe en la base de datos cuya lista de miembros no se ha almacenado en la base de datos.	
Resultado esperado: El sistema debe salvar en la base de datos y retornar la lista de miembros del grupo.	
Evaluación de la Prueba: Satisfactoria. Se obtuvo la lista de miembros y fueron almacenados en la base de datos.	

Caso de Prueba de Unidad

No. ruta: 7	**Ruta:** 1-5-7-8-12-15-23-25-12-27-39
Nombre de la persona que realiza la prueba: Felipe González Santiago	
Descripción de la prueba: Obtener la lista de miembros de un grupo.	
Entrada: Se envía como parámetro el uuid de un grupo que existe en la base de datos cuya lista de miembros ya está almacenada en la base de datos.	
Resultado esperado: El sistema debe retornar la lista de miembros del grupo.	
Evaluación de la Prueba: Satisfactoria. Se obtuvo la lista de miembros.	

Se realizaron tres iteraciones de la prueba unitaria. En la primera iteración se detectaron 4 no conformidades; en la segunda, una no conformidad la cual fue resuelta para la tercera iteración. Las no conformidades detectadas estaban asociadas a errores de validación y al tratamiento de las excepciones desde el código.

3.2.2 Prueba de integración

La prueba de integración es una técnica para construir la arquitectura del software. El objetivo es tomar componentes a los que se aplicó una prueba de unidad y construir una estructura de programa que determine el diseño [40]. Asumiendo la arquitectura por capas empleada y que el módulo debe incorporarse a una herramienta base, se emplea una estrategia de integración ascendente, donde los componentes se integran de abajo hacia arriba.

En el contexto de la prueba de integración ascendente se realiza la prueba de regresión que permite ejecutar nuevamente el mismo subconjunto de pruebas que ya se ha aplicado para asegurar que los cambios no han propagado efectos colaterales indeseables [40].

3.2.3 Prueba de validación

Con el objetivo de validar que el sistema cumple con el funcionamiento esperado y permitir que el usuario determine su aceptación desde el punto de vista de su funcionalidad y rendimiento, se realizan pruebas de aceptación. Para ello se tuvieron en cuenta las historias de usuario de mayor criticidad. A continuación se presentan los casos de prueba correspondientes a las historias de usuario *Editar unidad organizativa, Adicionar usuario* y *Gestionar configuraciones de contraseñas del dominio*, a partir de los cuales el cliente ejecutó las pruebas.

Caso de Prueba de Aceptación

Código Caso de Prueba: HMAST-Samba4-1	**Nombre Historia de Usuario:** Editar unidad organizativa.
Nombre de la persona que realiza la prueba: Yoandy Pérez Villazón	
Descripción de la prueba: Editar los parámetros de una unidad organizativa.	
Condiciones de Ejecución: El usuario debe estar autenticado en el sistema HMAST. Debe adicionar un servidor lógico y activar el módulo Samba4.	
Entrada/Pasos de Ejecución: 1. Acceder al módulo Samba4. 2. Opción Gestión de objetos. 3. Seleccionar en el árbol una unidad organizativa. 4. Seleccionar el botón Editar 5. Aceptar los cambios. 6. Aplicar los cambios en el servidor.	
Resultado esperado: Los parámetros de la unidad organizativa en el controlador de dominio deben tomar los nuevos valores y se muestra un mensaje notificándolo.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: HMAST-Samba4-2	**Nombre Historia de Usuario:** Adicionar usuario.
Nombre de la persona que realiza la prueba: Yoandy Pérez Villazón	
Descripción de la prueba: Adicionar un usuario con todos los atributos al directorio.	
Condiciones de Ejecución: El usuario debe estar autenticado en el sistema HMAST. Debe adicionar un servidor lógico y activar el módulo Samba4.	
Entrada/Pasos de Ejecución: 1. Acceder al módulo Samba4. 2. Opción Gestión de objetos. 3. Seleccionar en el árbol una unidad organizativa. 4. Seleccionar el botón Adicionar. 5. Seleccionar la opción usuario. 6. Aceptar los cambios. 7. Aplicar los cambios en el servidor.	
Resultado esperado: El usuario es adicionado al directorio.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código Caso de Prueba: HMAST-Samba4-3	**Nombre Historia de Usuario:** Gestionar configuraciones de contraseñas del dominio.
Nombre de la persona que realiza la prueba: Yoandy Pérez Villazón	
Descripción de la prueba: Editar los parámetros complejidad, encriptar, longitud del historial, longitud mínima, tiempo mínimo de cambio, tiempo de expiración, duración del bloqueo de la cuenta, límite de intentos y reiniciar cuenta.	
Condiciones de Ejecución: El usuario debe estar autenticado en el sistema HMAST. Debe adicionar un servidor y activar el módulo Samba4.	
Entrada/Pasos de Ejecución: 1. Acceder al módulo Samba4. 2. Opción Contraseñas en el menú lateral izquierdo. 3. Editar los parámetros de las contraseñas. 4. Aceptar los cambios. 5. Aplicar los cambios en el servidor.	
Resultado esperado: Los parámetros de las contraseñas deben tomar los nuevos valores y se muestra un mensaje notificándolo.	
Evaluación de la Prueba: Satisfactoria.	

Como parte de las pruebas de aceptación ejecutadas se realizaron 3 iteraciones, detectando diecisiete, cuatro y cero no conformidades en la primera, la segunda y la tercera iteración respectivamente.

3.2.4 Prueba de sistema

La fiabilidad de un sistema es una medida de su conformidad de acuerdo con una especificación [41]. La variable de mayor relevancia en el proceso de migración hacia Samba4 es la integridad de los datos. Por ello, se diseña un experimento como parte de la prueba de fiabilidad del sistema que permite validar dicha variable.

Prueba de fiabilidad (experimento)

Para migrar los datos y servicios del Directorio Activo hacia Samba4 se emplea el mecanismo *samba-tool domain join* descrito en la sección 1.3.1 del presente documento. El éxito de la migración se determina a partir de las siguientes comprobaciones.

1. Se encuentran todos los usuarios, grupos y computadoras.
2. Se mantiene la pertenencia de usuarios a los grupos existentes.
3. Se mantiene la estructura del árbol del Directorio Activo, así como los objetos dentro de ella.

4. Las claves asignadas a los usuarios se conservan y coinciden con las anteriores.
5. Funciona la replicación de objetos en ambas direcciones. Al ser creados en el controlador de dominio de Samba4 se replican en el Windows Server y viceversa.
6. Los objetos conservan todos los atributos que tenían anteriormente y se encuentran funcionales.
7. Se migraron todos los registros de DNS pertenecientes a la zona directa del controlador de dominio Windows Server.

El escenario de prueba seleccionado es el siguiente:

Tabla 4. Escenario de prueba.

Componente	Servidor Windows Server	Servidor GNU/Linux
Sistema operativo	Windows Server 2003	Ubuntu Server 14.04 a 64 bits
Memoria RAM	1GB	1GB
Microprocesador	Core Intel 3 a 3.3 GHz	Core Intel 3 a 3.3 GHz

En la siguiente tabla se muestran los resultados obtenidos luego de ejecutar la migración del Directorio Activo hacia Samba4.

Tabla 5. Resultados del experimento de migración.

No	Criterio	Windows Server 2003	Ubuntu Server 14.04
1.	Cantidad de usuarios	1000	1000
2.	Cantidad de grupos	100	100
3.	Cantidad de computadoras	50	50
4.	Membresía de grupos	Sí	Sí
5.	Estructura del árbol de directorio	Sí	Sí
6.	Autenticación de usuarios con sus claves	Sí	Sí
7.	Replicación de objetos (de Ubuntu Server hacia Windows Server)	Sí	Sí
8.	Replicación de objetos (de Windows Server hacia Ubuntu Server)	Sí	Sí
9.	Los objetos conservan los valores de sus atributos	Sí	Sí
10.	Zonas y registros DNS	Sí	Sí

La prueba realizada al proceso de migración del Directorio Activo hacia Samba4 resultó exitosa. Se comprobó la existencia de todos los objetos con sus atributos, así como la estructura del árbol de

directorio. Con ello, se validó que el módulo y el mecanismo de migración escogido permiten mantener la integridad de los datos.

3.3 Diagrama de despliegue

El diagrama de despliegue muestra las relaciones físicas de los distintos nodos que componen un sistema [42]. La herramienta HMAST se ejecuta sobre un servidor web Tomcat sobre el sistema operativo GNU/Linux Nova. El usuario accede a HMAST mediante un navegador web y empleando conexiones seguras con el protocolo HTTPS. A través del módulo de Directorio Activo de HMAST el usuario accede al servidor controlador de dominio y realiza las operaciones de migración y administración. A continuación se presenta la distribución física del módulo.

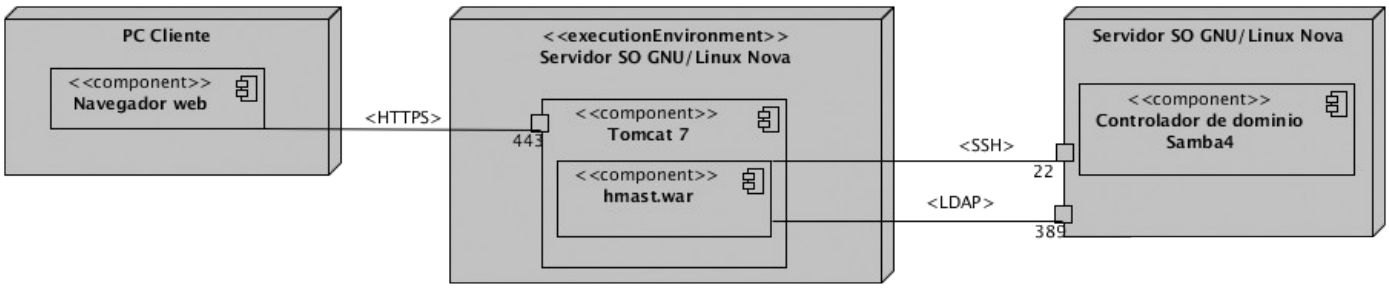


Figura 9. Diagrama de despliegue del módulo.

Conclusiones parciales

El empleo de los estándares de codificación definidos para la herramienta HMAST permitió desarrollar un código reutilizable, de fácil comprensión por los integrantes del equipo de desarrollo. La selección de una estrategia de prueba con un enfoque incremental permitió comprobar todas las partes del módulo y las relaciones entre estas. Mediante las pruebas realizadas se verificó que todas las instrucciones del módulo se ejecutan al menos una vez, que los componentes se integran correctamente y se validó que el módulo se ajusta de forma satisfactoria a los requisitos del sistema. La prueba realizada al proceso de migración mediante un experimento con el módulo confirmó el éxito durante la transferencia de los datos desde Windows Server hacia Samba4 garantizando a su vez, la integridad de los datos.

CONCLUSIONES GENERALES

A partir de la investigación realizada y de los resultados obtenidos referentes al desarrollo del módulo de HMAST para administrar y migrar hacia Samba4 el servicio Directorio Activo, se concluye que:

- La caracterización de las herramientas existentes para la administración del servicio Directorio Activo permitió concluir que estas no pueden ser integradas con HMAST, por ello se emplearon la herramienta *samba-tool* y la biblioteca *spring-ldap* para realizar el proceso de administración.
- La selección del mecanismo de migración que ofrece la herramienta *samba-tool* mediante la utilidad *domain join,* permitió transferir los datos manteniendo su integridad.
- A partir del diseño e implementación del módulo se obtuvo un sistema informático que permite migrar hacia Samba4 el servicio Directorio Activo y realizar una administración avanzada de los recursos de red.
- Las pruebas diseñadas y ejecutadas validaron el correcto funcionamiento del sistema así como el cumplimiento de todos los requisitos definidos. El experimento de prueba realizado demostró que el módulo garantiza la integridad de los datos durante el proceso de migración hacia Samba4.

RECOMENDACIONES

El presente trabajo permitió obtener un módulo de HMAST que garantiza la administración y la migración hacia Samba4 del servicio Directorio Activo. En aras de ampliar las funcionalidades de administración avanzada y otorgarle valor añadido al sistema, se recomienda:

- Implementar las funcionalidades que permitan
  - Delegar y revocar el control de tareas administrativas en el Directorio Activo.
  - Restaurar objetos eliminados del Directorio Activo.
  - Actualizar la versión de Samba4.
- Establecer conexiones LDAP seguras desde HMAST al controlador de dominio Samba4 utilizando para ello el protocolo SSL (capa de puertos seguros, del inglés *Secure Sockets Layer*).

Referencias bibliográficas

[1] Viera Hernández, Amaury y De La Rosa Gómez, Leonardo. MigrateAD: Migración del Directorio Activo a plataforma libre. 2009.

[2] Villazón Pérez, Yoandy. Metodología para la Migración a Software Libre de las Universidades del Ministerio de Educación Superior. Tesis de diploma. Universidad de las Ciencias Informáticas, Facultad 10, La Habana, Cuba, 2008.

[3] Microsoft Corporation. Windows Protocols Master Glossary. 2014.

[4] Arruebo Sosa, Pedro. Desarrollo de herramienta de integración con Directorio Activo para Nova. Tesis de diploma. Universidad de las Ciencias Informáticas, La Habana. 2013.

[5] Universidad de las Ciencias Informáticas. Misión | Portal de la Universidad de las Ciencias Informáticas. <http://www.uci.cu/?q=mision> [Accedido 5 de mayo de 2016].

[6] Pérez Villazón, Yoandy. Guía cubana de migración a aplicaciones de código abierto. 2009.

[7] DESMOND, Brian et al Active Directory 5th Edition

[8] Carter, Gerald. LDAP system administration. O'Reilly, 2003.

[9] Microsoft Corporation. Domain Name Service (DNS) Server Management Protocol. 2014.

[10] Samba. Time syncronisation. < https://wiki.samba.org/index.php/Time_syncronisation > [Accedido 3 de febrero de 2016].

[11] Windows Corporation. Active Directory Technical Specification. 2014.

[12] Samba. Samba - opening windows to a wider world. < https://www.samba.org/samba/ > [Accedido 3 de febrero de 2016]

[13] Samba. What is Samba? < https://www.samba.org/samba/what_is_samba.html > [Accedido 3 de febrero de 2016]

[14] García Chico, Joanna. Estudio de viabilidad de Directorio Activo en Linux. Proyecto de fin de carrera. Universidad Carlos III de Madrid, febrero, 2011.

[15] Computerworld. Active Directory comes to Linux with Samba 4. <http://www.computerworld.com.au/article/273515/active_directory_comes_linux_samba_4/> [Accedido 3 de febrero de 2016].

[16] Samba. Join an additional Samba DC to an existing Active Directory - SambaWiki. <https://wiki.samba.org/index.php/Join_an_additional_Samba_DC_to_an_existing_Active_Directory> [Accedido 5 de mayo de 2016].

[17] phpLDAPadmin. About - phpLDAPadmin. <http://phpldapadmin.sourceforge.net/wiki/index.php/About> [Accedido 9 de febrero de 2016].

[18] JXplorer. JXplorer - an open source LDAP browser. < http://jxplorer.org/ > [Accedido 9 de febrero de 2016].

[19] Sourceforge. LDAP Administration Tool. <https://sourceforge.net/p/ldap-at/wiki/Home/ > [Accedido 3 de mayo de 2016].

[20] Ldapadmin. LDAP Admin - a free LDAP directory browser and editor. < http://www.ldapadmin.org/ > [Accedido 9 de febrero de 2016].

[21] Samba. samba-tool. < https://www.samba.org/samba/docs/man/manpages/samba-tool.8.html > [Accedido 3 de mayo de 2016].

[22] Castillo Arbelo, Reidiel y Acosta Soria, Pablo. Herramienta para la Migración y Administración de Servidores (HMAS) [En línea]. La Habana, Universidad de las Ciencias Informáticas, 2012. <http://bibliodoc.uci.cu/RDigitales/2012/noviembre/7/TD_05109_12.pdf> [Accedido 10 de marzo de 2016]

[23] Laboratorio Nacional de Calidad del Software de INTECO. Ingeniería del Software: Metodologías y ciclos de vida. España, marzo, 2009.

[24] Universidad de las Ciencias Informáticas. Metodología de desarrollo para la Actividad productiva de la UCI. 2015.

[25] Bonata, Maximiliano. Programación y Algoritmos. octubre, 2006.

[26] Savit, Jeff; Wilcox, Sean; Jayaraman, Bhuvana. JAVA para la empresa. 2000.

[27] Cwalina, Krysztof y Abrams, Brad. Framework Design Guidelines. Second Edition.octubre, 2008.

[28] Williams S., Nick. Professional Java for Web Applications. 2014., ,

[29] Varanasi, Balaji. Practical Spring LDAP. octubre, 2013.

[30] Terry, B.y Logee, D. Terminology for Software Engineering and Computer-Aided Software Engineering. Software Engineering Notes, abril, 1990.

[31] Pilone, Dan y Miles, Russ. Head First Software Development. O'Reilly, 26 de marzo de 2008.

[32] Ecured. Requisitos de Software. < http://www.ecured.cu/Requisitos_de_Software > [Accedido 9 de febrero de 2016].

[33] IEEE-STD-830-1998: PRÁCTICA RECOMENDADA PARA LAS ESPECIFICACIONES DE REQUISITOS DEL SOFTWARE. 2008

[34] Palma Pérez, Nurisel. Módulo para la administración de los servidores web en HMAST. Universidad de las Ciencias Informáticas, 2013.

[35] Zorrilla Castro, César. Guía de Arquitectura N-Capas orientada al Dominio con .NET 4.0. . 2010. ISBN 978-84-936696-3-8.

[36] Gamma, Eric; Helm, Richard; Johnson, Ralph; Vlissides, John. Design Patterns - Elements of Reusable Object-Oriented Software.

[37] Larman, Craig. UML y patrones, 2da Edición.

[38] MSDN. Técnicas de codificación. <http://msdn.microsoft.com/es- es/library/aa291593(v=vs.71).aspx > [Accedido 5 de marzo de 2016].

[39] IEEE. Swebok - Guide to the Software engineering body of knowledge. 2004. ISBN 0-7695-2330-7.

[40] S. Pressman, Roger. Ingeniería de software. Un enfoque práctico. Quinta edición.

[41] De la Puente, Juan Antonio. Fiabilidad y tolerancia a fallos. DIT/UPM. 2001.

[42] Ecured. Diagrama de despliegue. < http://www.ecured.cu/Diagrama_de_despliegue > [Accedido 21 de marzo de 2016].

GLOSARIO DE TÉRMINOS

ACL: Lista de Control de Acceso, del inglés *Access Control List.* Es un concepto de seguridad informática usado para fomentar la separación de privilegios. Su objetivo es filtrar el tráfico, permitiendo o denegando el tráfico de red de acuerdo a alguna condición.

Apparmor: del inglés *Application Armor*. Es un programa de seguridad para Linux, lanzado bajo la licencia GPL que permite al administrador del sistema asociar a cada programa un perfil de seguridad que restrinja las capacidades de ese programa.

Complejidad ciclomática: es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa. Es una de las métricas de software de mayor aceptación, puesto que es independiente del lenguaje.

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

DLZ: *Dynamically Loadable Zones* es un parche para la versión 9 del servidor DNS Bind, que simplifica la administración, reduce el uso de memoria y el tiempo de arranque. DLZ permite almacenar la zona DNS en una base de datos, por ejemplo OpenLDAP y LDB, al igual que Microsoft Active.

IDE: Entorno de Desarrollo Integrado, del inglés *Integrated Development Environment.* Consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica.

LDIF: Formato de intercambio de datos, del inglés *LDAP Data Interchange Format*, se usa para representar entradas de LDAP en formato de texto sencillo.

Licencia GPL: Licencia Pública General de GNU que garantiza la libertad de compartir y modificar software libre, para asegurar que el software es libre para todos sus usuarios. Esta Licencia Pública General se aplica a la mayor parte del software de la Free Software Foundation y a cualquier otro programa si sus autores se comprometen a utilizarla.

OSI: Sistema de Interconexión Abierto, del inglés *Open System Interconnection,* es un modelo de interconexión de sistemas abierto creado por la Organización Internacional para la Estandarización y es marco de referencia para la definición de arquitecturas de interconexión de sistemas de comunicaciones.

Protocolo: Conjunto de normas que rigen un determinado proceso de comunicación.

Servidor: Es un proveedor de recursos, desempeñando un papel pasivo en las comunicaciones. Tras la recepción de una solicitud son los encargados de procesarla y enviar una respuesta al cliente. Puede por

lo general aceptar conexiones de un gran número de clientes, y por lo general no interactúan con los usuarios finales.

SSH: Intérprete de órdenes segur, del inglés *Secure SHell*. Es el nombre de un protocolo y del programa que lo implementa. Se emplea para acceder a máquinas remotas a través de una red. Permite administrar la computadora mediante un intérprete de comandos.