Instituto Superior Politécnico
"Jos Antonio Echeverría"
cujae

FACULTAD DE INGENIERÍA ELÉCTRICA

DEPARTAMENTO DE TELECOMUNICACIONES Y TELEMÁTICA

TESIS PRESENTADA EN OPCIÓN AL GRADO DE MÁSTER EN TELECOMUNICACIONES Y TELEMÁTICA

Diseño de un marco de trabajo para el desarrollo de software orientado a la gestión de servicios telemáticos de infraestructura en GNU/Linux

Autor: Ing. Ramón Alexander Anglada Martínez

Tutor: Dr. C. Tec Alain Abel Garófalo Hernández.

Ciudad de la Habana, 2013

Agradecimientos

Mis mayores agradecimientos para mi abuela Guarina, que ya no está con nosotros, mi motor impulsor y guía eterna. Le agradezco a mi madre Iliana por todo su amor, cariño, ternura y comprensión durante toda la vida, a mi hermano, el mejor de mis amigos, una persona admirable y de un corazón y coraje enormes, ejemplo para todos. Agradezco a mi mujer y a nuestro hijo que lleva en su vientre, agradezco todo el amor, cariño, ternura, apoyo que me has dado y toda la confianza depositada.

Quiero agradecer al tutor de esta tesis, al Dr. Alain, por su tiempo, su dedicación, por su tutoría excepcional a este trabajo.

Un agradecimiento especial al equipo de Servicios Telemáticos de la facultad 2 de la UCI, este resultado también es de ustedes.

Agradezco a todas mis amistades, tanto de Santiago de Cuba, como de la UCI, los cuales también han contribuido a este logro, no menciono a ninguno para evitar que se me quede alguno.

Agradezco a los profesores del diplomado de Telemática impartido en la UCI, al Centro de Telemática de la UCI y a todas las personas que de una forma u otra aportaron a esta tesis.

Dedicatoria

A la memoria de mi abuela Guarina y a mi hijo que está por nacer.

Síntesis

Los servicios telemáticos de infraestructura son importantes para el buen funcionamiento de las redes de datos actuales. Para ofrecer estos servicios se utiliza ampliamente el sistema operativo GNU/Linux, sin embargo las herramientas empleadas para su administración poseen deficiencias en la gestión de los parámetros de configuración, la integración entre los mismos y ofrecen pocas opciones para realizar una migración, de las implementaciones propietarias, hacia estos servicios.

La presente investigación realiza un estudio de los servicios telemáticos de infraestructura en GNU/Linux, sus posibles integraciones, principales herramientas utilizadas para su administración y las debilidades de estas, así como las bibliotecas de clases y componentes reutilizables que facilitan el desarrollo de herramientas para la gestión de los servicios de infraestructura en GNU/Linux.

La tesis describe el diseño de un marco de trabajo que propicia el desarrollo de herramientas orientadas a la gestión e integración de servicios telemáticos de infraestructura, reutilizando las bibliotecas de clases existentes especializadas en parte de la gestión de un determinado servicio de infraestructura. Se muestra también una herramienta implementada sobre este framework que facilita la gestión e integración de los servicios telemáticos de infraestructura en GNU/Linux disminuyendo las deficiencias de las herramientas actuales.

Tabla de contenido

INTRODUCCIÓN

En la actualidad las Tecnologías de la Información y las Comunicaciones (TIC) juegan un rol fundamental en el desarrollo, evolución y eficiencia de cualquier empresa, institución y país de manera general. Una de las ventajas y facilidades de las TIC lo constituyen los servicios telemáticos. Un servicio telemático es una entidad geográficamente distribuida, que provee a un número de personas, un conjunto de facilidades para cubrir un rango de necesidades de información y comunicación, utilizando los recursos existentes de las redes de telecomunicaciones [1].

Los servicios telemáticos funcionan de manera similar a cualquier servicio en el mundo y poseen lógicamente dos partes bien identificadas, un servidor que ofrece un servicio y un consumidor*,* cliente o usuario de este servicio. Tomando como ejemplo el servicio telemático correo electrónico, para este existe un servidor que ofrece un mecanismo para establecer comunicación, entre usuarios físicamente aislados, como vía alternativa y más rápida al correo tradicional. En los servicios telemáticos el servidor es un programa o software y los usuarios finales utilizan un software cliente para acceder a los beneficios del servidor. Estos servicios pueden ser ofrecidos por una empresa, centro de estudio, institución, país, organización, entre otros y el usuario puede consumir múltiples servicios telemáticos de diferentes proveedores.

Dentro de los servicios telemáticos existe un grupo de servicios de infraestructura, esenciales para el buen funcionamiento de la red, los cuales no son consumidos por el usuario final de manera directa, son servicios utilizados principalmente por otras aplicaciones o servicios las cuales dependen de estos para su funcionamiento. Uno de estos servicios es el Sistema de Nombres de Dominio (DNS, Domain Name System) [2]. Uno de los sistemas operativos ampliamente utilizados con el objetivo de proveer

servicios de infraestructura es GNU/Linux[1], sistema operativo software libre y de código abierto el cual ofrece muchas ventajas. En GNU/Linux, la mayoría de los servicios de infraestructura, poseen implementaciones igualmente de código abierto y software libre de calidad muy utilizadas. Para la administración de estos servicios en GNU/Linux el administrador de red utiliza regularmente herramientas informáticas las cuales ofrecen diversas funcionalidades, sin embargo, las mismas no son eficaces en la gestión de todos los parámetros de configuración del servidor, la integración de los servicios y la migración de los servicios alojados en plataformas propietarias hacia soluciones software libre de GNU/Linux.

Las configuraciones de los servidores en GNU/Linux están en ficheros de texto y la sintaxis es heterogénea. Cada servicio puede tener varios archivos de configuración en el servidor y dentro de cada uno definir disímiles parámetros con un alto grado de relación. La mayoría de las herramientas de gestión de servicios de infraestructura manipulan solo una parte de los parámetros de configuración del servidor, generalmente los esenciales para poner en marcha el servicio. El resto de los parámetros, los cuales también son esenciales e importantes, en dependencia de las necesidades del administrador de red y su nivel de conocimiento, no son manipulados por la herramienta o simplemente la herramienta muestra directamente el contenido del archivo de configuración al administrador para que este realice los cambios necesarios, lo cual puede provocar errores y fallas en el servicio afectando su disponibilidad y a los usuarios finales.

Estas mismas herramientas además ofrecen escasas posibilidades para realizar todas las integraciones posibles de los servicios, en la mayoría de los casos el administrador de red tiene que acudir a una tercera herramienta que si brinde la posibilidad de integración que necesita implementar u otras variantes como la ejecución de comandos en consola y la edición de varios archivos de configuración afectando nuevamente la disponibilidad del

[1]GNU\Linux: Sistema operativo software libre.

servicio y haciendo más compleja su administración. También constituye una insuficiencia de estas herramientas de gestión de servicios de infraestructura las pocas opciones para importar datos y configuraciones de los mismos servicios alojados en otros sistemas operativos lo que dificulta que se puedan trasladar los mismos a GNU/Linux, dejando de apoyar al proceso de migración a software libre que llevan a cabo muchas organizaciones y países de manera general.

Las herramientas informáticas de gestión de servicios de infraestructura son software cada vez más complejos, lo que impone la necesidad de una solución que permita reusar el diseño y el código fuente de las herramientas y componentes existentes, aumentar la productividad de los desarrolladores en la generación de código y la flexibilidad para cambiar los requerimientos del software [3] [4] [5]. Para desarrollar software y lograr reusabilidad de código fuente, productividad y flexibilidad se utilizan marcos de trabajo o frameworks. Un marco de trabajo es una aplicación reusable y semicompleta que puede ser especializada para producir aplicaciones personalizadas o específicas [5] [6]. Con un framework se logra un alto grado de estandarización, homogenización y reutilización del código fuente, organización del esfuerzo de todos los miembros del proyecto y rapidez en el desarrollo del software.

Existen proyectos, componentes y frameworks que gestionan en la actualidad partes de la gestión de servicios de infraestructura en GNU\Linux, como por ejemplo el framework twisted. Twisted es un framework de red de código abierto multiplataforma implementado en Python, el mismo soporta tanto cliente como servidor. Twisted soporta los protocolos HTTP[2], FTP[3], DNS[4], SMTP[5], SSH[6], entre otros [7]. Twisted está orientado al desarrollo de protocolos personalizados y provee implementaciones de protocolos comunes como los mencionados anteriormente pero no está orientado a la gestión del servidor de este

[2]HTTP: Hypertext Transfer Protocol
[3]FTP: File Transfer Protocol
[4] DNS: Domain Name System
[5]SMTP: Simple Mail Transfer Protocol
[6]SSH: Secure SHell

protocolo. Un framework similar a Twisted, desarrollado en el lenguaje de programación C++, lo constituye ACE [7] [8].

Otro proyecto que contiene un API[7], desarrollado en el lenguaje de programación perl, para el desarrollo de este tipo de aplicaciones es Zentyal, disponible en http://www.zentyal.com/, anteriormente denominado proyecto e-box. Zentyal es una solución que gestiona los parámetros de configuración comunes de los servicios, no realiza todas las integraciones posibles entre los servicios y tiene una versión comercial que incluye mejoras que no están en la versión de código abierto. JNDI constituye otra API escrita en el lenguaje de programación Java que provee servicios de directorio y nombrado, provee métodos para ejecutar operaciones estándar de directorio, tales como la asociación de atributos con objetos. Con JDNI se puede definir cualquier especificación de servicio de directorio o de nombre [9], sin embargo JDNI solo se enmarca en el servicio de nombre y el de directorio y no provee métodos para gestionar los otros servicios.

En la mayoría de los lenguajes de programación existen bibliotecas de clases que gestionan servicios de infraestructura directamente, pero son bibliotecas de clases independientes que solo abarcan una pequeña parte de la gestión del servidor.

Por todo lo anteriormente planteado se define como **problema a resolver**: Las insuficiencias en las herramientas para la gestión de configuración de los servicios telemáticos de infraestructura en GNU/Linux lo cual dificulta la administración e integración de los mismos.

El **objeto de estudio** son los servicios telemáticos de infraestructura y el **campo de acción** la gestión de la configuración de los servicios telemáticos de infraestructura en GNU/Linux.

Se plantea como **hipótesis**: Con el diseño de un marco de trabajo orientado a componentes, reutilizando las bibliotecas de clases especializadas en partes de la gestión

[7] API: Application Programming Interface

de los servicios telemáticos de infraestructura, se facilita y propicia el desarrollo de herramientas para la gestión de los servicios telemáticos de infraestructura en GNU/Linux y la integración de los mismos.

El **objetivo** del trabajo es diseñar un marco de trabajo orientado a componentes para el desarrollo de herramientas de gestión e integración de servicios telemáticos de infraestructura en GNU/Linux.

A continuación las tareas de la investigación:

1. Estudio de los servicios de infraestructura y sus características fundamentales así como las herramientas informáticas actuales para la gestión de configuración e integración de los mismos.
2. Estudio de los componentes, APIs, marcos de trabajo y bibliotecas de clases relacionados con la gestión de configuración e integración de servicios de infraestructura.
3. Estudio de los patrones de arquitectura y diseño de software utilizados para el desarrollo de marcos de trabajo.
4. Seleccionar los patrones de arquitectura y diseño de software que se utilizarán para el diseño del marco de trabajo.
5. Definición de la estructura y organización del marco de trabajo para la gestión de configuración e integración de servicios de infraestructura.
6. Diseño del marco de trabajo para la gestión de configuración e integración de servicios de infraestructura.
7. Validación del diseño del marco de trabajo con el desarrollo de una herramienta para la gestión de configuración de servicios de infraestructura.

Para la realización de la investigación se utilizaron los siguientes métodos científicos:

Métodos teóricos

Análisis y síntesis: Este método permitió realizar una división lógica del problema a

resolver y sus relaciones facilitando su estudio y comprensión durante toda la investigación.

Histórico y lógico: Este método fue fundamental en la realización del estado del arte y la fundamentación teórica de la investigación, el método permitió el estudio de los marcos de trabajo, bibliotecas de clases, componentes y herramientas relacionadas con la gestión de configuración e integración de los servicios de infraestructura.

Modelación: La modelación constituyó uno de los métodos principales en la investigación, por la representación realizada de todo el diseño del marco de trabajo, sus propiedades, relaciones de sus componentes y partes fundamentales.

Métodos empíricos

Observación: La observación fue otro de los métodos utilizados en el desarrollo de la investigación, específicamente en la observación de las pruebas realizadas al marco de trabajo y sus componentes para el desarrollo de software para la gestión de configuración de servicios de infraestructura.

Experimentación: La experimentación se utilizó en la validación del diseño realizado del marco de trabajo para comprobar que el diseño del marco cumplió con todos los objetivos trazados.

El presente trabajo realiza los siguientes aportes:

- Diseño e implementación de un marco de trabajo o framework, modular y extensible, para la gestión de configuración e integración de servicios de infraestructura contribuyendo al desarrollo de software orientado a la gestión e integración de este tipo de servicio.
- Definición de una arquitectura de software basada, en patrones, para el desarrollo de software orientado a la gestión e integración de servicios, aumentando la productividad y eficacia de este tipo de software.
- Definición de una manera estándar para acceder a las configuraciones de los

servicios de infraestructura y realizar una correcta manipulación del mismo.

- Agrupación de varias bibliotecas de clases de código abierto definiendo un marco de trabajo especializado en la gestión de configuración de servicios telemáticos de infraestructura reutilizando código fuente y esfuerzo.

La tesis consta de 3 capítulos, Conclusiones, Recomendaciones y Bibliografía. Los capítulos están divididos de la siguiente forma:

Capítulo 1: En el capítulo se realiza un estudio de los servicios telemáticos de infraestructura en GNU/Linux, sus principales características y vías de integración. Se realiza un estudio del estado del arte sobre los marcos de trabajo, APIs, bibliotecas de clases y soluciones existentes para la gestión de configuración de servicios de infraestructura en el sistema operativo GNU/Linux así como sus deficiencias.

Capítulo 2: Se presenta el diseño del marco de trabajo orientado al desarrollo de herramientas para la gestión de configuración de servicios de infraestructura y la integración de los mismos, su arquitectura y componentes así como los patrones de diseño utilizados.

Capítulo 3: En el capítulo se muestra una herramienta informática desarrollada a partir del diseño del marco de trabajo objetivo de esta tesis. Se describen las tecnologías utilizadas, su arquitectura de software y las características fundamentales de la herramienta.

CAPÍTULO 1. GESTIÓN E INTEGRACIÓN DE SERVICIOS TELEMÁTICOS DE INFRAESTRUCTURA

1.1 Introducción

Los servicios telemáticos influyen cada vez más en el desarrollo tecnológico de las empresas y constituyen una parte importante de la misma. Dentro de los servicios telemáticos se encuentran los servicios de infraestructura, los cuales son importantes para el buen funcionamiento del resto de los servicios. La administración de los servicios de infraestructura es una tarea compleja pues el resto de los servicios en su mayoría depende de estos [2]. Las tareas de gestión de los servicios de infraestructura se hacen más difíciles si los servidores están alojados en sistemas operativos GNU/Linux pues las herramientas utilizadas ofrecen funcionalidades importantes pero no son eficaces en determinadas tareas, tales como la gestión y manipulación de todos los parámetros de configuración del servidor, la integración de los servicios y la migración de estos hacia soluciones software libre.

En el presente capítulo se realiza un estudio del estado del arte de las herramientas de gestión de configuración para los servicios de infraestructura en GNU/Linux y las deficiencias que presentan así como las bibliotecas de clases, APIs y marcos de trabajo para el desarrollo de software orientado a la gestión e integración de servicios de infraestructura.

1.2 Servicios de Infraestructura

Los servicios de infraestructura son servicios mayormente utilizados por el resto de los servicios telemáticos y aplicaciones de la red y no directamente por el usuario. Dentro de los servicios de infraestructura más utilizados se pueden mencionar el Sistema de Nombres de Dominio (DNS) [2], el Protocolo de Configuración Dinámica de Host (DHCP)

y el Servicio de Directorio basado en el LDAP (Lightweight Directory Access Protocol). En GNU/Linux los servicios de infraestructura poseen implementaciones software libre ampliamente utilizadas los cuales brindan al administrador disímiles ventajas tales como la reducción de los costos por concepto de adquisición de software, acceso al código fuente de la implementación, entre otras ventajas. OpenLDAP[8], Bind y DHCP de ISC [10] son implementaciones software libre y de código abierto, de los servicios Directorio, DNS y DHCP respectivamente, muy utilizadas en GNU/Linux.

OpenLdap es un proyecto de código abierto que implementa el protocolo LDAP, es un sistema flexible y escalable, optimizado para las operaciones de lectura. Permite establecer criterios de búsqueda complejos y realizar réplicas de la información entre servidores LDAP. OpenLDAP funciona en la mayoría de las distribuciones de GNU/Linux, Mac OS X y la mayoría de las versiones de Windows, entre otros. OpenLDAP soporta todas las versiones del protocolo LDAP, SSL[9], IPv6, LDIF[10], DSML[11], representa su información de manera jerárquica en un árbol de directorio y permite la gestión de los esquemas del directorio [11], a continuación una imagen que ilustra esta jerarquía:

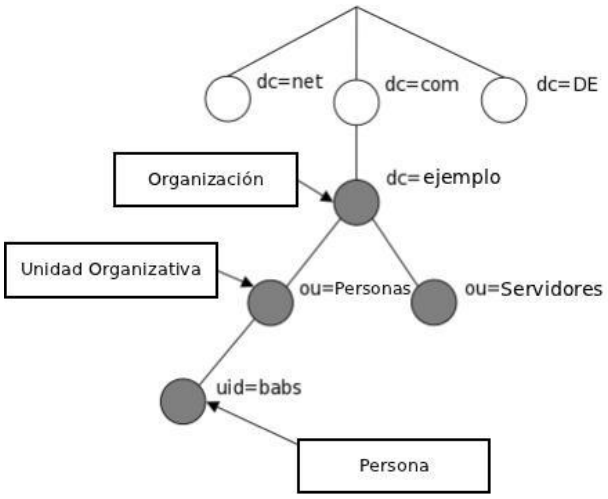


Figura 1. Ejemplo de Árbol de Directorio LDAP tomada de [11].

OpenLdap es una de las implementaciones software libre que implementa el protocolo

[8] OpenLDAP: Implementación de código abierto del protocolo LDAP.
[9] SSL: Security Socket Layer
[10] LDIF: LDAP Data Interchange Format
[11] DSML: Directory Services Markup Language

LDAP más utilizadas y con varias posibilidades de integración para el resto de los servicios de infraestructura. El servicio de Directorio puede ser utilizado como punto de autenticación, almacén de información de los servicios y de configuraciones específicas de otras aplicaciones [11], entre otras funcionalidades.

El servicio DNS es uno de los servicios de infraestructura más importantes y es utilizado prácticamente por el resto de los servicios con el objetivo de resolver direcciones IP [2]. Bind es la implementación más usada de servicio DNS en Internet, es una implementación de código abierto del protocolo DNS que provee y mantiene ISC. Bind está compuesto por tres partes fundamentales: el servidor DNS encargado de proveer el servicio DNS, el cual se instala como un proceso o demonio en GNU/Linux, una librería reusable que provee todo el mecanismo para la resolución de nombres y un conjunto de herramientas para comprobar el correcto el funcionamiento de Bind. Actualmente la versión estable mantenida y distribuida por ISC es Bind9 pero ya se encuentra en desarrollo una versión 10 que incluyen un conjunto de mejoras respecto a la versión actual [10].

ISC provee además de Bind, una implementación de código abierto de DHCP, actualmente en su versión 4.2. El DHCP de ISC es uno de los más utilizados en Internet, el mismo está compuesto por el servidor DHCP que recibe y responde a las peticiones de los clientes DHCP, un cliente DHCP para realizar peticiones DHCP y un DHCP *relay* para enviar peticiones DHCP de una red LAN[12] a otra [10].

OpenLdap, Bind y DHCP de ISC constituyen las implementaciones software libre de servicios de infraestructura más utilizadas para los protocolos LDAP, DNS y DHCP en GNU/Linux respectivamente. La mayoría de las herramientas de gestión, desarrolladas y mantenidas por la comunidad de software libre para estos servicios, tienen deficiencias relacionadas con la manipulación de todos los parámetros de configuración del servidor, la

[12] LAN: Local Area Network

integración de los servicios y las vías para importar los datos del mismo servicio alojado en otro sistema operativo. En el siguiente epígrafe se profundiza en estas deficiencias tomando como referencia estas tres implementaciones de servicios de infraestructura.

1.3 Gestión de Servicios de Infraestructura en GNU/Linux

GNU/Linux es uno de los sistemas operativos ampliamente utilizados para alojar el software servidor de los servicios de infraestructura. Para administrar correctamente estos servicios y obtener un resultado deseado es fundamental tener en cuenta todos los parámetros de configuración del servidor, las características del sistema operativo sobre el cual se ejecuta, como importar y exportar las configuraciones del servidor así como las vías de integración con otros servicios. Una de las principales deficiencias de las herramientas actuales es que no manipulan todos los parámetros contenidos en los archivos de configuración del servidor.

1.3.1 Parámetros de configuración

Las tareas de administración de los servicios de infraestructura se realizan en el software servidor. Uno de los componentes esenciales para la administración del servidor lo constituyen sus parámetros de configuración y aquí radica una de las principales ineficiencias de las herramientas actuales para la administración de los servicios de infraestructura en GNU/Linux. Los parámetros de configuración de los servicios de infraestructura en GNU/Linux están en archivos de texto y no existe un estándar en la mayoría para definir su contenido. Cada implementación de servicio de infraestructura define sus archivos de configuración y la sintaxis es distinta en la mayoría de ellos. Además en varios de estos archivos de configuración se define información que durante la ejecución del servicio puede ser actualizada y a la vez consultada por las herramientas de administración.

Uno de los servicios de infraestructura que ha definido estándares con el objetivo de

homogenizar sus configuraciones y la información que almacena es el servicio de directorio. OpenLDAP tiene todas sus configuraciones en el Formato de Intercambio de Datos del protocolo LDAP(LDIF, LDAP Data Interchange Format), el cual está definido en la RFC 2849, dicho formato surgió con el objetivo de describir la información contenida en el servicio de directorio y es usado típicamente para importar y exportar información del servicio de directorio [12]. Existe otra iniciativa de estándar para el servicio de directorio, DSML (Directory Services Markup Language), cuyo objetivo es definir una especificación basada en XML[13] para representar la información del servicio de directorio [13].

Para la administración de OpenLdap se han desarrollado herramientas software libre y de código abierto, tales como Webmin[14], Phpldapadmin[15] , entre otras, pero no manipulan todos los parámetros de configuración, se concentran solamente en los que consideran esenciales, un ejemplo de parámetros que no son manipulados por las herramientas son los esquemas del servicio de directorio. Los esquemas representan toda la información que puede estar contenida en el OpenLDAP a través de un conjunto de definiciones de formularios de nombre, reglas de estructura y de contenido del árbol de directorio, reglas de coincidencia, sintaxis, clases de objetos y tipos de atributos [14]. Webmin, Phpldapadmin y la mayoría de las herramientas no ofrecen la posibilidad de manipular los esquemas del OpenLdap cuya su gestión facilita la personalización de la información que puede estar contenida en el servicio de directorio.

El servicio de directorio se ha convertido en un lugar apropiado para almacenar la información heterogénea proveniente de las fuentes de datos de las aplicaciones y servicios [15] [16] y la gestión de los esquemas del directorio es un paso importante si se tiene como propósito almacenar cualquier tipo de información en el LDAP. La mayoría de las herramientas para administrar OpenLdap no gestionan los esquemas por lo complejo

[13] XML: eXtensible Markup Languaje
[14] Webmin: Herramienta para la gestión de servicios en GNU/Linux
[15] Phpldapmin: Herramienta software libre para la gestión de OpenLdap.

que resulta su tratamiento disminuyendo las ventajas que proporciona OpenLdap, y es a través de la gestión de esquemas, que se pueden realizar varias de las integraciones de otros servicios a OpenLdap.

En el caso de los servicios DNS y DHCP no está definido ningún estándar para escribir las configuraciones aunque el DNS y DHCP de ISC en las configuraciones poseen sintaxis similares y son semejantes en la definición de la mayoría de los parámetros de configuración. Los servidores DNS y DHCP de ISC son menos complicados en cuanto a configuración comparados con OpenLdap pero igualmente las herramientas actuales no abarcan o gestionan todos sus parámetros de configuración. En el servidor DHCP de ISC la deficiencia en cuanto configuración de las herramientas radica en que no permiten todas las combinaciones de declaraciones posibles, regularmente la configuración radica en definir las subredes, con sus rangos de IP correspondientes, para las cuales el servidor DHCP ofrecerá sus servicios, pero puede ser necesidad del administrador realizar una configuración más avanzada, definiendo grupos o redes compartidas y dentro de estos definir las declaraciones necesarias. La mayoría de las herramientas no ofrecen la posibilidad de realizar todas las combinaciones posibles en las configuraciones de declaraciones del DHCP de ISC.

Otras de las configuraciones que no son tratadas por la mayoría de las herramientas para gestionar el DNS y DHCP de ISC son las relacionadas con la integración entre ellos y con otros servicios. La integración de un servicio con otro en la mayoría de los casos implica modificación en sus archivos de configuración, por ejemplo en la integración entre DNS y DHCP para la actualización dinámica del DNS, es necesario modificar los archivos de configuración del DNS y del DHCP, si la herramienta no manipula los parámetros correspondientes a la integración es imposible mediante la herramienta realizar la configuración deseada y el administrador se ve obligado a modificar directamente los ficheros de configuración del servidor.

De manera general las configuraciones de los servicios no son homogéneas y la herramienta de gestión de los servicios de infraestructura debe ser capaz de manipular todas las configuraciones de estos archivos y adaptarse a los posibles cambios de ubicación de estos así como la sintaxis definida en los mismos, de lo contrario la herramienta de gestión o administración deja de ofrecer una posibilidad de configuración del servicio que puede ser esencial en un determinado momento y obliga al administrador de red a utilizar una tercera herramienta que gestione este parámetro o simplemente ir directo al archivo de configuración lo cual puede provocar errores, fallas en el funcionamiento del servicio afectando su disponibilidad y a los usuarios finales y si del servicio dependen otros servicios, como el caso del DNS, entonces se afectarían también el resto de los servicios.

1.3.2 Integración de los servicios de infraestructura

Los servicios de infraestructura ofrecen beneficios para el buen funcionamiento de la red. A estos servicios pueden integrarse el resto con el propósito de cubrir necesidades de la organización. Estas integraciones en buena medida aumentan el grado de complejidad y dificultad para la administración de los servicios requiriendo personal más calificado, sin embargo las herramientas pueden jugar un papel fundamental en simplificar esta complejidad en cuanto a gestión de los servicios. Desafortunadamente las herramientas actuales no abarcan la mayoría de las integraciones posibles que se pueden definir con los servicios de infraestructura. A continuación se describen varias integraciones de servicios de infraestructura y las ventajas que ofrecen.

Integración al Servicio de Directorio

El servicio de directorio posee características que lo convierten en un servicio idóneo para el respaldo y almacén de información del resto de los servicios. El servicio de directorio posee una base de datos optimizada para el acceso rápido a la información siendo más ágil en las operaciones de lectura del árbol de directorio que contiene [11] [15] [16]. En su

base de datos el servicio de directorio almacena información de los usuarios de la organización, las máquinas de la red, entre otros datos. Uno de los mayores usos que se le da al servicio de directorio, en cuanto a integración con el resto, es el uso de este como punto de autenticación a través del protocolo LDAP para comprobar las credenciales de acceso de un usuario a un determinado servicio sin embargo existen otras vías de integración poco explotadas.

En el servicio de directorio es posible guardar información del resto de los servicios con el propósito de utilizar al servicio de directorio como base de datos para el resguardo y acceso rápido a la información. Los registros de una zona del DNS Bind es posible llevarlos a la base de datos del servicio de directorio lo cual puede ser útil para escenarios donde el servidor DNS posee un elevado número de registros y es más factible tener los registros en el servidor LDAP y no en ficheros de texto, como pudiera ser el caso de una solución para un proyecto de implementación del protocolo ENUM definido en la RFC 3761, el cual utiliza como base de datos el DNS [17]. Otra ventaja que ofrece llevar los registros del servidor DNS a OpenLDAP es el respaldo de la información, el servidor LDAP en su base de datos ofrece resguardo para los servicios lo cual es provechoso en caso de cualquier tipo de fallas en el servidor que implique perdida de datos o reinstalación del servicio en cuestión.

En OpenLdap se hace posible almacenar y resguardar información de otros servicios a través de la gestión de los esquemas. Para que un servicio determinado utilice el servidor de directorio con este propósito se deben definir en OpenLdap los esquemas, específicamente las clases de objetos que representarán la información con sus tipos de atributos y en el servicio que desea integrarse a OpenLdap faltaría, a través de los mecanismos o APIs correspondientes, conectarse al OpenLdap y utilizar el esquema definido para integrarse con el servicio de directorio. De esta forma la mayoría de los servicios podrían definir esquemas en el servidor de directorio y beneficiarse de las

ventajas que ofrece OpenLdap para almacenar información.

Las herramientas actuales Webmin, Phpldapadmin, entre otras, solo muestran los esquemas del OpenLdap, principalmente las clases de objetos, de manera organizada y la manipulación que ofrecen es la edición directa de los archivos de configuración donde están definidos los esquemas, de ser necesario adicionar un esquema las herramientas ofrecen poco apoyo, en este caso el administrador debe conocer la estructura de los esquemas en el servidor, realizar las definiciones y asociaciones de clases de objetos correspondientes y varias tareas complejas que hacen el trabajo de la gestión de esquemas complicado y difícil para el administrador de red. Una manipulación errónea de los esquemas pudiera provocar la detención del servicio afectando su disponibilidad.

Integración del DNS y DHCP

El servicio DNS constituye uno de los servicios de infraestructura fundamentales para el funcionamiento de la red el cual permite la resolución de nombres de dominio a direcciones IP de cualquier máquina de la red [2]. En GNU/Linux la implementación más utilizada de DNS es Bind. El servidor Bind cuando inicia lleva toda la información a la memoria RAM del servidor por tanto un aumento del número de zonas y sus registros aumenta las necesidades de RAM. Bind al iniciar lee toda la información contenida en sus archivos de zonas lo cual implica tiempo para estar disponible como servicio y la información contenida en estos archivos si cambia se debe realizar un reinicio del servicio afectando su estabilidad y disponibilidad en un determinado momento [18].

Para disminuir estas de deficiencias una de las soluciones que se han planteado es llevar a otro tipo de almacenamiento los registros de zona del Bind tales como base de datos o servicios de directorio como se menciona en el epígrafe anterior. Una de estas soluciones es DLZ (Dynamically Loadable Zones) [18] la cual constituye una de las implementaciones actuales para integrar el DNS Bind con otros servicios como el de directorio. DLZ es una mejora realizada al Bind que permite al mismo almacenar sus registros en varios gestores de base de datos tales como Postgres, MySql, Oracle y LDAP, entre otras

funcionalidades.

El tipo de almacenamiento que se decida utilizar en el servidor Bind, ficheros de texto, base de datos o LDAP, debe estar en correspondencia con las necesidades de la organización y el resto de los servicios que lo utilicen. Para determinados escenarios llevar los registros del Bind a una base de datos o LDAP puede ser útil y contribuir a elevar la calidad del servicio. Los mecanismos de replicación de los gestores de bases de datos o del servicio de directorio puede ser uno de los beneficios que se adquieren al integrar Bind con uno de estos servicios. En una red donde ya se tiene un servicio centralizado de base de datos o directorio con un sistema eficaz de replicación, puede favorecer el sistema DNS en cuanto a seguridad, respaldo de la información y actualización del sistema DNS [19]. Independientemente de que Bind es utilizado mayormente con ficheros de texto en sus configuraciones, el mismo se adapta a escenarios donde la información de los registros puede estar alojada en lugares más idóneos convirtiéndose en un servicio flexible y escalable. Las herramientas utilizadas por los administradores de red en su mayoría no gestionan estas posibilidades que ofrece Bind y las configuraciones asociadas las realiza el administrador directamente lo cual puede provocar los errores mencionados anteriormente.

El servidor DHCP de ISC es la implementación de DHCP para GNU/Linux ampliamente utilizada en Internet. El DHCP de ISC ofrece al Bind la posibilidad de actualizar los registros de computadores clientes de la red ante un cambio de nombre o dirección IP. En GNU/Linux luego de instalar Bind y DHCP de ISC, estos no están integrados por defecto y hay que establecer la configuración correspondiente si se desea que Bind y el DHCP de ISC estén integrados. La mayoría de las herramientas de código abierto o software libre actuales para la integración de servicios no ofrecen la opción de realizar esta integración para la cual hay que ejecutar cambios en los archivos de configuración del DNS y del DHCP aumentando las posibilidades del administrador de red de cometer errores.

Para las integraciones mencionadas las herramientas actuales de administración de servicios, no brindan el apoyo suficiente. El administrador de red nuevamente tiene que realizar estas tareas de integración editando archivos de texto y ejecutando comandos en consola, haciendo su trabajo más difícil y aumentando las posibilidades de cometer errores que conlleven a fallas en el servicio. Es real que este tipo de integraciones no se realizan periódicamente pero en la actualidad en el momento de realizarla el administrador se ve ante una situación compleja, por el nivel de conocimiento que implica, situación que puede ser minimizada por una herramienta informática.

Realizar una integración rápida y sencilla como tarea de administración entre los diferentes servicios es una de las deficiencias que presentan las herramientas actuales de gestión de servicios de infraestructura en GNU/Linux, lo cual dificulta el trabajo del administrador de red e influye en la calidad de los servicios de infraestructura de manera general. Para realizar una integración determinada hay que modificar en la mayoría de los casos ficheros de configuración por tanto las herramientas deben hacer una eficaz manipulación de todos los parámetros de configuración de los servicios. La integración de los servicios de infraestructura ofrece beneficios que deben ser abarcados por las herramientas de gestión para que estas sean de mayor utilidad para los administradores de red.

Importación de datos alojados en servicios de licencia propietaria

La migración a software libre constituye una alternativa para todo país, organización, empresa o centro que decida reducir los costos por concepto de adquisición de licencias de software propietarias, acceso al código fuente del software sin restricciones, poder hacer modificaciones y redistribuirlas [20], entre otras. Los servicios de infraestructura no están ajenos a la migración a software libre. Aunque los servicios de infraestructura implementan protocolos que son estándares, las configuraciones de estos en su mayoría no están basadas en estándares y los sistemas propietarios no hacen públicas sus configuraciones con el objetivo de aliviar el proceso de migración, lo cual dificulta más aun

el proceso.

En dependencia del servicio de infraestructura es el nivel de complejidad de la migración, por ejemplo en el caso del servicio de directorio, a través del formato LDIF se puede importar y exportar toda la información del árbol de directorio logrando llevar la información de una plataforma a otra con facilidad pero no en todos los casos de servicio de directorio la migración terminaría con exportar e importar la información en formato LDIF, si la migración que se desea realizar es del Directorio Activo de una versión de Windows Server a OpenLdap el proceso es aún más complejo pues se necesita también migrar las políticas de grupo del Directorio Activo para lo cual todavía no se ha logrado una solución estable y se espera que Samba4 brinde el soporte necesario [21]. En el caso de otros servicios de infraestructura como DNS y DHCP el proceso de migración es menos complejo. En ambos casos no existe un estándar para exportar e importar la información del servicio pero la información de estos en plataformas propietarias puede ser consultada con el objetivo de importarla hacia soluciones de código abierto como el DNS y DHCP de ISC.

La migración de los servicios de infraestructura es una de las tareas complicadas para un administrador de red y es fundamental en esto el apoyo que puedan ofrecer las herramientas de gestión e integración de servicios. Las herramientas actuales de administración de servicios de infraestructura en dependencia del servicio ofrecen pocos mecanismos para apoyar o realizar este proceso de migración. Soluciones como Webmin, phpldapmin, entre otras, brindan la posibilidad de gestionar los parámetros básicos de servicios de infraestructura pero no ofrecen la posibilidad de importar las configuraciones de estos servicios alojados en otros sistemas operativos, por tanto el administrador de red que decide llevar sus servicios de plataformas propietarias a soluciones software libre para utilizar herramientas como Webmin, primero debe realizar configuraciones necesarias a través de comandos en consola, ediciones de múltiples ficheros de

configuración y en muchas ocasiones no se logra el objetivo y se pierde información del servicio en el traslado de una plataforma a otra.

1.4 Soluciones para la gestión de servicios

Las herramientas actuales de gestión de servicios de infraestructura de código abierto o software libre facilitan la administración de los diferentes servicios. En su mayoría estas herramientas poseen las limitaciones mencionadas en los epígrafes anteriores en cuanto a gestión de todos los parámetros de configuración, integración entre los servicios y apoyo en el proceso de importar datos y configuración del mismo servicio ejecutado en otra plataforma, a continuación una descripción de las fundamentales.

Webmin

Webmin es una aplicación web para la administración del sistema operativo GNU/Linux. A través de Webmin se puede administrar las cuentas de usuario del sistema, los servidores DNS, DHCP, LDAP, entre muchas otras funcionalidades. A Webmin se integra, Usermin, solución que permite la gestión de las contraseñas de usuarios y varias tareas de administración relacionadas con las cuentas de usuario en los servidores de correo [22].

Virtualmin es otra solución relacionada a Webmin, básicamente es uno de sus módulos que permite la ejecución y administración de máquinas virtuales donde alojar determinados servidores como Apache, Bind DNS, MySql, entre otros. Virtualmin posee una versión comercial con más funcionalidades, actualizaciones y corrección de errores que no se encuentran en la versión libre, lo cual implica un costo. Otra de las soluciones integradas a Webmin es CloudMin. CloudMin permite la gestión de sistemas virtuales tales como Xen, KVM, OpenVZ, entre otros, al igual que VirtualMin posee una versión comercial con mejoras, actualizaciones y correcciones no presentes en la versión de código abierto [22].

Webmin es una de las soluciones de código abierto útiles para administrar los servicios de infraestructura y realizar configuraciones necesarias en el servidor, sin embargo presenta

algunas desventajas como por ejemplo la no administración de varios servidores de manera simultánea, en una red con varios servidores DNS para su administración sería necesario la instalación de Webmin en el sistema operativo de cada uno de estos servidores incluyendo además el servidor web correspondiente. Webmin no ofrece la posibilidad de realizar las integraciones posibles entre los distintos servidores como es el caso de la integración entre los servidores DNS y DHCP obligando al administrador de red a utilizar otras vías para realizar esta integración. Webmin no permite importar datos de otras soluciones como por ejemplo de Windows Server tema importante en la migración a software libre.

Zentyal

Zentyal es otro proyecto desarrollado en el lenguaje de programación perl disponible en http://www.zentyal.com/, anteriormente denominado proyecto E-box. Zentyal ofrece una herramienta de gestión e integración de servicios de infraestructura y tiene un API para el desarrollo de los módulos del sistema. Zentyal en la gestión de los parámetros de configuración de los servicios y la integración de los mismos no abarca todos los parámetros y se concentra en los comunes. El resto de las configuraciones hay que hacerlas directamente interactuando en los archivos de configuración del servicio [23] lo cual puede provocar fallas del sistema e implica un mayor grado de conocimientos para el administrador de red. Zentyal tampoco permite importar configuración alojadas en servicios de infraestructuras que puedan estar ejecutándose en otras plataformas como Windows y posee una versión comercial con mejoras indiscutibles respecto a la versión libre de costo.

FreeIPA

FreeIPA constituye otra solución para la administración de servicios. Es una solución integrada de seguridad y administración que combina la distribución basada en GNU/Linux Fedora, Mit Kerberos, NTP, DNS, entre otros servicios. Consiste en una aplicación web y un conjunto de líneas de comandos de administración [24]. FreeIPA

posee un API implementada en el lenguaje python disponible en http://freeipa.org/developer-docs/. Para la administración de FreeIPA se debe utilizar la línea de comandos lo cual dificulta la administración de los servicios y a través de la interfaz web no es posible realizar todas las configuraciones. FreeIPA tampoco abarca la mayoría de las integraciones que se puede establecer entre los servicios.

Las herramientas mencionadas ofrecen al administrador de red facilidades para la gestión de los servicios de infraestructura pero poseen deficiencias principalmente en la gestión de todos los parámetros de configuración de los servicios y la integración de estos. Otra de las deficiencias que poseen estas soluciones está relacionada con la posibilidad de importar configuraciones de los servicios alojados en otros sistemas operativos como por ejemplo Windows Server. La mayoría de estas soluciones no ofrecen un API bien documentada que pueda servir de base para reutilizar las bibliotecas de clase y componentes que tienen estos proyectos limitando la colaboración y tampoco como proyectos reutilizan las bibliotecas de clases software libre y código abierto existentes empezando en su mayoría de cero implementando nuevas API y bibliotecas de clases. La mayoría de las bibliotecas de clases, APIs, marcos de trabajo y componentes existentes gestionan determinados servicios de infraestructura pero se carece de un diseño común o modelo de gestión que agrupe todos estos componentes reutilizables en un solo proyecto que permita disminuir las deficiencias descritas en este trabajo.

1.5 Desarrollo de software orientado a la gestión de servicios de infraestructura

Las herramientas informáticas para la gestión de servicios de infraestructura son software cada vez más complejos y por tanto se impone la necesidad de reutilizar código fuente, APIs, bibliotecas de clases y marcos de trabajo especializados en la gestión de servicios, todo con el objetivo de no duplicar esfuerzo, disminuir el tiempo de desarrollo, obtener una aplicación eficaz y acorde a las necesidades reales del administrador de red. Son

disímiles las API, bibliotecas de clases y marcos de trabajo, de código abierto o software libre existentes y públicos en Internet, que son reutilizables y orientados a la gestión de determinados servicios de infraestructura.

Bibliotecas de clases reutilizables

Existen múltiples proyectos en Internet disponibles que se especializan en la gestión de un determinado servicio y ofrecen la posibilidad de gestionar la mayoría de los parámetros de configuración del servidor en GNU/Linux. Python es uno de los lenguajes de programación sobre el cual se han desarrollado bibliotecas de clases con este propósito, tal es el caso de python-ldap y python-dnspython. Como el nombre indica gestionan el servidor LDAP y el servidor DNS respectivamente. Python-ldap es una de las bibliotecas de clases, orientada a objetos, más completa, para la gestión y configuración de un servidor LDAP. Ofrece la mayoría de las funcionalidades necesarias para la gestión del servicio de directorio. En comparación con otras bibliotecas, como por ejemplo python-ldaptor [25], es más actualizada y mantenida con mayor frecuencia. Python-ldap posee funcionalidades como la gestión de esquemas así como la lectura y generación de formato DSML que otras bibliotecas de clases no incorporan hasta el momento [26]. Otra biblioteca de clases similar a python-ldap para el lenguaje de programación Java lo es Java Ldap [27].

La biblioteca de clase dnspython es una biblioteca de clases para la gestión del servidor DNS. Esta biblioteca permite realizar consultas al DNS, transferencia de zonas así como la manipulación directa de las zonas del DNS, registros, entre otros [28]. Dnspython tiene una homóloga en el lenguaje java disponible en http://www.dnsjava.org/. Existen otras bibliotecas de clases para el acceso al servidor mediante el protocolo SSH, entre ellas se encuentran, python-paramiko y python-pyexpect. Ambas librerías implementan un cliente para la comunicación mediante SSH. Existen otras bibliotecas de clases tales como pydhcplib, disponible en [29], para la manipulación y codificación de paquetes DHCP pero no gestiona el servidor DHCP.

Marcos de Trabajo

Los marcos de trabajo son aplicaciones reusables y semicompletas, especializadas para producir software personalizado o específico [5] [6]. Un marco de trabajo permite reutilizar código fuente, disminuir el tiempo de desarrollo del software concentrando al desarrollador en los temas específicos del proyecto lo cual proporciona ventajas para el desarrollo de cualquier herramienta. Para la gestión de servicios de infraestructura existen frameworks que gestionan una parte de un determinado servicio y sirven de referencia para el diseño de un marco de trabajo orientado a la gestión e integración de servicios de infraestructura, a continuación una descripción de los principales.

ACE

ACE, Adaptative Communication Enviroment, es un entorno de comunicación de código abierto y libre distribución, es un framework orientado a objetos que implementa varios patrones para la comunicación entre procesos. Posee un conjunto de componentes multiplataforma reutilizables implementados en el lenguaje de programación C++ [8]. En la siguiente figura una vista de la arquitectura de ACE.

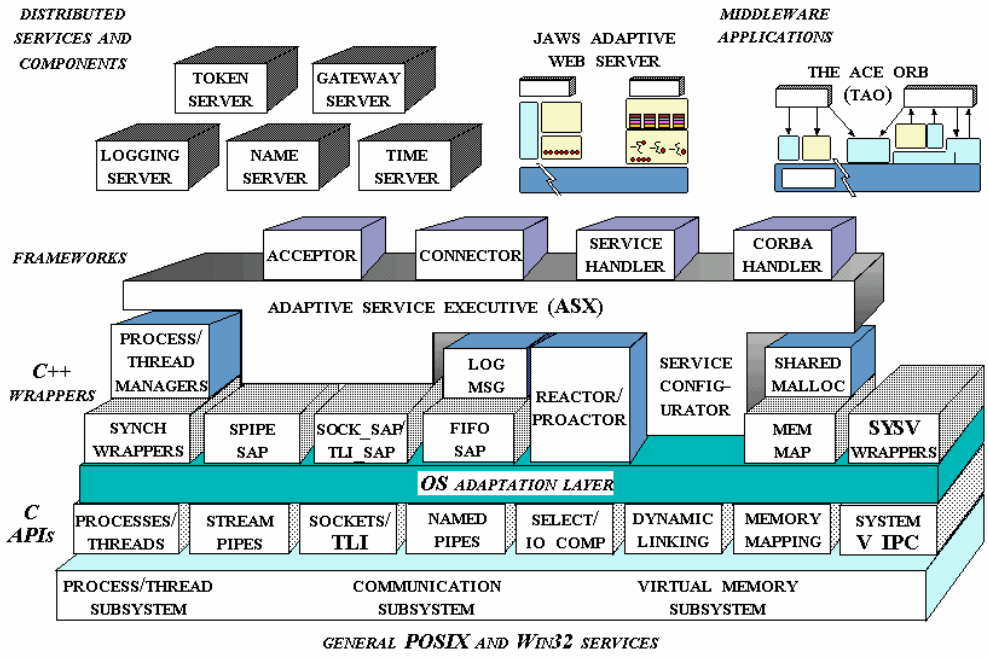


Figura 2. Estructura y funcionalidad de ACE tomada de [8].

ACE está dirigido a los desarrolladores de servicios de comunicaciones de alto

rendimiento y en tiempo real [8]. ACE no está orientado a la gestión de servicios de infraestructura tales como LDAP, DNS, DHCP. ACE se utiliza principalmente en aplicaciones distribuidas que necesiten un alto rendimiento, es un framework para el desarrollo de software distribuido, para la comunicación entre procesos y la programación con sockets así como el desarrollo propiamente de servidores y no el caso específico de gestión de los mismos. De ACE para la gestión de servicios de infraestructura, entre otros elementos, se puede tomar el diseño en capas y los patrones aplicados, lo cual permite a ACE ofrecer un alto nivel de reutilización, abstracción y modularidad en la comunicación entre procesos.

TWISTED

Twisted es un framework de red de código abierto, implementado en Python. Está diseñado para el desarrollo tanto de clientes como servidores y se ejecuta en múltiples plataformas y sistemas operativos [7] [30]. Twisted ofrece un conjunto de librerías integradas para el manejo de protocolos y tareas comunes de programación, tales como la autenticación de usuarios y el acceso a objetos remotos. Twisted soporta los protocolos HTTP, FTP, DNS, SMTP, SSH, Jabber, entre otros, y está compuesto por diferentes subproyectos logrando una mejor organización del framework. En estos paquetes o subproyectos se encuentra una implementación a más alto nivel con un mayor número de funcionalidades para clientes y servidores de los protocolos DNS y HTPP. Con twisted además se pueden desarrollar aplicaciones que necesiten protocolos personalizados [7] [30].

Twisted básicamente es un framework para la comunicación asincrónica entre cliente y servidor y soporta protocolos comunes como HTPP, SSH, DNS, FTP, entre otros. Twisted como marco de trabajo no está orientado a la gestión de servicios de infraestructura sino a la comunicación mediante un protocolo, para el servidor del servicio DNS no ofrece la posibilidad de gestionar todos sus parámetros de configuración, hasta el momento este

marco de trabajo no soporta la gestión del servidor DHCP, y tampoco implementa un cliente DHCP. Twisted soporta, de conjunto con la biblioteca de clases python-ldaptor, un cliente para acceder a un servidor ldap pero no ofrece la posibilidad de gestionar todos los parámetros de configuración del servidor LDAP y componentes importantes como los esquemas.

La documentación publicada en Internet acerca el marco de trabajo Twisted no muestra la arquitectura definida por los desarrolladores lo cual dificulta su comprensión para una posible incorporación al marco de las necesidades requeridas para la gestión de servicios de infraestructura. Varios libros recomendados en la bibliografía consultada en internet hay que pagarlos lo cual también dificulta el acceso a una mejor comprensión del marco de trabajo.

API JDNI

JNDI, Java Naming and Directory Interface, es un API desarrollada en el lenguaje de programación java, forma parte de la plataforma java y provee servicios de nombrado y directorio. Está definido para ser independiente de cualquier implementación de servicio de directorio.

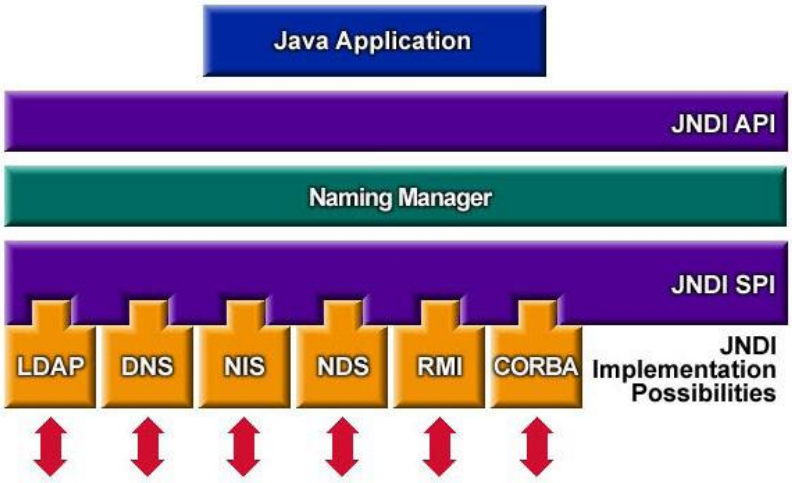


Figura 3. Estructura de JNDI tomada de [9].

JDNI consiste en un API y una interfaz de proveedor de servicios (SPI, Service Provider Interface). La arquitectura SPI de JNDI define un conjunto de plugins que de manera

transparente ofrecen el servicio de nombrado y directorio. JNDI incluye 3 implementaciones de proveedores para los servicios de nombrado y directorio, entre ellos LDAP, CORBA[16] y RMI[17] [9]. JDNI no está orientado a la gestión de servicios de infraestructura sino al servicio de nombrado y directorio. Incluir en su arquitectura e implementación plugins para la gestión de servicios de infraestructura sería costoso y complejo y no cumpliría el objetivo por el cual está definida esta API.

Las disímiles bibliotecas de clases, APIs y marcos de trabajo actuales para la gestión e integración de servicios de infraestructura se especializan en determinados servidores y no están integradas acortando su alcance a determinados servicios, son soluciones en su mayoría orientada a objetos pero independientes. La Programación Orientada a Objeto (POO) es una técnica de programación, un paradigma para desarrollar "buenos" programas para un conjunto de problemas, es un método de implementación en el que los programas se organizan como colecciones cooperativas de objetos [31] [32]. La POO ofrece muchas ventajas para el desarrollo de software complejo como lo son las herramientas de gestión de servicios de infraestructura. El estilo orientado a objetos es el adecuado para el más amplio conjunto de aplicaciones, este paradigma de programación sirve como el marco de referencia arquitectónico en el que se emplean otros paradigmas [32].

Con el objetivo de abarcar la gestión e integración de los servicios de infraestructura actuales y futuros bajos los principios de la POO es necesario la definición de un modelo orientado a objeto que unifique y guie estos desarrollos independientes.

1.6 Conclusiones

La gestión y administración de los servicios telemáticos de infraestructura, con herramientas adecuadas, es vital para brindar servicios de calidad. GNU/Linux es uno de

[16]CORBA: Common Object Request Broker Architecture
[17]RMI: Java Remote Method Invocation

los sistemas operativos más utilizados para ofrecer servicios de infraestructura y la mayoría de las herramientas de código abierto y software libre para la administración de los servicios en plataformas GNU/Linux carecen de la gestión de todos los parámetros de configuración del servidor y posibilidades de integración de los servicios así como de opciones para apoyar el proceso de migración de los servicios ejecutados en plataformas propietarias a soluciones software libre.

Reutilizando las bibliotecas de clases existentes como python-ldap, python-dns, los patrones de diseño del framework ACE, JDNI y Twisted y las buenas prácticas de las soluciones integradoras descritas anteriormente, como FreeIPA, Zentyal y Webmin, se puede diseñar un marco de trabajo orientado a objeto que potencie la gestión de configuración e integración de los servicios de infraestructura en GNU/Linux.

El desarrollo de software es complejo y aún más cuando se trata de gestión de servicios de infraestructura por tanto es importante reutilizar todos los componentes que en la actualidad gestionan al menos un servicio determinado de manera eficaz. Todos estos componentes, conceptos, bibliotecas de clases y soluciones de manera integrada en un solo diseño propician el desarrollo de herramientas eficaces para la gestión de configuración los servicios de infraestructura y la integración de los mismos.

CAPÍTULO 2. DISEÑO DE UN MARCO DE TRABAJO ORIENTADO A LA GESTIÓN DE SERVICIOS TELEMÁTICOS DE INFRAESTRUCTURA

2.1 Introducción

El desarrollo de las herramientas informáticas es cada vez más complejo, el tiempo que se emplea se traduce en costo para las empresas que lo desarrollan y para alcanzar un resultado deseado se deben realizar varias acciones importantes. Una de estas acciones principales es el diseño y la arquitectura del software. La arquitectura del software dirige la organización de todo el sistema, la interrelación entre los distintos componentes que lo conforman así como los principios que orientan su diseño, define la estructura del sistema [33] [34]. El diseño y arquitectura del software influye en la calidad final del producto y su aceptación por el usuario.

En el presente capítulo se describe el diseño y arquitectura de un marco de trabajo orientado a la gestión de configuración de servicios de infraestructura, sus características fundamentales, componentes y patrones de diseño utilizados.

2.2 Diseño y arquitectura del marco de trabajo

Un marco de trabajo o framework es un diseño abstracto orientado a objetos para un determinado tipo de aplicación, es un patrón arquitectónico que proporciona una plantilla extensible para un tipo específico de aplicaciones [35]. Según [36] un framework o “esquema” es un subsistema expandible de un conjunto de servicios, es un conjunto cohesivo de interfaces y clases que colaboran para proporcionar los servicios de la parte central e invariable de un subsistema lógico.

Para la concepción del marco de trabajo se tuvieron en cuenta varios principios:

- Definir áreas que de manera lógica agrupen los conceptos fundamentales para el

desarrollo de software orientado a la gestión de los servicios de infraestructura.

- Organizar las áreas por niveles definiendo un esquema de uso y abstracción de los componentes principales.
- Cada área definida debe exportar o brindar al resto sus funcionalidades principales fomentando la reutilización en todas las áreas.
- Cada área debe definir interfaces de comportamiento con el objetivo de poder incorporar nuevos componentes a las áreas y hacer extensible el marco de trabajo.

Basado en estos principios el marco de trabajo posee una arquitectura orientada a componentes y está compuesto por cinco paquetes principales: **Núcleo**, **Servicios**, **Persistencia**, **Componentes** y **Aplicación de Usuario**. A continuación la vista principal del diseño y arquitectura del marco de trabajo.

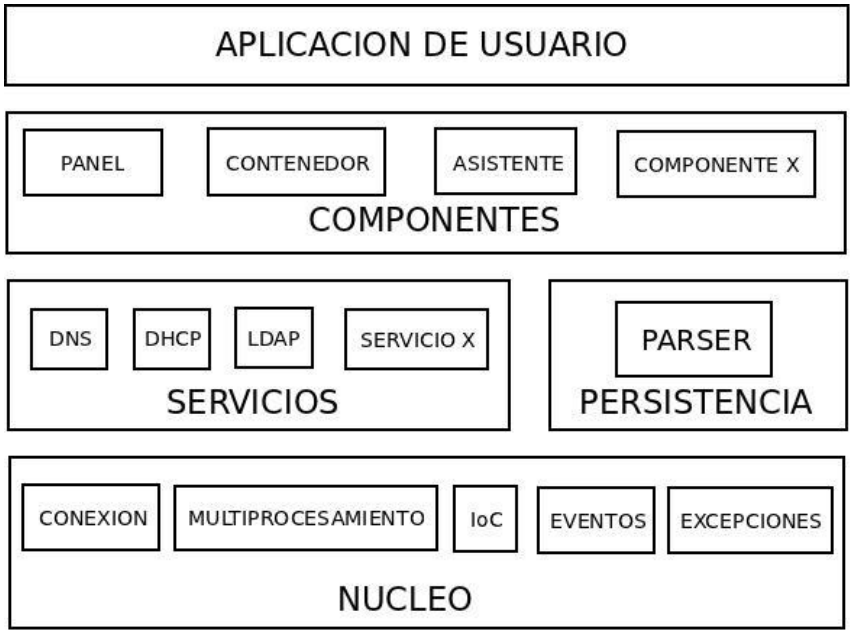


Figura 4. Marco de trabajo.

Núcleo es el paquete principal del modelo y en el se agrupan todos los componentes base del marco de trabajo tales como la definición de Eventos, Conexión, Inversión del Control, entre otros. El núcleo es utilizado por el resto de los paquetes.

En **Persistencia** se concentran los componentes y clases especializados en la manipulación del contenido de los archivos de configuración de los servicios de infraestructura. En el paquete **Servicios** se encuentra los componentes y clases

especializadas en la gestión de los servicios de infraestructura, como realizar sus posibles integraciones con otros servicios y se reutilizan las bibliotecas de clases de terceros que gestionan parte del servicio. En **Componentes** se definen los componentes gráficos para la aplicación de usuario utilizados con el objetivo de representar la información de los servicios y ejecutar acciones sobre los mismos. **Aplicación de Usuario** concentra la mayor parte de la lógica de la aplicación de usuario, donde interviene principalmente el desarrollador que utilice el marco de trabajo.

La definición en componentes del marco de trabajo permite especializar las áreas en la gestión de configuración e integración de los servicios concentrando al desarrollador en temas específicos de su aplicación. Los niveles permiten organizar los paquetes de componentes aumentando el grado de reutilización del marco.

En la mayoría de los marcos de trabajo se definen configuraciones, con un determinado formato, las cuales que permiten extender las funcionalidades del marco de trabajo. El formato a utilizar en las configuraciones de este framework es XML. La mayoría de las plataformas de desarrollo tienen clases que realizan la lectura y escritura de información en estos formatos. El desarrollador además tendrá la posibilidad de proveer la información necesaria que se obtiene de estas configuraciones a partir de otro tipo de formato.

2.3 Núcleo del marco de trabajo

El Núcleo, paquete principal del marco de trabajo, está compuesto por los componentes Inversión del Control (IoC), Eventos, Multiprocesamiento, Excepciones y Conexión. El componente Eventos implementa el patrón de diseño Observer u Observador. Un patrón es una descripción de un problema y su solución, que recibe un nombre y puede emplearse en otros contextos [36]. Un patrón de diseño nombra, motiva y explica un diseño general que resuelve un problema de diseño que ocurre sistemáticamente en sistemas orientados a objetos [37] [38]. Observer es un patrón de comportamiento que

permite notificar a un conjunto de objetos el cambio de estado de otro objeto estableciendo una relación de uno a muchos y un esquema de comunicación entre estos objetos [37]. Eventos define una implementación del patrón Observer y es utilizado por varios componentes.

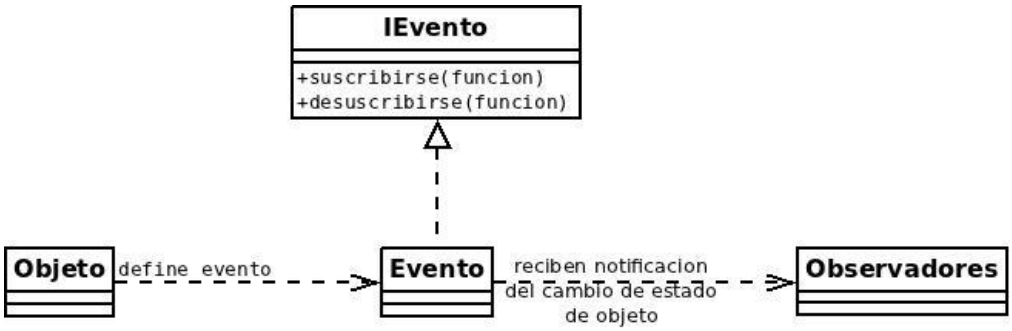


Figura 6. Componente Eventos.

En la figura 6 Objeto representa a las clases que definen un evento y Observadores al grupo de clases que puedan estar suscritas al evento en espera de una notificación de cambio de estado de Objeto. IEventos es la interfaz para la definición de eventos del marco de trabajo.

Conexión es otro componente de Núcleo el cual ofrece un grupo de clases para garantizar el acceso a los archivos de configuración del servicio. A continuación una vista del componente Conexión:

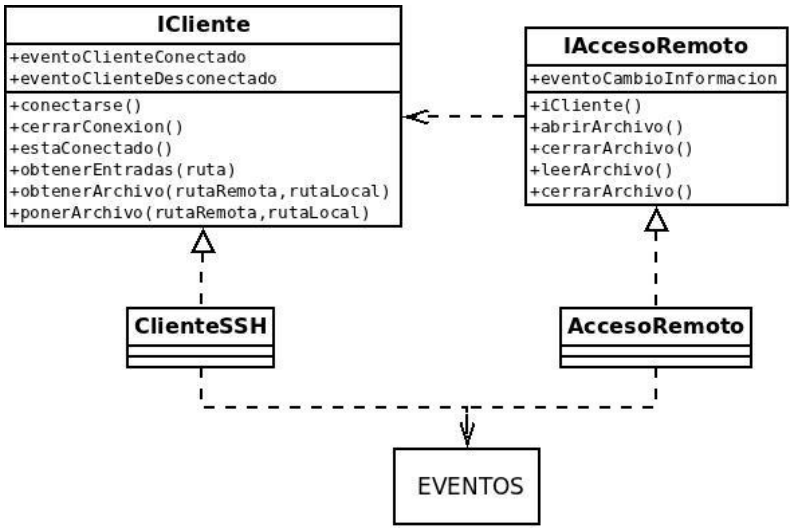


Figura 7. Componente Conexión.

ICliente es la interfaz que define al cliente de conexión por SSH que utiliza el marco de

trabajo para el acceso a las configuraciones de los servicios. IAccesoRemoto es una abstracción que utiliza una definición de ICliente para el acceso y manipulación de los archivos de configuración del servicio, no el contenido del archivo, de la manipulación del contenido se encarga otro componente a un nivel superior. ClienteSSH y AccesoRemoto hacen uso del componente Eventos con el objetivo de informar los momentos de conexión y desconexión del cliente SSH y los cambios de información de los ficheros de configuración del servicio.

Conexión implementa el patrón Adapter [37] o Adaptador, definiendo distintas vías de conectarse a un servicio de infraestructura y obtener sus archivos de configuraciones respectivamente. Conexión es un componente especializado en el acceso a los servicios de infraestructura a través del protocolo SSH. Se puede extender y adaptar a otros protocolos como LDAP en el caso del Servicio Directorio.

Uno de los términos generales que expone Larman en [36] es el “Principio de Hollywood”: “No nos llames, nosotros le llamamos” donde plantea que las clases definidas por el usuario recibirán mensajes de las clases previamente definidas en el esquema (framework). La mayoría de los objetos tienen dependencias por lo que necesitan una forma de buscar y satisfacer estas dependencias aplicando buenas practicas [39]. Esto se logra invirtiendo el control de creación de las instancias de los objetos, el usuario desarrollador ya no es el responsable de crear los objetos sino el marco de trabajo, el cual se encarga de resolver las dependencias. La inversión de control define a los marcos de trabajo [6].

IoC es el componente principal del Núcleo e implementa el principio Inversión del Control. Este componente brinda la posibilidad de definir contextos o dominios de aplicación, abstrae al usuario de la implementación concreta de un grupo de clases, definiendo objetos que son publicados para su futuro acceso.

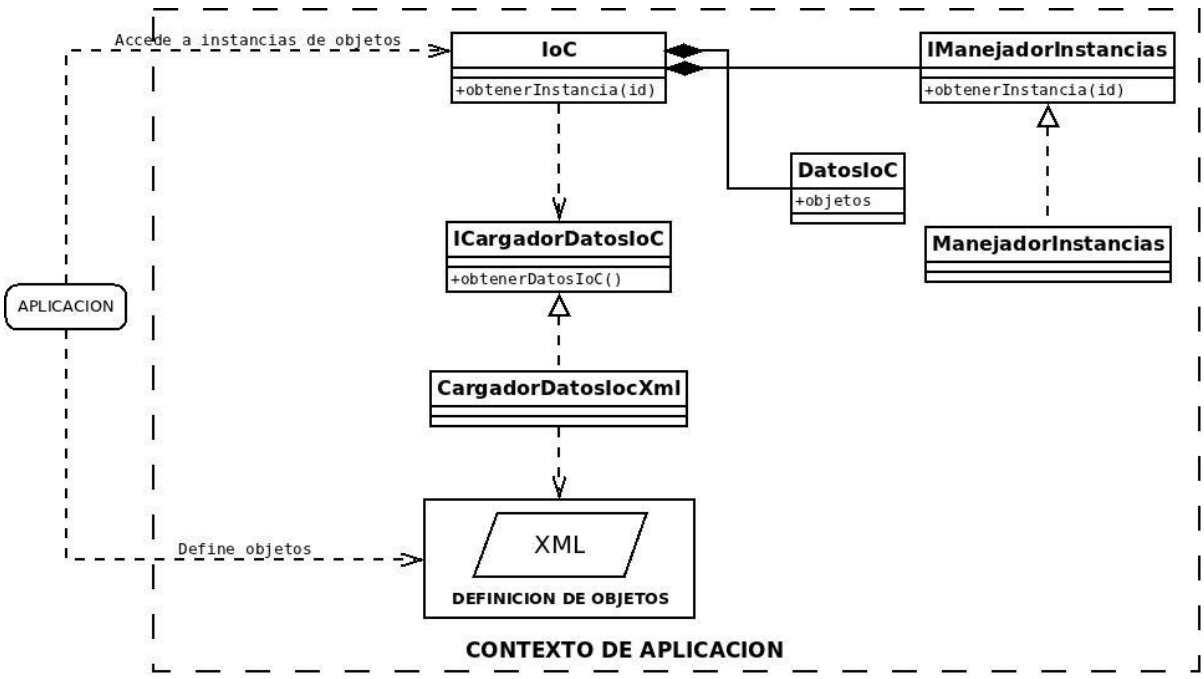


Figura 8. Componente IoC del núcleo.

La interfaz ICargadorDatosIoC posee implementaciones para obtener los objetos definidos en configuraciones XML. El desarrollador tiene la posibilidad de incorporar nuevos formatos para definir objetos, solo debe implementar la interfaz ICargadorDatosIoC. Internamente este componente utiliza una implementación de la interfaz IManejadorInstancias para gestionar las instancias de los objetos definidos.

Los objetos para el componente IoC se definen con un enfoque similar a los *beans* del framework spring [40] lo cual puede ser provechoso en cuanto a adaptación, para programadores que deseen utilizar este marco de trabajo y provengan de un proyecto donde se utilice el framework spring. A continuación un ejemplo de la definición de un objeto en formato XML:

```
<objeto id="servidor" ruta="moduloA.interfazB" imp="moduloA.ImpClaseB" singleton="no" />
```

El atributo ruta es la ubicación relativa de la interfaz de comportamiento que define al objeto que se desea representar, el atributo *imp* es la ruta a la clase que implementa interfaz de comportamiento definida en el atributo ruta. El atributo *id* de objeto en el xml definen el identificador a través del cual se accederá al objeto, *singleton* define si obtener la misma instancia de la clase creada por primera vez o una diferente cada vez que se

solicite, proporcionando una implementación del patrón Singleton o Instancia Única. El patrón Singleton define la creación de una instancia única para una clase y provee un punto global de acceso para esta instancia. [36] [37] [38]. Con las rutas definidas se obtienen dinámicamente las clases y se asocian al identificador dado, luego el acceso se proporciona de acuerdo al ámbito definido en el atributo singleton. A partir de estas configuraciones el componente IoC proporciona un contexto de aplicación para el acceso a los objetos definidos.

Los sistemas operativos modernos brindan la posibilidad de ejecutar procesos o tareas simultáneas y dentro de estos hilos o *threads* de ejecución, aumentando su rendimiento. Para el desarrollo de herramientas informáticas es importante el uso de estas facilidades que brindan los sistemas operativos modernos con el propósito de incrementar el rendimiento y rapidez de la aplicación [41], con este objetivo se define el componente Multiprocesamiento.

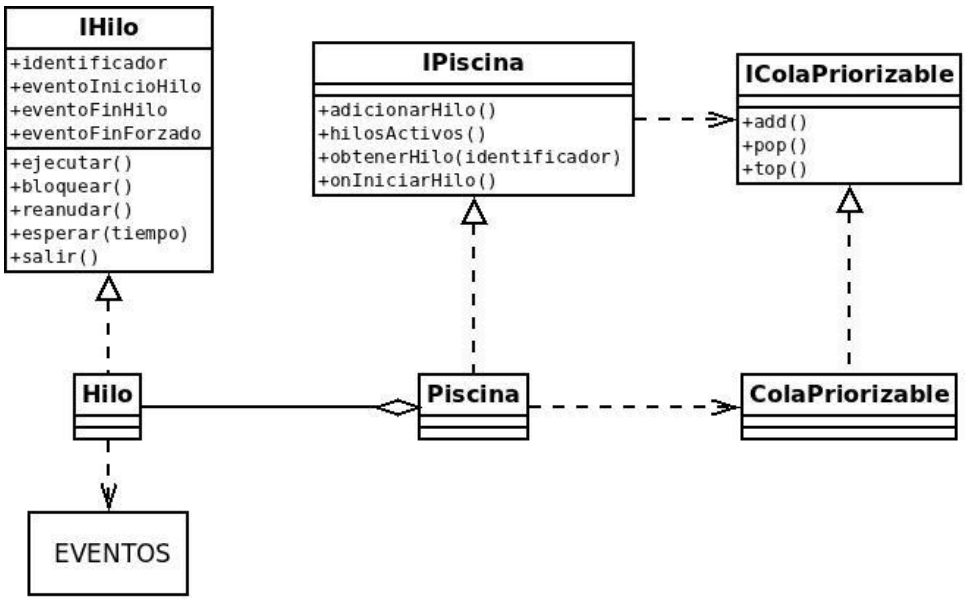


Figura 9. Componente Multiprocesamiento.

IHilo representa la ejecución simultánea de una tarea y define eventos para conocer el momento de inicio, finalización y fuerce del cierre de un hilo. IPiscina define la ejecución de los hilos a través de una cola con prioridad. Esta piscina tiene una implementación similar al patrón Thread Pool o Piscina de Hilos [42].

Excepciones es el otro componente de Núcleo en el cual se agrupan todas las clases que definen los tipos de error que pueden arrojar los métodos de todas las clases del marco de trabajo, como clase error principal se define la clase AplicacionExcepcion y el resto de las clases error heredan de esta.

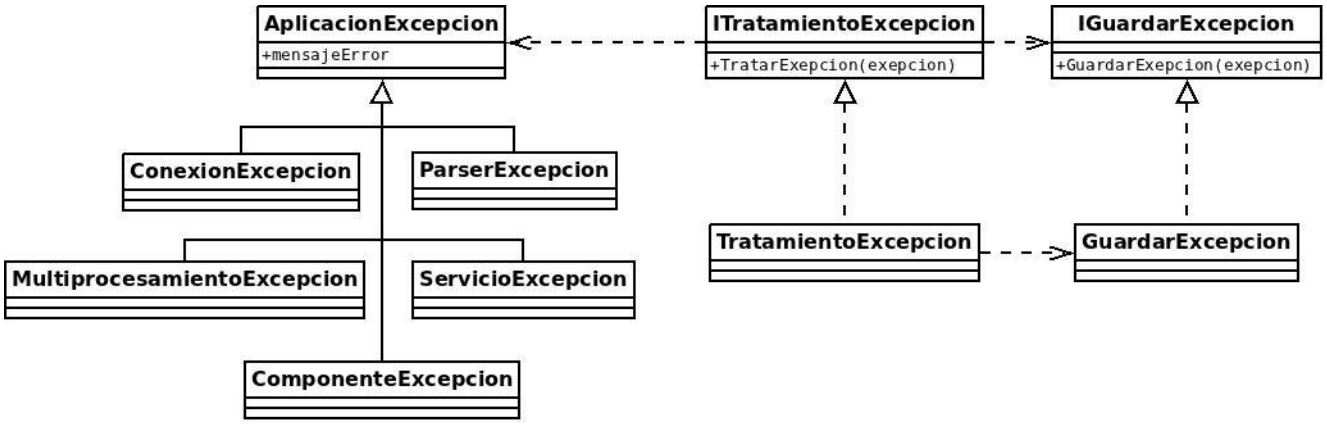


Figura 10. Componente Excepciones del Núcleo.

El componente Excepciones ofrece además la posibilidad de guardar el mensaje error de las excepciones en un fichero local o en los espacios definidos por el sistema operativo sobre el cual se ejecute la herramienta que utiliza este marco de trabajo, como syslog en GNU/Linux [43].

De manera general en todos los componentes de Núcleo, así como en el resto del marco de trabajo se hace uso de los patrones GRASP[18] (patrones generales de software para asignar responsabilidades) con el objetivo de realizar un diseño eficaz y una correcta asignación de responsabilidades en todas las clases [36]. Núcleo es la base del marco y en él se definen los componentes básicos para el funcionamiento de la aplicación que se desee desarrollar.

2.4 Paquete Servicios y Persistencia del marco de trabajo

El núcleo agrupa las clases y componentes esenciales para el funcionamiento de todo el marco de trabajo, en Servicios y Persistencia se concentran los componentes para el trabajo directo con los servicios. Persistencia se especializa en la manipulación del

[18] GRASP: Del inglés: General Responsability Asignment Software Paterns.

contenido de los archivos de configuración del servicio y Servicios en la gestión del servicio de infraestructura, su integración con otros servicios así como las vías para exportar e importar los datos de un servicio.

La sintaxis de las configuraciones de los servicios de infraestructura no es homogénea y en la mayoría de los casos no se utiliza un formato estándar para definirla. Para la lectura de esta información de los servicios regularmente las herramientas de gestión de servicios de infraestructura utilizan expresiones regulares. La definición de expresiones regulares para leer la información contenida en los archivos de configuración de un servicio de infraestructura se puede convertir en una tarea engorrosa y no llegar a cumplir su objetivo. Con el propósito de obtener la información contenida en los archivos de configuración del servicio de una forma orientada a objeto se define en Persistencia el componente Parser el cual implementa el patrón de diseño Adapter.

Parser se nutre de plantillas que representan el patrón de información que puede estar contenido en los archivos de configuración del servicio de infraestructura y almacena directamente la información a las clases de dominio que representan al servicio, básicamente posibilita a partir de estos patrones de información llevar el contenido de los archivos del servicio a clases que representan los objetos del servicio.

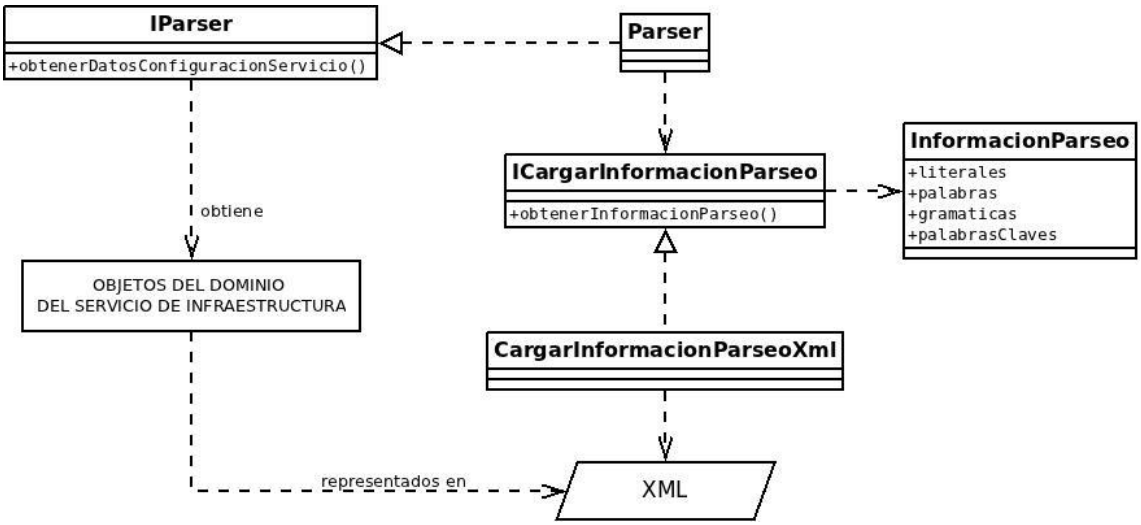


Figura 11. Componente Parser

Para el uso de este parser se debe crear una plantilla, que no es más que un XML, donde

se definen *Literales*, *Palabras* y *Gramáticas* que representan la información del texto del fichero de configuración. Las gramáticas constituyen los patrones de información mencionados y su función es definir la relación que existe entre los Literales, Palabras y Gramáticas de la plantilla, y como estos forman las clases que representan los objetos del servicio de infraestructura.

Los literales representan a los símbolos que denotan delimitación, puntuación, agrupación, entre otros, tales como llaves, puntos, corchetes, comillas, etc. Las palabras son todas las combinaciones de letras y números, dentro las palabras se encuentran las palabras reservadas que son aquellas palabras que identifican partes en el texto del fichero, por ejemplo: *subnet* es una palabra reservada del archivo de configuración del DHCP de ISC que denota el inicio de la definición de una subred. Una Gramática es una agrupación de Literales, Palabras y Gramáticas. A continuación un ejemplo de definición de Literales, Palabras y Gramática, partiendo del siguiente texto perteneciente un archivo de configuración del DHCP de ISC:

Texto:

```
subnet 10.5.6.0 netmask 255.255.255.224
{
  range 10.5.6.26 10.5.6.30;
}
subnet 10.7.7.0 netmask 255.255.255.224
{
  range 10.7.7.26 10.7.7.30;
}
```

Podemos representar este texto en una plantilla XML para el componente Parser de la siguiente forma con Literales, Palabras y Gramáticas.

Literales:

```
<Literales>
      <Literal id="lbrace" valor="{"/>
      <Literal id="rbrace" valor="}"/>
      <Literal id="semi" valor=";"/>
</Literales>
```

Palabras:

```
<Palabras>
      <Palabra id="subnet"/>
      <Palabra id="range"/>
```

```
      <Palabra id="netmask" />
</Palabras>
```

Gramáticas:

```
<Gramaticas parsear="DHCP">

      <Gramatica id="Subnet" objeto= "SubRed" OneOrMore = "True">
            <p id = "subnet"/>
            <P id = "IPv4"  param = "ipSubred"/>
            <p id = "netmask"/>
            <P id = " IPv4"  param = "mask"/>
            <l id = "lbrace"/>
                  <p id = "range"/>
                  <P id = " IPv4" param = "ipInicioRango"/>
                  <P id = " IPv4" param = "ipFinRango"/>
                  <l id = "semi"/>
            <l id = "rbrace"/>
      </Gramatica>

      <Gramatica id="DHCP" objeto= "Dhcp">
            <g id = "Subnet" param = "subred"/>
      </Gramatica>

</Gramaticas>
```

Las gramáticas agrupan las palabras y literales en el orden de aparición en el texto y la información útil se almacena en clases mediante el atributo *objeto* del xml, el atributo *param* del xml define una propiedad para el objeto referenciado en el atributo id del mismo tab xml, básicamente las gramáticas referencia y construyen las clases de dominio que representan los objetos del servicio. Los tab xml en la gramática, definidos con la letra p y la letra l hacen referencia a las palabras y literales definidos previamente. En el caso del identificador IpV4 se refiere a una palabra contenida por defecto en el parser ya que es común, en la mayoría de las configuraciones de servicio de infraestructura, encontrar direcciones IP. Otro detalle importante es que las gramáticas pueden tener una definición recursiva para declaraciones que se contienen así mismas dentro de un texto, como por ejemplo las definiciones de grupos en el DHCP de ISC, los cuales pueden contener grupos dentro de sí mismos.

Parser utiliza una implementación de la interfaz ICargarInformacionParseo para obtener los literales, palabras y gramáticas definidos en formato XML, esta interfaz, ICargarInformacionParseo, puede ser extendida por el desarrollador si desea definir otra

forma de representar los literales, palabras y gramáticas. Puede además extender la interfaz IParser si desea realizar una implementación que permita obtener de otra forma los objetos que representa al servicio de infraestructura.

El desarrollador de una herramienta para la gestión de configuración servicios de infraestructura, con este componente que ofrece el marco de trabajo, logra llevar toda la información del servicio a objetos de una manera intuitiva y eficaz. De esta forma el desarrollador se abstrae de la sintaxis del servicio y se concentra en los detalles específicos del mismo.

Otro de los paquetes del marco de trabajo es **Servicios** donde se agrupan las clases que se especializan en la gestión de los servicios de infraestructura apoyadas en bibliotecas de clases de terceros. En la siguiente figura se muestra un diagrama de clases que representa de manera general el contenido del paquete Servicios:

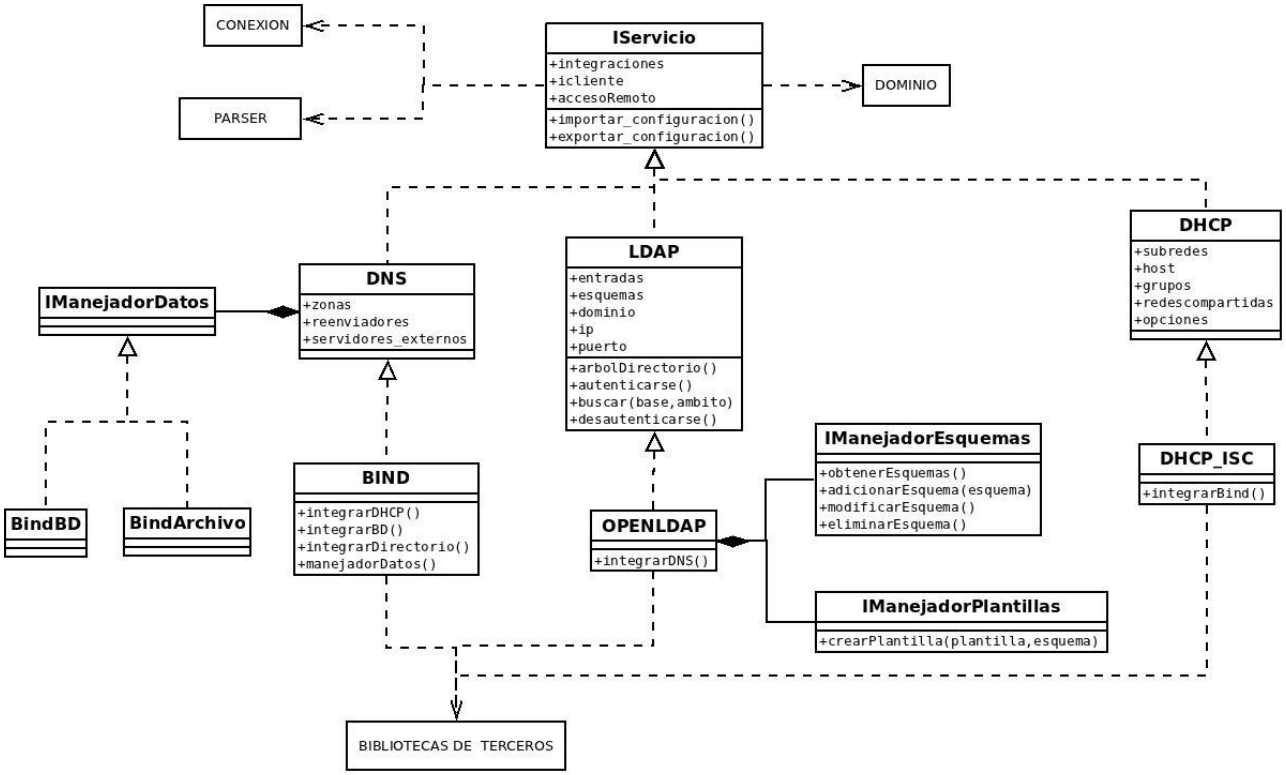


Figura 15. Paquete Servicios

El paquete Servicios define una interfaz general para la representación de los servicios de infraestructura, IServicio, y de esta heredan el resto de las clases servicio, esta interfaz implementa el patrón Fachada o Facade ofreciendo una interfaz común para un conjunto

heterogéneo de interfaces [36] como lo son las clases DNS, LDAP y DHCP. Para el acceso a las configuraciones de los servicios se utiliza el componente Conexión y para la manipulación del contenido de los archivos de configuración el componente Parser. Cada servicio de manera independiente igualmente define una interfaz de comportamiento y de esta hereda la implementación específica la cual utiliza las bibliotecas de terceros para realizar toda la gestión del servicio de infraestructura, las integraciones posibles de los servicios y todo detalle específico relacionado con el servicio telemático de infraestructura.

La clase Bind que implementa la interfaz DNS se encarga de gestionar todos los parámetros del Bind de ISC, zonas, reenviadores, servidores externos, vistas, entre otros parámetros a través de las implementaciones de la interfaz IManejadorDatos las cuales se especializan en la gestión de los datos de Bind en dependencia del lugar donde estén almacenados. La clase Bind también se encarga de realizar las distintas integraciones con el servicio DHCP de ISC, con bases de datos a través de la variante Bind DLZ, y con el servicio de Directorio e importar las configuraciones de servidor DNS de Windows Server 2003. Bind internamente para proveer estas facilidades debe utilizar las bibliotecas de clases existentes que proporcionan parte de esta gestión.

La clase DHCP_ISC gestiona todas las declaraciones posibles que se pueden realizar con el DHCP de ISC, así como las asignaciones de direcciones IP que realiza este servidor a los distintos host de la red, permite además realizar la integración con el servidor Bind. DHCP_ISC es una clase sencilla comparada con la clase Bind y OpenLdap.

OpenLdap es la otra clase del paquete Servicios y se encarga de manipular las configuraciones del servidor OpenLdap, gestionar los esquemas del servicio, tema importante para la personalización de la información que puede estar contenida en el servicio de directorio y para la integración de este servicio con el resto, para la manipulación de los esquemas utiliza dos interfaces, IManejadorEsquemas e IManejadorPlantillas, especializadas en la gestión de los esquemas y plantillas del

servidor OpenLdap. La clase OpenLdap permite además realizar la integración de OpenLdap con BIND.

Servicios es un paquete totalmente reutilizable, el cual puede extenderse de muchas formas, se puede extender incorporando otros servicios, heredando directamente de la interfaz IServicio y definiendo una interfaz de comportamiento propia del nuevo servicio a incorporar. También para las interfaces ya definidas que implementan Bind, DHCP_ISC y OpenLdap se pueden realizar implementaciones para gestionar otras variantes de los servicios DNS, DHCP y Directorio.

2.5 Paquete Componentes y Aplicación de Usuario

Los paquetes Componentes y Aplicación de Usuario están relacionados principalmente con la interfaz gráfica de la aplicación de usuario que se desee desarrollar. En Componente se definen los componentes visuales de usuario, generales y reutilizables para gestionar los servicios de infraestructura, con el propósito de facilitar aún más el trabajo del desarrollador y construir una herramienta intuitiva para interactuar con el usuario de la aplicación. En este paquete también se alojan las implementaciones de los componentes, no solo su definición. Varios de los componentes tienen configuraciones específicas de cómo manipular la información del servicio de infraestructura, necesarias para su funcionamiento las cuales se escriben en formato XML.

Se han definido un grupo de componentes básicos para la gestión de configuración de los servicios de infraestructura: Autenticación, Menú, Árbol, Asistente, Panel, Contenedor, Diálogo Remoto y Barra de Herramientas. Cada componente implementa una interfaz propia de comportamiento la cual a su vez extiende la interfaz IComponente.

Los componentes Árbol, Menú y Barra de Herramientas se deben implementar basados en el patrón Command o Acción [37] encapsulando en un objeto la acción a ejecutarse a partir de la selección de las posibles opciones de menú, árbol y la barra de herramientas.

A continuación se muestra un diagrama con las interfaces principales de la capa

Componentes:

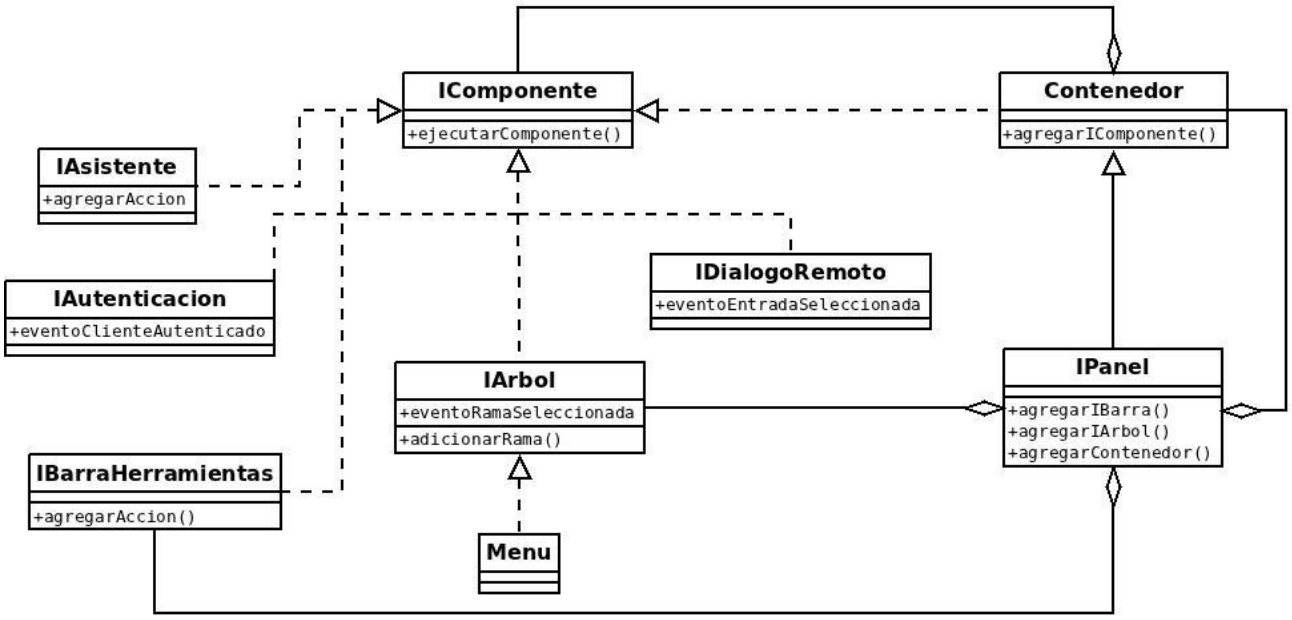


Figura 19. Paquete componentes.

Cada componente define eventos para su interrelación con la aplicación y a su vez con el resto de los componentes. Autenticación define el evento ClienteAutenticado, para notificar la autenticación correcta al sistema operativo del servicio utilizando una definición ICliente del núcleo. Árbol es otro de los componentes y para su funcionamiento se nutre de los objetos que representan al servicio de infraestructura, se utiliza fundamentalmente para mostrar toda la información contenida en los objetos del servicio de manera jerárquica. Árbol define el evento RamaSeleccionada a partir del cual la función observadora conoce el nodo del árbol encuestado y puede realizar acciones de gestión sobre el mismo o la ejecución de otros componentes. El componente Menú es un árbol y define un menú clásico de aplicación a ubicarse en la parte superior de la interfaz gráfica principal.

Diálogo Remoto define una ventana de diálogo clásica de cualquier entorno de desarrollo con la diferencia que utiliza una definición de ICliente y muestra el sistema de ficheros remoto del servidor, este componente define el evento EntradaSeleccionada para las operaciones comunes de ventanas de diálogo como salvar y guardar. Los archivos de

configuración de los servicios de infraestructura en GNU/Linux se ubican en rutas conocidas pero puede darse el caso que cambien su ubicación a partir de una actualización del servicio o de una instalación personalizada, este componente DiálogoRemoto facilita la ubicación visual de los ficheros de configuración del servicio de no estar en el directorio conocido por defecto.

Asistente establece un orden de ejecución de varias ventanas gráficas con el objetivo de llegar a realizar una determinada acción final. Asistente es un componente común utilizado en aplicaciones que guían al usuario a través de pasos lógicos para realizar determinadas configuraciones como es el caso de las integraciones de los servicios de infraestructura. Barra de Herramientas es otro componente que define una interfaz para mostrar en un orden lineal un grupo de acciones y con un ícono de manera representativa, la selección de una acción constituye la ejecución de otro componente o interfaz visual.

Panel y Contenedor son componentes que agrupan componentes, un panel está formado por un menú, Barras de Herramientas y Árboles y estos componentes pueden ser agrupados en contenedores. La aplicación puede definir un panel como interfaz gráfica general y sobre este incorporar el resto de los componentes. La implementación de los componentes está en correspondencia con la aplicación a desarrollar, el marco de trabajo solo define el componente.

Para la mayoría de estos componentes se define un XML de configuración. Panel define un XML que agrupa al resto de los componentes:

```
<?xml version="1.0" encoding="utf-8" ?>
<Panel nombre = "Nombre de la herramienta">
      <Menu ruta=" menu.xml" />

      <Arbol ruta="arbol.xml"/>

      <BarraHerramientas ruta="Configuraciones/barraH.xml"/>

      <EntornoInicial ruta="Modulo.Clase"/>

      <Propiedades largo="0" ancho="0" maximizado="True" largoMaximo="0"
```

```
                    largoMinimo="0" anchoMaximo="200" anchoMinimo="300" />
</Panel>
```

Dentro de este XML el desarrollador define la ubicación de las configuraciones del resto de los componentes que integran el Panel, como el árbol, la barra de herramientas y la clase a ejecutar que muestra la interfaz inicial dentro del panel. El árbol, menú y barra de herramientas del panel tienen configuraciones similares igualmente en XML definiendo tres atributos principales: el *nombre* a mostrar en la interfaz, icono (*ico*) a utilizar y *entorno*, parámetro opcional que establece la ruta a la clase que se suscribe automáticamente al evento RamaSeleccionada de estos componentes con el objetivo de ejecutar otro componente o una determinada interfaz gráfica a mostrar en el panel. A continuación un ejemplo de XML de configuración del componente árbol y del menú del panel.

Ejemplo de XML de configuración para el componente árbol.

```
<?xml version="1.0" encoding="utf-8" ?>
<arbol nombre="DHCP" ancho="300" ico=":/Dhcp.png">
      <Rama nombre="Subredes" ico=":/SubRed.png" />
      <Rama nombre="Host" ico=":/Zona.png" />
      <Rama nombre="Redes Compartidas" ico=":/Reenviadores.png" />
      <Rama nombre="Grupos" ico=":/Reenviadores.png" />
      <Rama nombre="Opciones" ico=":/Acl.png" />
</arbol>
```

Ejemplo de XML de configuración de menú para el panel.

```
<?xml version="1.0" encoding="utf-8" ?>
<menu nombre=" DNS">
       <Rama nombre = "Archivo" >
            <Rama nombre="Nuevo Servidor" ico=":/Dns.png" ent="Modulo.Clase" />
            <Rama nombre="Importar" ico=":/Dns.png" ent=" Modulo.Clase" />
            <Rama nombre="Exportar" ico=":/Dns.png" ent=" Modulo.Clase " />
            <Rama nombre="Reportes" ent=" Modulo.Clase ico=":/Zona.png" />
```

```
    </Rama>
</menu>
```

El XML de configuración de una barra de herramientas es similar al del menú o el árbol, solo se diferencian en el tab inicial que sería *barra*, el resto de los tab xml y atributos son los mismos.

Estos componentes también se pueden crear de manera dinámica como por ejemplo para el caso del servidor de Directorio donde no se conoce exactamente el número de entradas del árbol y su organización. El desarrollador puede definir nuevos componentes y ubicarlos dentro del marco de trabajo, extenderlos y mejorar los ya definidos.

El paquete Aplicación de Usuario se compone por el sistema a desarrollar. En cada uno de los módulos del sistema el desarrollador podrá acceder a los componentes del marco de trabajo y redefinir el uso de cualquiera de ellos e incorporar nuevos servicios al framework. El programador para el desarrollo de la aplicación cuenta con el componente Eventos para la definición de eventos basados en el patrón Observador, Multiprocesamiento para ejecutar las tareas de la aplicación de manera simultánea y aumentar su rendimiento, Excepciones para tratar los errores del sistema, Conexión para acceder al sistema operativo de los servicios y con Parser manipular los ficheros de configuración, Servicios le facilita la gestión de los servicios LDAP, DNS y DHCP.

Para la ejecución de las aplicaciones que se desarrollen sobre este marco de trabajo se debe definir un punto inicial encargado de engranar a través del componente IoC todas las partes de la aplicación una forma transparente y adaptable, esto se realiza a partir de las configuraciones en XML donde estarán principalmente la definición de los componentes y los objetos a alojar en el contexto de aplicación. El contexto de aplicación es un espacio lógico que proporciona el componente IoC a través del cual el desarrollador puede dividir lógicamente su aplicación en capas, componentes, en dependencia de la arquitectura que tenga su aplicación.

2.6 Conclusiones

El uso de marcos de trabajo propicia el desarrollo de herramientas eficaces y flexibles, la reutilización de esfuerzo y código fuente así como el aumento de la productividad de los desarrolladores. El marco de trabajo diseñado ofrece organización y rapidez en el desarrollo de aplicaciones orientadas a la gestión de configuración e integración de servicios de infraestructura.

La arquitectura del framework modelado está basada en patrones que garantizan su extensión y escalabilidad permitiendo al desarrollador en todo momento adaptar el marco de trabajo a sus necesidades. Este marco de trabajo ofrece un diseño orientado a objeto, reutiliza las bibliotecas de clases existentes especializadas en la gestión de determinados servicios, define componentes que pueden ser reutilizados por otros tipos de proyectos y homogeniza la manipulación de los archivos de configuración de los servicios de infraestructura.

El diseño y arquitectura de este framework facilita el desarrollo de herramientas informáticas que disminuyan o eliminen las deficiencias de las aplicaciones actuales en cuanto a gestión de configuración e integración de servicios de infraestructura.

CAPÍTULO 3. HERRAMIENTA DE GESTIÓN E INTEGRACIÓN DE SERVICIOS TELEMÁTICOS DE INFRAESTRUCTURA EN GNU/LINUX

3.1 Introducción

El desarrollo de herramientas informáticas modulares, escalables y ajustadas a las necesidades del usuario final es un proceso difícil. Para simplificar esta dificultad se utilizan marcos de trabajo los cuales proveen un esquema para desarrollar un tipo específico de aplicaciones en el menor tiempo posible. En el Capítulo 2 de esta tesis se expuso el diseño y arquitectura de un marco de trabajo o framework que propicia el desarrollo de herramientas orientadas a la gestión de configuración e integración de servicios de infraestructura, sus componentes y características fundamentales así como los patrones de diseño utilizados. A partir de este modelo, en el presente capítulo, se describe la implementación del marco de trabajo, el lenguaje de programación, tecnologías de desarrollo y las bibliotecas de clases de terceros utilizadas así como una herramienta para la gestión de configuración e integración de los servidores OpenLDAP v2.3, Bind9.x y el DHCP v4.x de ISC desarrollada a partir del marco de trabajo expuesto.

3.2 Lenguaje de programación y tecnologías de desarrollo

El lenguaje de programación que se escoja para la construcción de un determinado tipo de software debe proveer al desarrollador facilidades con respecto al resto de los lenguajes, ventajas que lo identifiquen como el lenguaje idóneo para el desarrollo de este tipo de software. Para el desarrollo de herramientas orientadas a la gestión de configuración e integración de servicios de infraestructura en GNU/Linux, el lenguaje de programación Python ofrece ventajas.

Python es un lenguaje de alto nivel orientado a objeto, de una sintaxis clara y limpia,

posee una amplia biblioteca de clases que facilita el desarrollo de software [44] [45] y una comunidad de software libre que soporta el lenguaje y provee las actualizaciones correspondientes frecuentemente [46].

Como se evidencia en el estudio del estado del arte realizado en el capítulo I python posee, a diferencia de otros lenguajes de programación, varias bibliotecas de clases que realizan una parte de la gestión de los servicios telemáticos de infraestructura contribuyendo al desarrollo de software para la gestión de este tipo de servicios, tales como python-ldap y python-dnspython. Python-ldap es una biblioteca de clases orientada a objeto muy completa para el acceso al servicio directorio, permite entre muchas otras funcionalidades la gestión de las entradas del LDAP, así como el acceso a los esquemas del servicio de Directorio lo cual es importante para personalizar la información que puede estar contenida en el servicio de Directorio, y es una de las deficiencias de las herramientas actuales para la gestión del servicio de Directorio. La otra biblioteca de clases, dnspython, posibilita la gestión de los registros de las zonas del Bind de una manera fácil e intuitiva. Otra de las bibliotecas que posee python que contribuyen a la gestión de los servicios de infraestructura es python-paramiko la cual a través de SSH facilita el acceso al servicio de infraestructura, esta biblioteca permite la autenticación al sistema operativo con usuario y contraseña y también a través de las claves RSA que se pueden generar como parte de la conexión SSH, permite también la ejecución de comandos en el servidor la cual facilita aún más la gestión del servicio, y si es necesario una mejor interacción con la consola del servidor se utiliza python-pexpect una biblioteca muy útil para interactuar con procesos hijos en este caso con SSH.

Para la lectura de los patrones de información en los archivos de configuración python también posee ventajas con respecto a otros lenguajes como es el caso de la biblioteca python-pyparsing la cual facilita la lectura y escritura correcta de los archivos de configuración de los servicios telemáticos de infraestructura de una manera orientada a

objeto, esta biblioteca permite la definición de literales, palabras y tipos de gramáticas a través de las cuales el programador puede representar toda la información contenida en los archivos de configuración del servicio y realizar una gestión eficaz de los archivos de configuración del servicio.

Para la implementación de este marco se escogió Python como lenguaje de programación por todas las ventajas mencionadas que ofrece. Las bibliotecas de clases python-ldap y dnspython se reutilizaron en la implementación del paquete Servicios del marco de trabajo, para la gestión del servicio DHCP de ISC se implementó una biblioteca de clases puramente sobre python que se incorporó al paquete Servicios. Las bibliotecas python-paramiko y python-pexpect fueron reutilizadas en el componente Conexión del Núcleo y python-pyparsing en el paquete Persistencia. Para el resto de los componentes python como lenguaje de programación posee funciones y módulos que facilitaron su desarrollo, para el desarrollo del componente IoC python permitió con varias funciones, tales como *__import__* y *getattr* el acceso e instanciación dinámica de clases a través de las rutas definidas en el XML de configuración. Para el trabajo con las configuraciones en XML se utilizó el módulo *xml.etree*, el cual permite la lectura de una manera fácil de archivos XML. Para la implementación de Multiprocesamiento se utilizaron los módulos *threading* y *multiprocessing* los cuales permiten la ejecución de tareas simultáneas en python con el objetivo de elevar el rendimiento del sistema. En Excepciones para guardar en syslog los mensajes de error se utilizó el módulo *syslog*.

Para el diseño de las interfaces gráficas de los módulos de la herramienta se utilizó la herramienta Qt Designer la cual permite construir interfaces gráficas con la librería Qt. Qt es una librería multiplataforma mantenida por Qt-Project disponible en http://qt-project.org/. La librería Qt está desarrollada en C++ pero tiene implementaciones ampliamente utilizadas en python, en el desarrollo de este software se utilizó la implementación PyQt lo cual está en correspondencia con la selección de python como

lenguaje de programación. Tanto la herramienta Qt Designer, la librería Qt, así como PyQt tienen licencias de software libre y código abierto lo cual permite el uso de estas tecnologías libre de costo siempre y cuando no se utilice para proyectos comerciales. La implementación del paquete Componentes del marco de trabajo como es lógico se realizó con PyQt. Los componentes desarrollados con PyQt son reutilizables para otros proyectos que utilizan Qt como librería gráfica.

Para la codificación de todo el software en python se utilizó la herramienta EasyEclipse disponible en http://www.easyeclipse.org. Como su nombre lo indica está basado en Eclipse, Entorno de Desarrollo Integrado basado en plugins que permite el desarrollo de aplicaciones en varios lenguajes de programación. EasyEclipse es un Eclipse pre configurado para trabajar en python y otros lenguajes de programación, para el trabajo con python específicamente se utiliza el plugin pydev disponible en http://pydev.org/ el cual brinda facilidades para desarrollar software con python.

El marco de trabajo implementado en python define como paquetes principales Núcleo, Servicios, Persistencia y Componentes y dentro de estos se agrupan el resto de los paquetes y módulos python del marco de trabajo como se muestra en la siguiente figura:

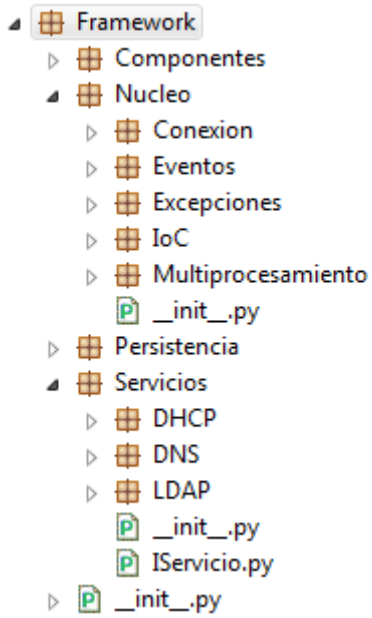


Figura 20. Paquetes principales en python del marco de trabajo.

Python como lenguaje de programación, de conjunto con las bibliotecas de clases y tecnologías mencionadas, posee ventajas que lo convierten en el lenguaje de programación idóneo para la implementación de este marco de trabajo y la herramienta de gestión e integración de servicios de infraestructura. Para el desarrollo del marco de trabajo y esta herramienta su utilizó software libre y de código abierto aprovechando las ventajas que este tipo de software proporciona.

3.3 Arquitectura de software de la herramienta

La herramienta que se muestra en este capítulo está divida en varios módulos, LDAP, DNS y DHCP. Cada uno de estos módulos gestiona el servicio correspondiente y existe otro módulo con nombre Integrador que facilita la instalación e integración de los servicios DNS, DHCP y Directorio. Cada módulo posee una arquitectura basada en el patrón arquitectónico N capas [36] y se soporta sobre el marco de trabajo diseñado en el capítulo II. En la siguiente figura se muestra el diseño de la arquitectura de los módulos:

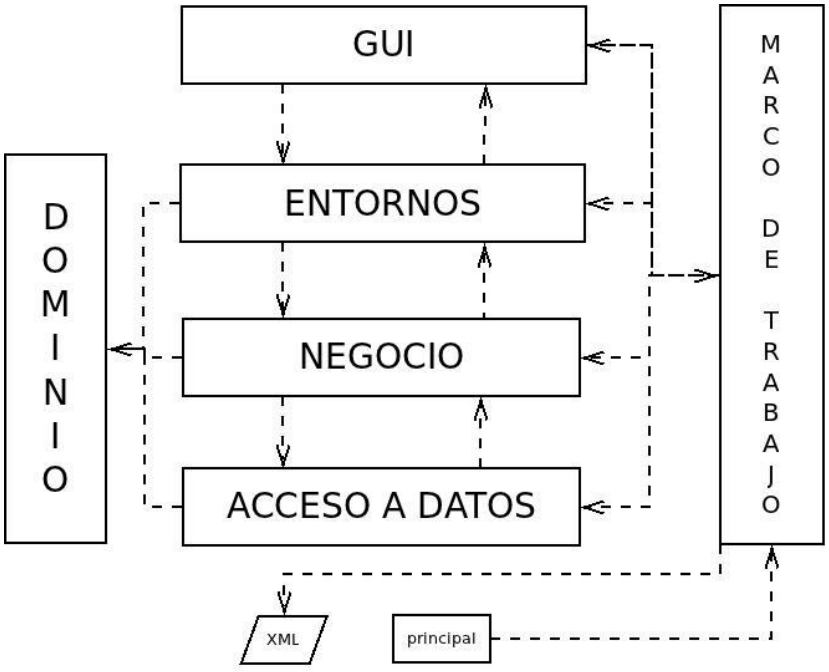


Figura 21. Arquitectura de los módulos.

Una arquitectura multicapas permite aislar la lógica de las aplicaciones en componentes independientes que luego pueden reutilizarse, se alcanza una mejor distribución del

sistema y se puede lograr una especialización por parte de los desarrolladores al estar asignado su trabajo a una sola capa [36].

Las capas de esta arquitectura son GUI, Entornos, Negocio, Acceso a Datos y Dominio. En la capa GUI se alojan todas las clases que representan las ventanas gráficas de usuario y los archivos con extensión .ui resultado del diseño de los interfaces gráficas con la herramienta Qt Designer. Cada una de las clases en esta capa definen con el componente Eventos del Núcleo del marco de trabajo los eventos correspondientes a cada una de las posibles acciones del usuario. GUI también define en sus clases las propiedades a exportar de las ventanas gráficas que tiene información, útil para la gestión de servicios, que es introducida por el usuario.

En la capa Entornos se alojan todas las clases que enlazan la interfaz gráfica de usuario con la gestión propia de los servicios, este enlace se realiza a través de la suscripción a los eventos que define la capa GUI y la información se obtiene a través de las propiedades que exportan cada una de las clases de la capa GUI. Por cada interfaz gráfica de usuario debe existir un entorno que permita el enlace entre GUI y Negocio, encapsulando en un objeto entorno cada acción del usuario, proporcionando una implementación del patrón Acción o Command [39] mencionado anteriormente en el capítulo II en la definición de los componentes Árbol, Menú y Barra de Herramientas. En Entornos el desarrollador utiliza el componente Multiprocesamiento para ejecutar de manera simultánea las acciones que necesiten un mayor rendimiento de la aplicación. Esta capa Entornos garantiza además que el usuario introduce los datos correctos para realizar cualquier acción con los servicios y que las capas GUI y Negocio sean reutilizables incluso para otros tipos de proyectos.

Negocio como capa contiene toda la lógica que permite realizar la gestión de los servicios telemáticos de infraestructura y se apoya fundamentalmente de la capa Acceso a Datos para realizar su trabajo. Negocio manipula la información del servicio representada en las

clases de la capa Dominio sin tener que conocer la forma exacta en que están almacenados los datos.

La capa Acceso a Datos como su nombre lo indica provee el acceso a los datos del servicio y se encarga de manipular los archivos de configuración del mismo a través del componente Conexión y el paquete Servicios del marco de trabajo. Acceso a Datos conoce exactamente como están almacenados los datos de los servicios de infraestructura y abstrae al sistema completo de cómo se realiza el acceso a la información de los servicios haciendo reutilizable la capa Negocio.

Dominio es una capa vertical a las capas Entornos, Negocio y Acceso a Datos donde se concentran las clases que representan el servicio telemático de infraestructura. Cada módulo de la herramienta de gestión e integración de servicios representa su servicio en clases de Dominio facilitando la gestión de los mismos de una forma orientada a objeto.

En esta arquitectura multicapas la ubicación de las capas define un flujo de información que no debe violarse. La información debe fluir desde la capa superior a las capas inferiores y viceversa sin saltar ninguna de las capas. El orden en que están ubicadas las capas por niveles permite una mayor organización del desarrollo de la herramienta, se logra que cada capa se especialice en un área determinada del desarrollo del software y por tanto se logra un mayor grado de reutilización y organización del sistema [36].

La comunicación entre las capas GUI, Entornos, Negocio y Acceso a Datos se realiza a través del componente IoC del marco de trabajo. Cada una de las capas de esta arquitectura, excepto Dominio, define un grupo de interfaces de comportamiento que son exportadas a través del componente IoC del framework permitiendo la comunicación entre las capas y evitando un bajo acoplamiento [36] entre las mismas, esto permite que el sistema sea adaptable a futuros cambios y que no exista una dependencia directa entre las capas lo cual disminuye su reutilización. Estas interfaces y sus implementaciones se definen en el archivo de configuración XML definido en la arquitectura. El archivo XML de

configuración contiene la definición por capa de las interfaces de comportamiento y el punto de ejecución de la herramienta como se muestra de manera general a continuación:

```
<Aplicacion>
      <PuntoEjecucion>
            <Componente nombre = "Entorno">
                  <Interfaz id="Panel.Qt" />
            </Componente>
      </PuntoEjecucion>

      <Capa nombre="nombreCapa">
            < Interfaces>
                  <Interfaz id=" " ruta="" imp=" " singleton="no" />
            </Interfaces>
      </Capa>
<Aplicacion>
```

El punto de ejecución define el componente a ejecutar cuando inicie la aplicación. Por defecto esta implementación del marco de trabajo reconoce la implementación del componente Panel para el id *Panel.Qt*. Luego para acceder a la instancia creada de este componente se haría a través del nombre y el id, *Entorno.Panel.Qt*.

La definición de las capas es similar al punto de ejecución, el nombre de la capa seguido del atributo id de la interfaz define el identificador con el que automáticamente el marco de trabajo guarda, con el componente IoC, el acceso las implementaciones de cada interfaz de comportamiento. El atributo *ruta* define la interfaz de comportamiento y el atributo *imp* la implementación de la interfaz de comportamiento.

A continuación un ejemplo sencillo de configuración XML para un posible módulo:

```
<?xml version="1.0" encoding="utf-8" ?>
<Aplicacion>

      <PuntoEjecucion>
            <Componente nombre = "Panel">
                  <Interfaz id="Panel.Qt" />
            </Componente>
      </PuntoEjecucion>

      <Capa nombre="CapaGUI">
            <Interfaces>
                  <Interfaz id="subred" ruta="GUI.Modulo.ClaseInterfazSubred"
imp="GUI.Modulo.ImpClaseInterfazSubred" singleton="no" />
            </Interfaces>
      </Capa>

      <Capa nombre="CapaServicios" cargar="true">
            <Interfaces>
                  <Interfaz id="servGestDHCP"
```

```
ruta="Negocio.Interfaces.IGestionarDHCP.IServicioGestionarDHCP"
imp="Negocio.GestionarDHCP.ImpGestionarDHCP"/>
            </Interfaces>
      </Capa>

      <Capa nombre="CapaAD"  cargar="true">
            <Interfaces>
                  <Interfaz id="adDHCP"
ruta="AD.Interfaces.IADGestionarDHCP.IADGestionarDHCP"
imp="AD.ADGestionarDHCP.ADGestionarDHCP"/>
            </Interfaces>
      </Capa>
</Aplicacion>
```

Cada capa de la arquitectura se define en el XML y dentro de cada una, las interfaces de comportamiento con sus implementaciones correspondientes. La definición de las interfaces de comportamiento permite a los módulos del sistema adaptarse a cambios futuros en la implementación y abstrae a cada capa de los detalles de la implementación de su capa adyacente.

Para cada uno de los módulos de esta herramienta se utiliza el componente gráfico Panel como interfaz principal. A continuación una imagen que muestra un posible Panel para el servicio DHCP de ISC.

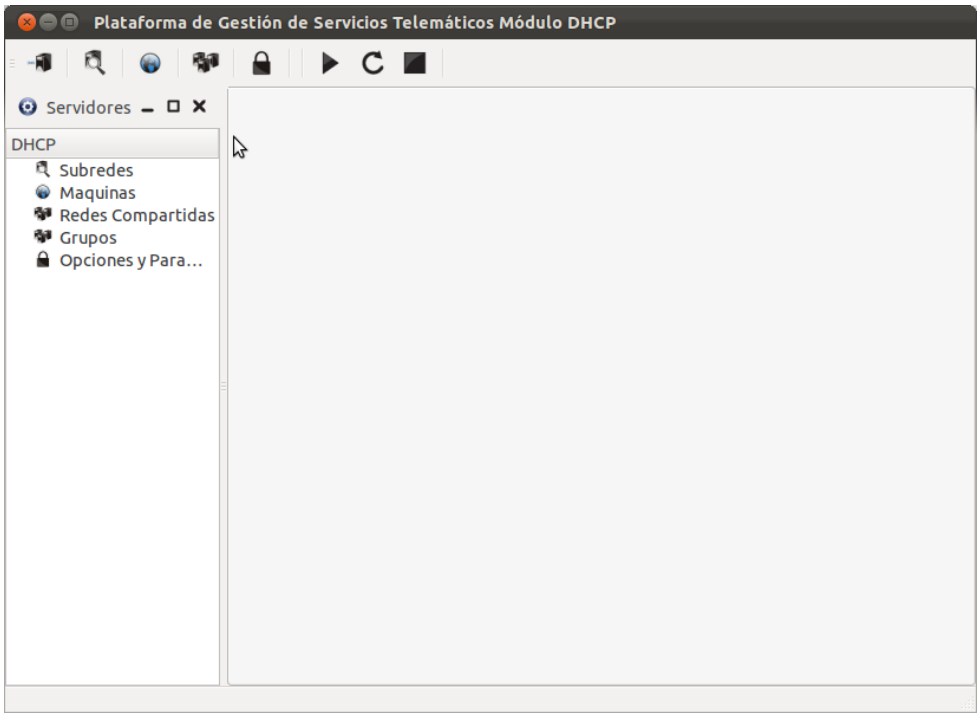


Figura 22. Vista panel de interfaz principal.

Panel define también un XML donde se especifican cada uno de los componentes que

conforman al panel como el menú, la barra de herramientas, el árbol que muestra la información del servicio, y para cada uno de estos las clases entornos correspondientes como se describe en el capítulo II. El XML de configuración del Panel se debe ubicar al mismo nivel en el directorio que el XML de configuración del módulo.

La definición de este panel como interfaz principal para los módulos de la herramienta permite homogenizar la interfaz de usuario para cada uno de los módulos, de esta forma se logra una mejor adaptación del usuario en el uso de esta herramienta de gestión e integración de servicios. A través del menú y la barra de herramientas el usuario podrá acceder a todas las acciones sobre los servicios que ofrece el módulo correspondiente y el árbol le mostrará la información del servicio, sobre la cual también se pueden realizar operaciones de gestión. En el centro del panel se mostrarán las interfaces definidas en la capa GUI que son ejecutadas por los entornos a los cuales se hace referencia en cada uno de los componentes del panel.

Luego de especificar las interfaces de comportamiento de cada una de las capas y sus implementaciones, las configuraciones en XML del componente IoC y el panel, el marco de trabajo, a través del objeto *principal* definido en la arquitectura, es el encargado de ejecutar la herramienta. A partir de la arquitectura definida la estructura del código fuente de los módulos quedaría de la siguiente forma:

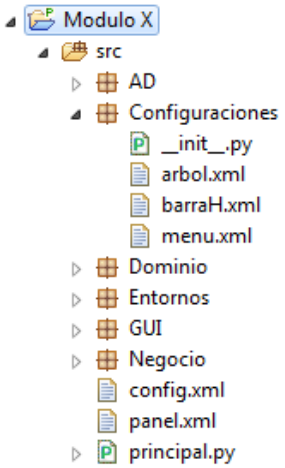


Figura 23. Estructura del código fuente de los módulos.

Para cada capa en la arquitectura existe un paquete python y dentro, las clases correspondientes que hacen uso del marco de trabajo. En el paquete Configuraciones se encuentran las configuraciones de los componentes que referencia el componente Panel. El módulo python *principal.py* ubicado en la raíz, como se explicó anteriormente, es el encargado de ejecutar la herramienta, apoyado en el marco de trabajo y las configuraciones alojadas en el archivo *config.xml*.

Las excepciones de la herramienta que sean tratadas con el componente Excepciones del núcleo del marco de trabajo generarán un archivo con nombre *errores.logs* en la raíz del código fuente de la herramienta, con la traza completa de los errores.

La arquitectura definida para los módulos de esta herramienta es una arquitectura modular y extensible apoyada en marco de trabajo que provee los paquetes y componentes especializados en la gestión de servicios telemáticos de infraestructura concentrando al desarrollador en los detalles específicos de la aplicación.

3.4 Módulos de la herramienta

Los módulos de la herramienta de gestión e integración de servicios telemáticos de infraestructura son aplicaciones de escritorio, basados en la arquitectura descrita en el epígrafe anterior: LDAP, DNS, DHCP e Integrador. Los módulos LDAP, DNS, y DHCP permiten gestionar simultáneamente más de un servidor del servicio correspondiente e importar los datos del mismo servicio ubicado en Windows Server 2003 contribuyendo a la migración paulatina a software libre. Los módulos también brindan opciones de parada, inicio y reinicio del servidor que se esté gestionando en caso de que los cambios realizados en la configuración de estos servicios lo necesiten. Las integraciones entre los servicios se realizan a través del módulo Integrador. A continuación se detallan las opciones que de manera particular brindan cada uno de los módulos.

Módulo DHCP

El módulo DHCP permite gestionar todas las declaraciones del servidor DHCP de ISC

instalado en GNU/Linux, dígase subredes, redes compartidas, host y grupos, así como el resto de los parámetros y opciones de su fichero de configuración. Si el servicio DHCP está instalado de manera personalizada y los archivos de configuración están en otro directorio que no sea el común donde se ubican estos ficheros de configuración, el módulo ofrece la posibilidad de buscar ese otro directorio para poder gestionar el servicio DHCP. El módulo en la parte izquierda en forma de árbol grupa toda la información de sus parámetros de configuración en subredes, redes compartidas, host, grupos y opciones.

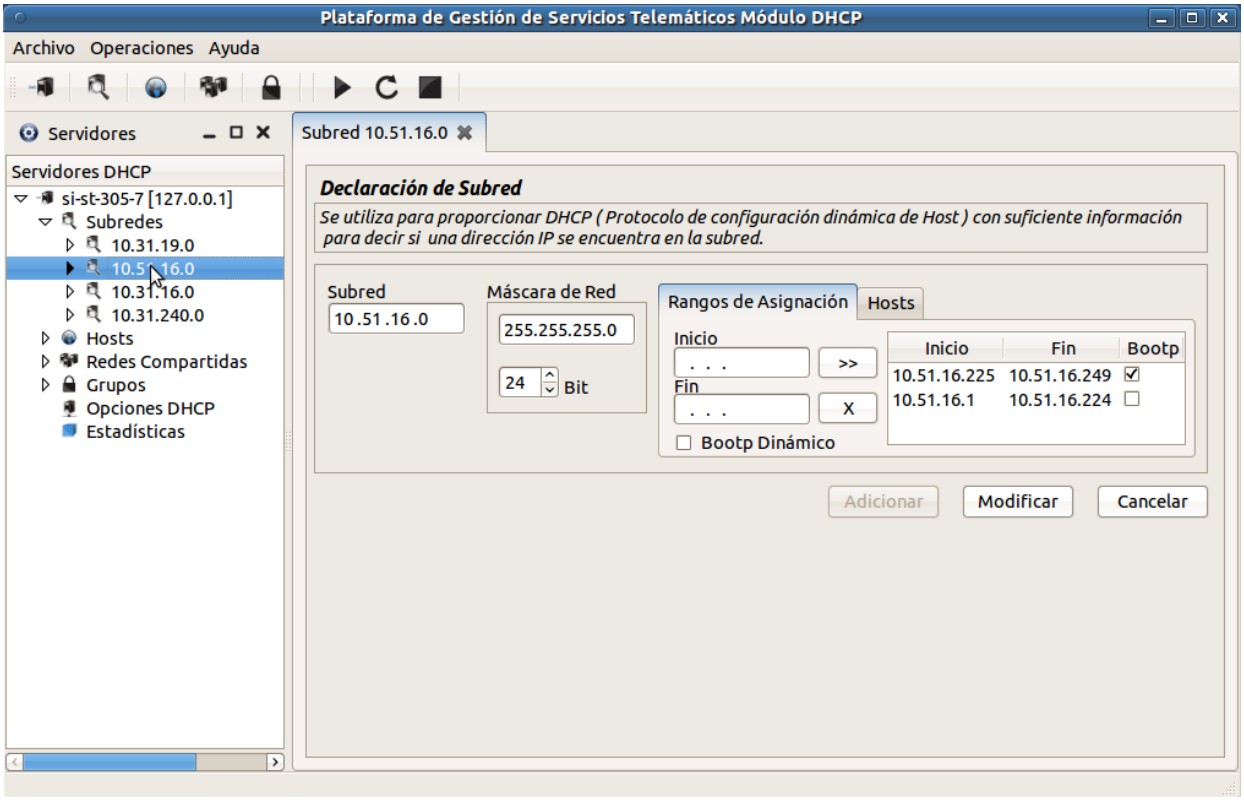


Figura 23. Módulo DHCP

Este módulo también permite al administrador conocer estadísticamente las asignaciones de direcciones IP por subred, que ha realizado el DHCP de ISC. Otras de las opciones que brinda este módulo es importar y exportar las configuraciones del DHCP de ISC con el objetivo de realizar salvas de seguridad. Para importar los datos de un DHCP de Windows Server en su versión 2003 el módulo utiliza una utilidad desarrollada que debe ejecutarse en el Windows Server la cual permite extraer los datos de las subredes definidas en este DHCP.

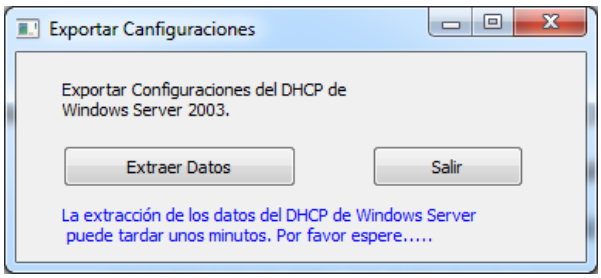


Figura 23. Utilidad para extraer datos del DHCP de Windows Server 2003.

Esta pequeña herramienta extrae la información de las subredes del DHCP de Windows Server 2003 a través del comando *netsh*, genera un archivo de configuración del DHCP de ISC y ejecuta una ventana de diálogo para salvar el fichero generado, listo para ser importado al DHCP de ISC a través de la funcionalidad de importar configuración del módulo.

Módulo DNS

El módulo DNS permite la gestión de los parámetros de configuración del servidor Bind de ISC en su versión 9, facilita la gestión de las zonas del Bind, sus parámetros y todos los tipos posibles de registros de la zona, permite además la definición de vistas, reenviadores, servidores externos y listas de control de acceso. Toda esta información se muestra en forma de árbol en la parte lateral izquierda del módulo. El módulo permite activar las extensiones de seguridad DNSSEC proporcionando seguridad al Bind.

Otra de las funcionalidades que permite este módulo es la integración del servidor Bind con una base de datos, específicamente con Postgres, el módulo proporciona un asistente que paso a paso guía al desarrollador, esta integración se hace posible con Bind DLZ como se mencionó en el capítulo I, para realizarla el administrador debe tener preparado un servidor Postgres y credenciales de acceso al mismo. A continuación una imagen que muestra una vista del módulo DNS:

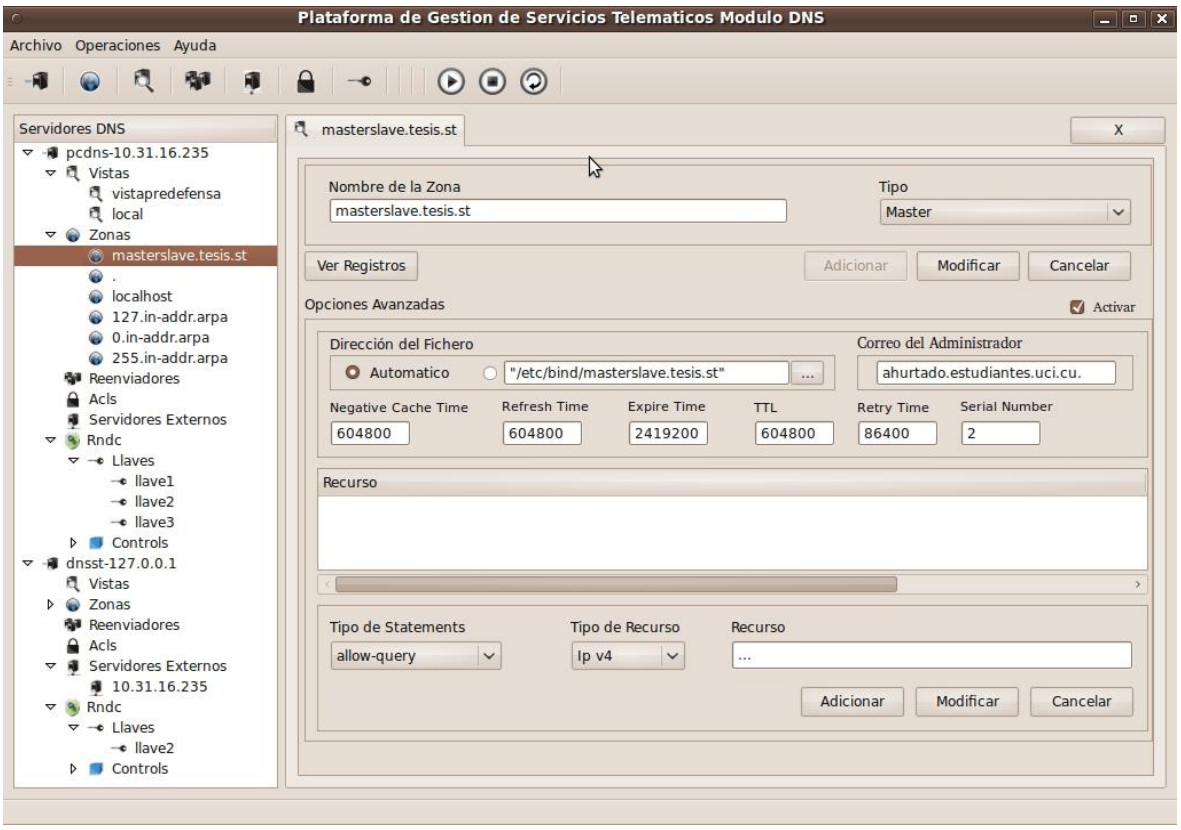


Figura 23. Módulo DNS

El módulo también permite importar los datos de un Windows Server 2003, el módulo se conecta al sistema operativo Windows Server 2003 con usuario y contraseña de administración a través del protocolo SMB[19] y del directorio Windows/system32/dns extrae los ficheros de zona los cuales poseen la misma sintaxis del Bind.

Módulo Directorio

El módulo Directorio permite la gestión del servidor Openldap en su versión 2.3. Este módulo en su parte lateral izquierda muestra el Árbol de Directorio del servidor OpenLdap con todas las entradas del mismo. Es común que este árbol de directorio sea extenso por lo que, la herramienta carga este árbol por niveles y de manera simultánea aumentando el rendimiento y rapidez de la aplicación. La herramienta permite realizar las tareas comunes en la gestión de un servicio de directorio, como por ejemplo la gestión de entradas.

[19] SMB: Server Message Block

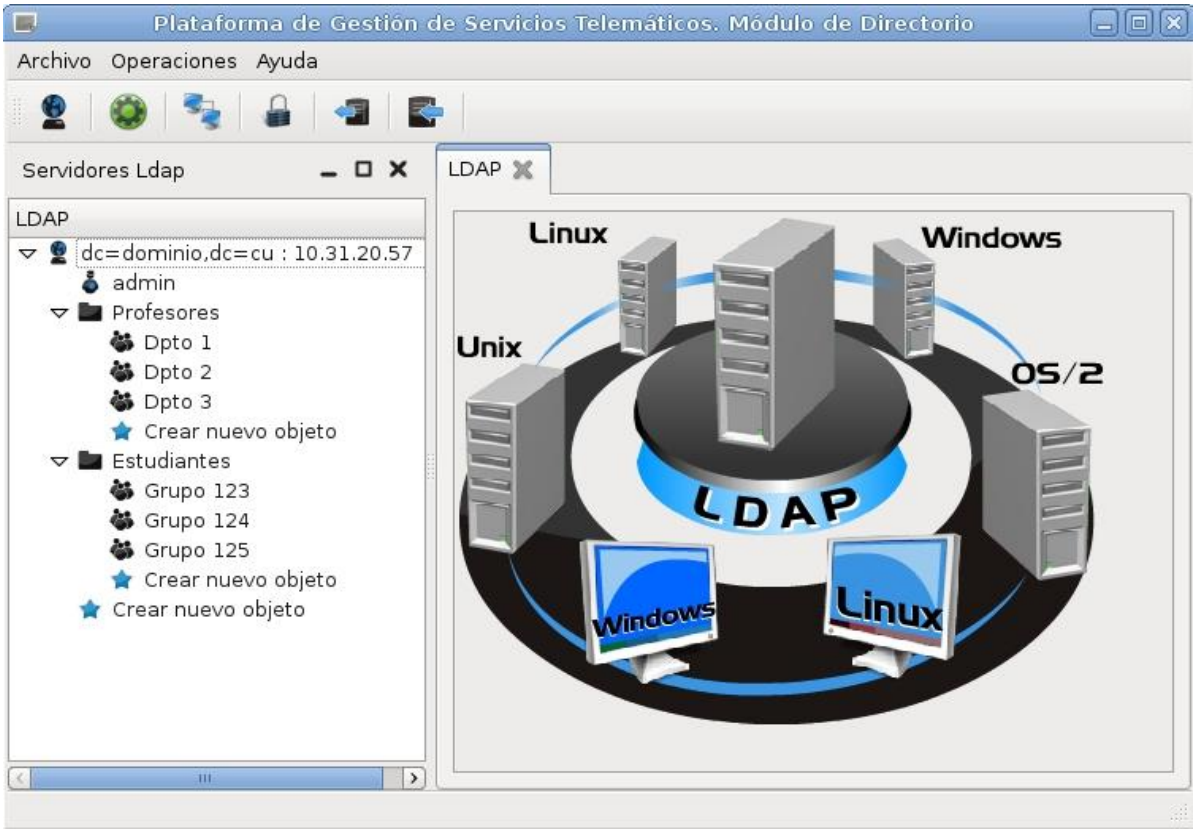


Figura 24. Módulo Directorio

Las entradas que permite gestionar el módulo son las cuentas de usuario, cuentas de correo, grupos, unidad organizativa, rol, computadora, entre otros. Dentro de esta adición de entradas permite la incorporación de imágenes a las entradas que lo permiten como es el caso de las cuentas de usuario. Para cada entrada el módulo de manera dinámica carga un formulario que permite la gestión de la entrada.

Una de las funcionalidades más importantes de este módulo es la gestión de esquemas, plantillas y tipos de entradas lo cual permite personalizar la información contenida en el directorio y la integración con otros servicios como se explica en el capítulo I de la tesis.

Este módulo permite también exportar e importar información del directorio en formato LDIF lo que permite realizar salvas de seguridad de la información del directorio. Otras de las funcionalidades que tiene el módulo es la realización de búsquedas en el árbol de directorio así como la migración de los datos de las entradas un servidor de Directorio Activo, no las políticas de grupo, contribuyendo al proceso de migración a software libre.

Módulo Integrador

El módulo integrador es un módulo que complementa el trabajo del resto de los módulos y los agrupa en un solo punto generalizando las integraciones de los mismos. Este módulo permite instalar cada uno de los servicios mencionados, Directorio, DNS y DHCP, local y remotamente en GNU/Linux, específicamente en Ubuntu y Debian.

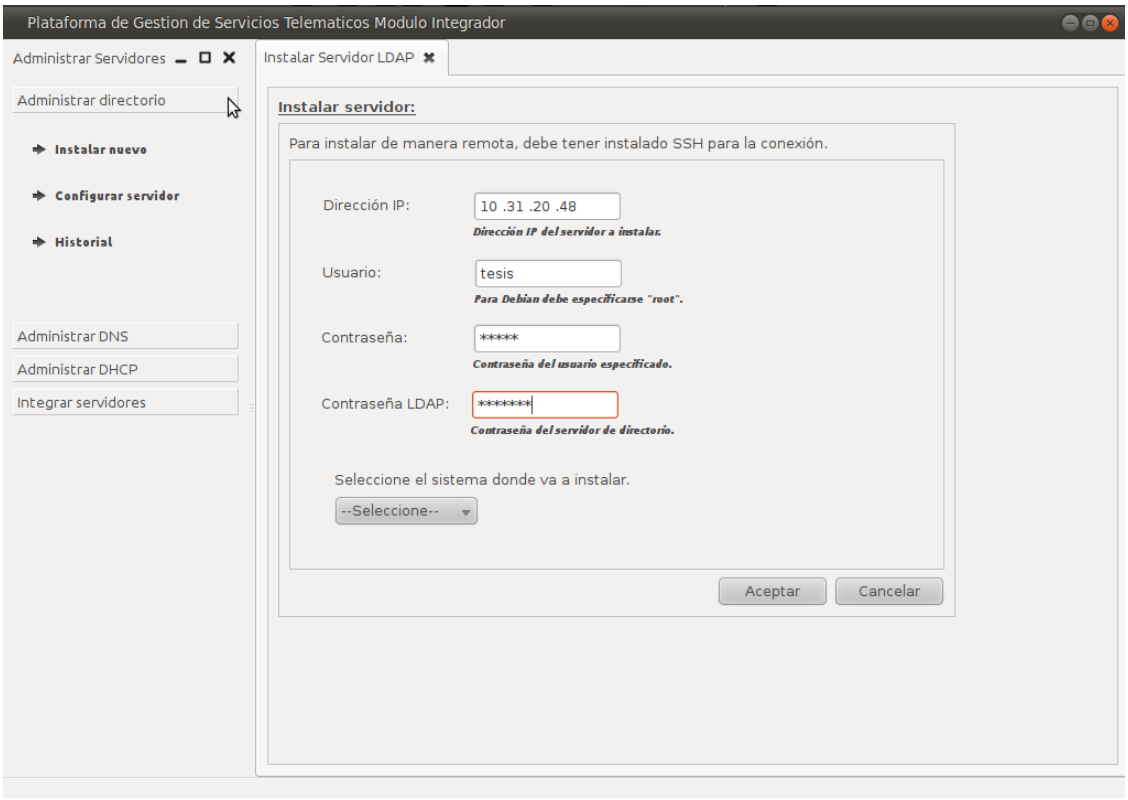


Figura 25. Módulo Integrador

Integrador facilita la integración de los servicios mencionados, permite integrar los servidores DNS y DHCP para facilitar la actualización dinámica de las zonas del Bind, permite también la integración del Bind con un servidor OpenLdap para guardar los registros del DNS en el servidor de Directorio aumentando el rendimiento del servidor DNS si es necesario. Para realizar cada una de estas integraciones el módulo se autentica en cada uno de los sistemas operativos de los servicios por SSH como lo realizan los módulos. Integrador permite además ejecutar los módulos DNS, DHCP y Directorio directamente desde su interfaz.

3.5 Conclusiones

Las herramientas informáticas de gestión e integración de servicios telemáticos de infraestructura facilitan el trabajo del administrador de red y en su desarrollo los marcos de trabajo juegan un rol importante disminuyendo el tiempo y costo de implementación. Para el desarrollo de herramientas de configuración e integración de servicios se deben tener en cuenta las bibliotecas de clases actuales que gestionan una parte del servicio, las cuales agrupadas en un marco de trabajo ofrecen un paquete completo de gestión y se reutiliza el código fuente existente sin necesidad de comenzar de cero ahorrando tiempo y esfuerzo.

Python como lenguaje de programación de alto nivel ofrece ventajas para el desarrollo de herramientas de gestión de servicios telemáticos de infraestructura en comparación con otros lenguajes, son varias las bibliotecas software libre y de código abierto que facilitan la gestión de los servicios telemáticos de infraestructura, lo cual constituye una ventaja con respecto a otros lenguajes.

La herramienta desarrollada facilita la gestión e integración de los servicios telemáticos de infraestructura DNS, DHCP y Directorio, está basada en una arquitectura modular y flexible que permite incorporar otros módulos y contribuye a la migración a software libre de los servicios de infraestructura.

CONCLUSIONES

Los servicios telemáticos de infraestructura son importantes en el desempeño de las redes de datos actuales las cuales ofrecen disímiles servicios para satisfacer necesidades de comunicación e información de los usuarios. Para ofrecer estos servicios se utiliza ampliamente el sistema operativo GNU/Linux por las ventajas que ofrece como software libre y de código abierto. En la gestión e integración de los servicios de infraestructura se deben utilizar herramientas informáticas adecuadas que permitan la correcta gestión de sus parámetros de configuración y facilite la integración de los mismos.

En la presente investigación se identificaron deficiencias en las herramientas actuales de gestión de configuración e integración de servicios de infraestructura alojados en GNU/Linux haciendo un análisis que muestra sus principales debilidades. A partir de las deficiencias descritas se realizó el diseño de un marco de trabajo ligero, orientado a objeto y modular que agrupa las bibliotecas de clases software libre y código abierto existentes especializadas en partes de la gestión de un determinado servicio de infraestructura reutilizando código fuente y esfuerzo, validando la hipótesis planteada y cumpliendo con los objetivos de la investigación. Este marco de trabajo ofrece un conjunto de componentes totalmente reutilizables en otros proyectos relacionados con la gestión e integración de los servicios telemáticos de infraestructura DNS, DHCP y Directorio.

A partir de este marco de trabajo se implementó una herramienta modular, con tecnología software libre y de código abierto, basada en una arquitectura multicapas facilitando de una manera simple e intuitiva la gestión e integración de los servicios DNS, DHCP y Directorio, disminuyendo las deficiencias presentes en las herramientas actuales. La arquitectura definida permite además la incorporación de otros módulos para la gestión de otros servicios telemáticos de infraestructura.

RECOMENDACIONES

Basado en el proceso de migración que paulatinamente lleva a cabo el país, paso importante en lograr una soberanía tecnológica, se propone desplegar la herramienta desarrollada en varias redes locales con el objetivo de realizar las pruebas necesarias en un entorno controlado y comenzar a utilizar esta herramienta que proporciona al administrador de red una aplicación eficaz para la gestión e integración de los servicios telemáticos de infraestructura. El proceso de despliegue de esta herramienta puede iniciar por varias universidades del país o por algún centro o institución que posea una red de datos.

Incorporar al diseño del marco de trabajo la gestión de otros servicios telemáticos no solamente los de infraestructura. El sistema operativo GNU/Linux es ampliamente utilizado para ofrecer servicios telemáticos y estos al igual que los servicios de infraestructura poseen configuraciones y se integran con el resto de los servicios por tanto, componentes para su gestión puede ser incorporados al marco de trabajo diseñado.

A partir de la incorporación al marco de trabajo de los componentes para la gestión de otros servicios telemáticos desarrollar los módulos correspondientes a la herramienta de gestión e integración de servicios.

Se recomienda también a partir de las integraciones actuales definidas entre los servicios telemáticos de infraestructura trabajar en la definición de un modelo único de integración. La mayoría de las integraciones de los servicios telemáticos son integraciones específicas y no están basadas en un modelo o estándar que permita su integración sin depender del proveedor de la implementación del servicio, cada servicio telemático ofrece vías de integración independientes y no de manera integrada con el resto de los servicios.

REFERENCIAS BIBLIOGRÁFICAS

[1] Dionisis, A., *Object-Oriented Development of Telematic Services.* IEEE Symposium on Computers and Communications, 1998.

[2] Peterson, L.L., *Computer Networks a systems approach*. 5th ed. 2011.

[3] Bosch, Jan y otros.“Object-Oriented Frameworks - Problems & Experiences ”.1997, 2012.

[4] Fayad, Mohamed E. y otros. “Object-Oriented Application Frameworks”. Communications of the ACM, 1997, 2012.

[5] Badías Ibarra, L. Kabir , “Propuesta de Procedimiento de Evaluación de los Frameworks. Un enfoque práctico. ”, Repositorio Institucional de la Universidad de las Ciencias Informáticas, UCI, julio 2007.

[6] Johnson, Ralph E. y otros. “Designing Reusable Classes”. Journal of Object-Oriented Programming, 1988.

[7] Shtull-Trauring , Itamar, “An Introduction to the Twisted Networking Framework ”. 2004. Disponible en http://tim.oreilly.com/pub/a/python/2004/01/15/twisted_intro.html , 2012.

[8] “Overview of ACE”. Disponible en http://www.cs.wustl.edu/~schmidt/ACE-overview.html, 2012.

[9] “JNDI Overview”. Documentación en Sitio Web de Oracle. Disponible en http://docs.oracle.com/javase/jndi/tutorial/getStarted/overview/index.html, 2012.

[10] “Internet Systems Consortium, Inc”. Disponible en: http://www.isc.org , 2012.

[11] “Introduction to OpenLDAP Directory Services.” Disponible en: http://www.openldap.org/doc/admin24/intro.html, 2012.

[12] “The LDAP Data Interchange Format (LDIF) - Technical Specification.” Disponible en: http://tools.ietf.org/html/rfc2849, 2012.

[13] “Directory Services Markup Language (DSML)”. Disponible en: http://xml.coverpages.org/dsml.html, 2012.

[14] Zeilenga, N.W.G.K. “Request for Comments: 4512. Lightweight Directory Access Protocol (LDAP): Directory Information Models.”, 2006.

[15] VASSILIKI KOUTSONIKOLA, A.V., “A Clustering-Driven LDAP Framework”. 2011. p. 34.

[16] VASSILIKI KOUTSONIKOLA, A.V., “A Structure-Based Clustering on LDAP Directory Information”.

[17] García, M.A.P., “Impacto del ENUM en las redes y los servicios”, 2008.

[18] “Bind DLZ, Dynamically Loadable Zones. Why is DLZ needed?” Disponible en: http://bind-dlz.sourceforge.net/, 2012.

[19] “Bind DLZ. Best Practices”. Disponible en: http://bind-dlz.sourceforge.net/best_practices.html

[20] Mejía, L.M.A. “Sistema Operativo GNU. La Definición de Software Libre.”, 2001, Disponible en: http://www.gnu.org/philosophy/free-sw.es.html , 2012.

[21] “Samba4/HOWTO”. Disponible en: http://wiki.samba.org/index.php/Samba4/HOWTO, 2012.

[22] “Webmin”. Disponible en http://www.webmin.com/ , 2012.

[23] “Zentyal 3.0 Documentación Oficial”. Disponible en: http://doc.zentyal.org/es/, 2012.

[24] “Freeipa”. Disponible en http://freeipa.org/page/Main_Page , 2012.

[25] “Python-ldaptor: Pure-Python LDAP Library”. Disponible en http://www.python.org/pyvault/SRPMS/repodata/repoview/python-ldaptor-0-0.0.31-1.html., 2012

[26] “Documentación Online de python-ldap”. Disponible en http://www.python-ldap.org/docs.shtml , 2012.

[27] “Java LDAP Overview.”, Disponible en: http://www.openldap.org/jldap/overview.html, 2012.

[28] “Sitio oficial de dnspython. A DNS toolkit for Python”. Disponible en http://www.dnspython.org/ , 2012.

[29] “Sitio oficial de PyDhcpLib”. Disponible en: http://pydhcplib.tuxfamily.org/pmwiki/, 2012.

[30] Kinder, Ken, “Event-Driven Programming with Twisted and Python”. Linux Journal, 2012.

[31] Stroustrup, B., “What Is 'Object-Oriented Programming”. 1991, New Jersey AT&T Bell Laboratories Murray Hill.

[32] Booch, G., “Analisis y Diseño Orientado a Objetos con Aplicaciones”, 1996, Addison Wesley.

[34] Garlan, D., “An Introduction to Software Architecture”. School of Computer Science Carnegie Mellon University, 1994.

[35] Abowd, G., “Recommended Best Industrial Practice for Software Architecture Evaluation”. Software Engineering Institute Carnegie Mellon University Pittsburgh, Pennsylvania 15213.

[35] Booch., G., “El Lenguaje Unificado de Modelado”. 1999: Addison Wesley Iberoamericana.

[36] Larman, C., “UML y Patrones. Introducción al análisis y diseño orientado a objetos”. 1999: Prentice Hall Hispanoamericana.

[37] Gamma, E., “Design Patterns. Elements of Reusable Object-Oriented Software”. 1995: Addison Wesley.

[38] Pressman, R.S., “Ingeniería del Software. Un enfoque práctico”. 2002: Mc Graw-Hill/ Interamericana de España, S. A

[39] Johnson, R., “Expert One-on-One J2EE Development without EJB”. 2004: Wiley Publishing, Inc.

[40] Johnson, R., “The Spring Framework - Reference Documentation”. 2004.

[41] Stallings, W., “OPERATING SYSTEMS. Second edition”. 1997: PRENTICE HALL, INC.- Simón & Schuster Intemational Group.

[42] Garg, R., “Techniques for Optimizing Applications - High Performance Computing”. 2002: Prentice-Hall

[43] Bauer, M., “syslog Configuration”. Linux Journal, 2001

[44] Lutz, M., “Learning Python. THIRD EDITION” ed. 2008: O’Reilly Media

[45] “PyPI - the Python Package Index”. Disponible en: http://pypi.python.org/pypi, 2012

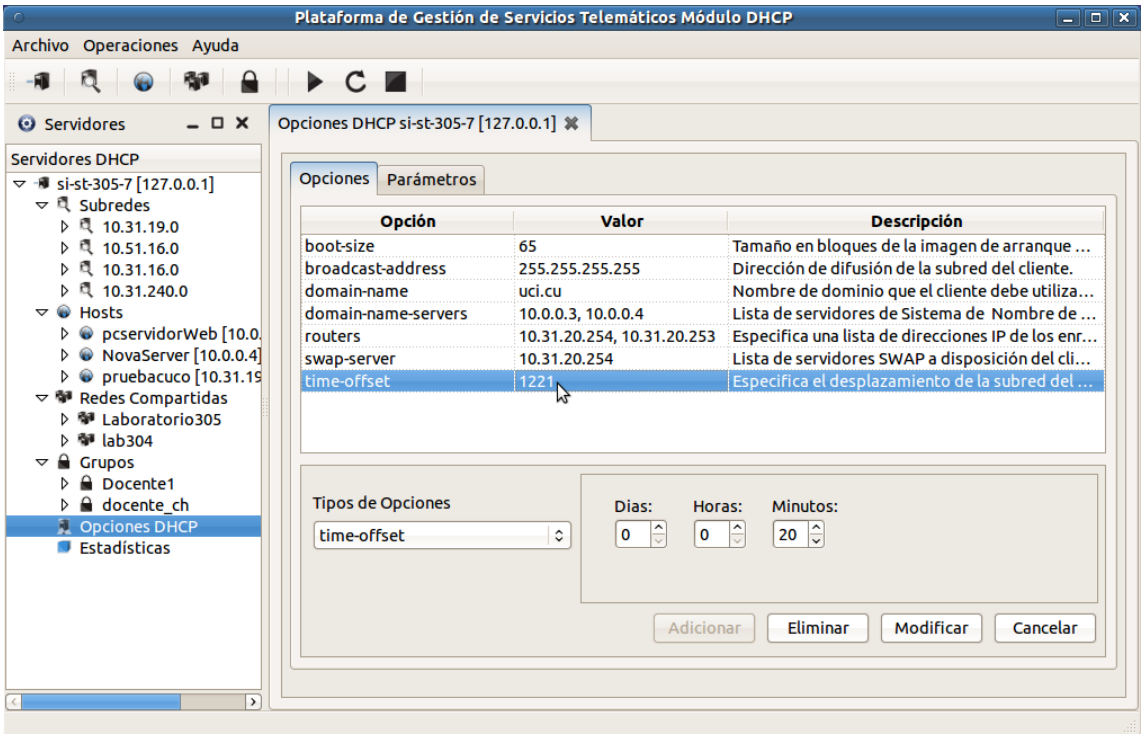
[46] “Python Programming Language – Official Website”. Disponible en: http://python.org/., 2012.

GLOSARIO DE TÉRMINOS

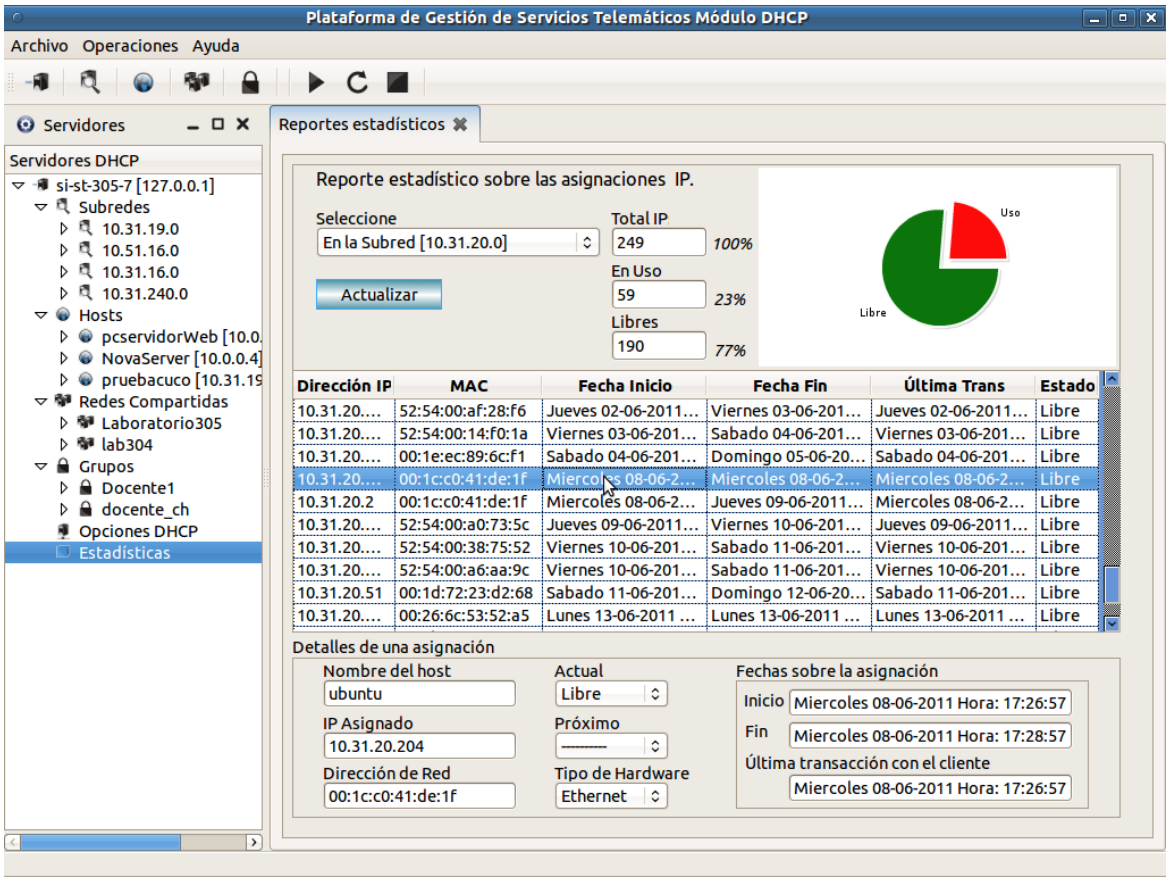
API	Application Programing Interface
CORBA	Common Object Request Broker Architecture
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DSML	Directory Services Markup Language
FTP	File Transfer Protocol
GRASP	General Responsability Asignment Software Paterns
HTTP	Hypertext Transfer Protocol
IoC	Invertion of Control
IP	Internet Protocol
ISC	Internet Systems Consortium
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Data Interchange Format
RFC	Request for Comments
RMI	Remote Method Invocation
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
SSL	Security Socket Layer
TIC	Tecnologías de la Información y Comunicaciones
XML	eXtensible Marcup Language

ANEXOS

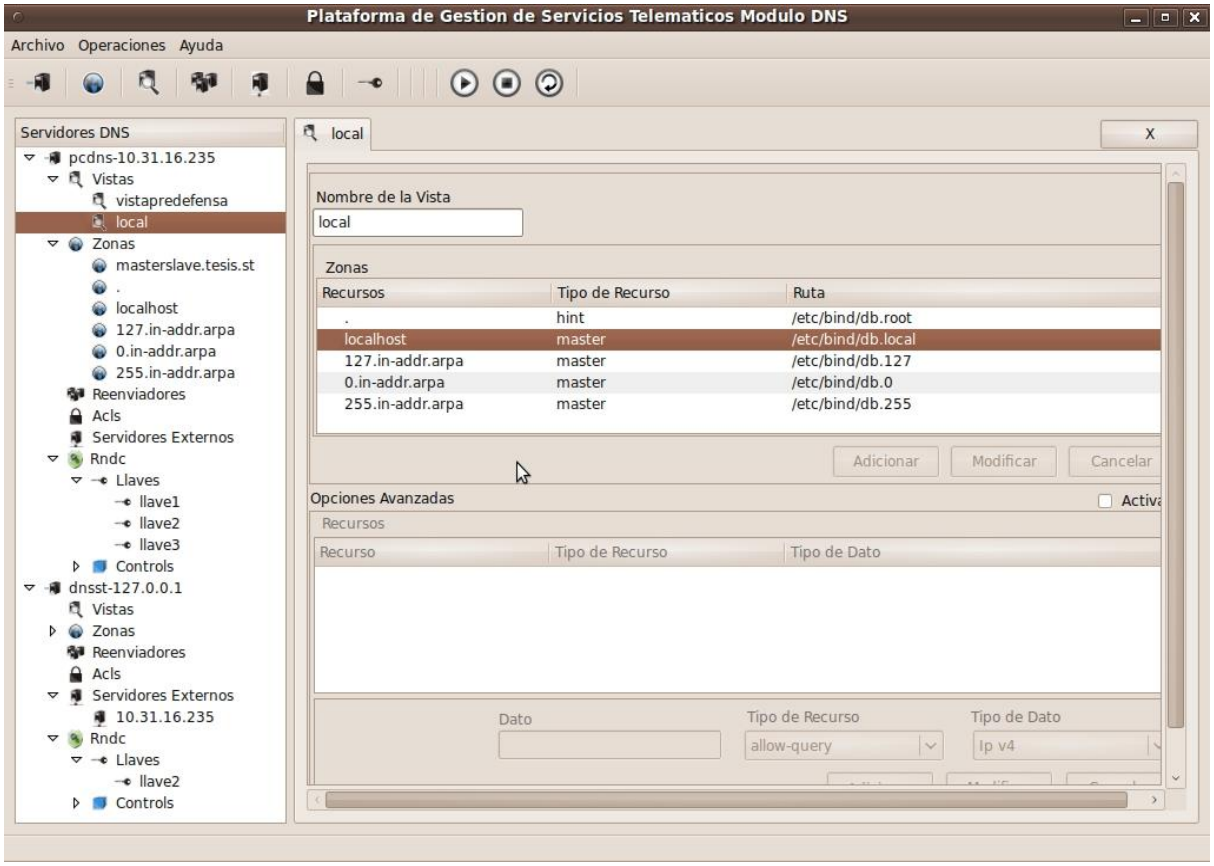
Anexo 1. Vista del Módulo DHCP para la gestión de varias opciones del DHCP de ISC.



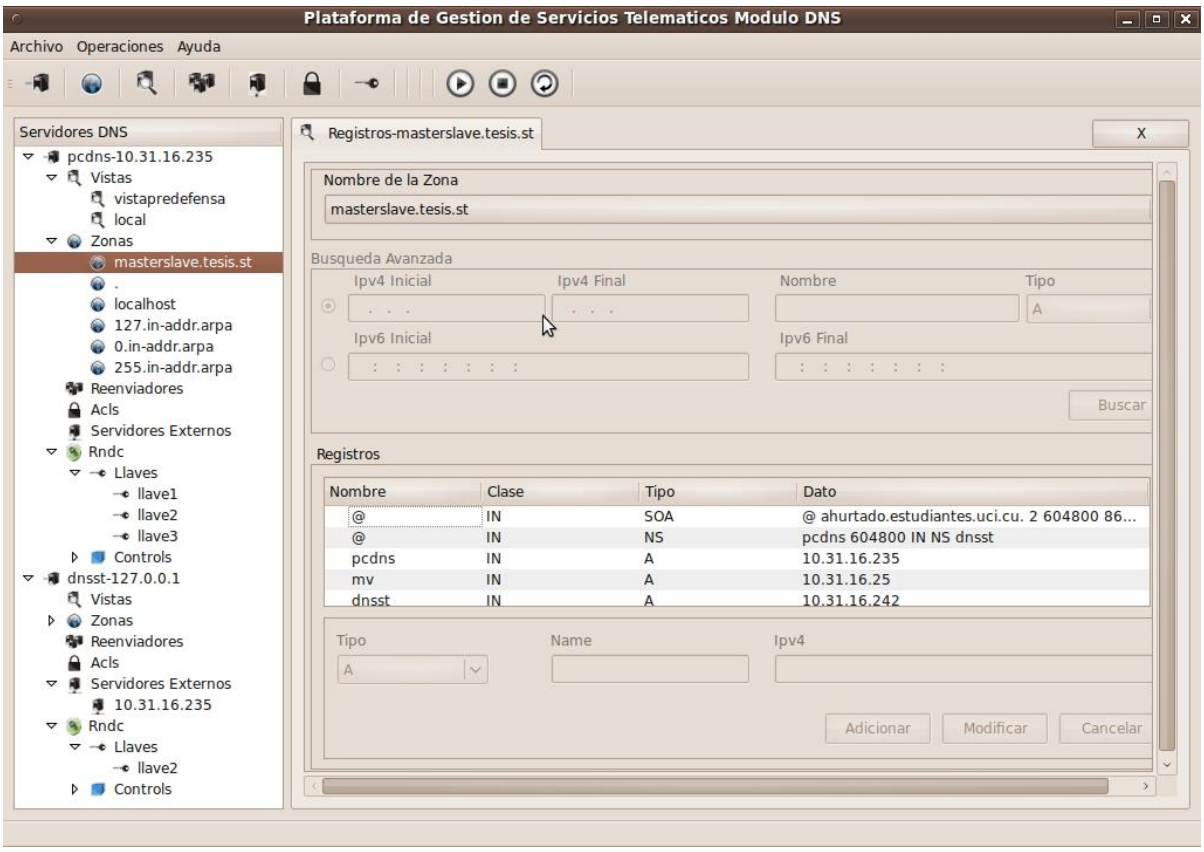
Anexo 2. Vista del Módulo DHCP para los reportes estadísticos de las asignaciones.



Anexo 3. Vista Módulo DNS para mostrar una vista del Bind y sus zonas.

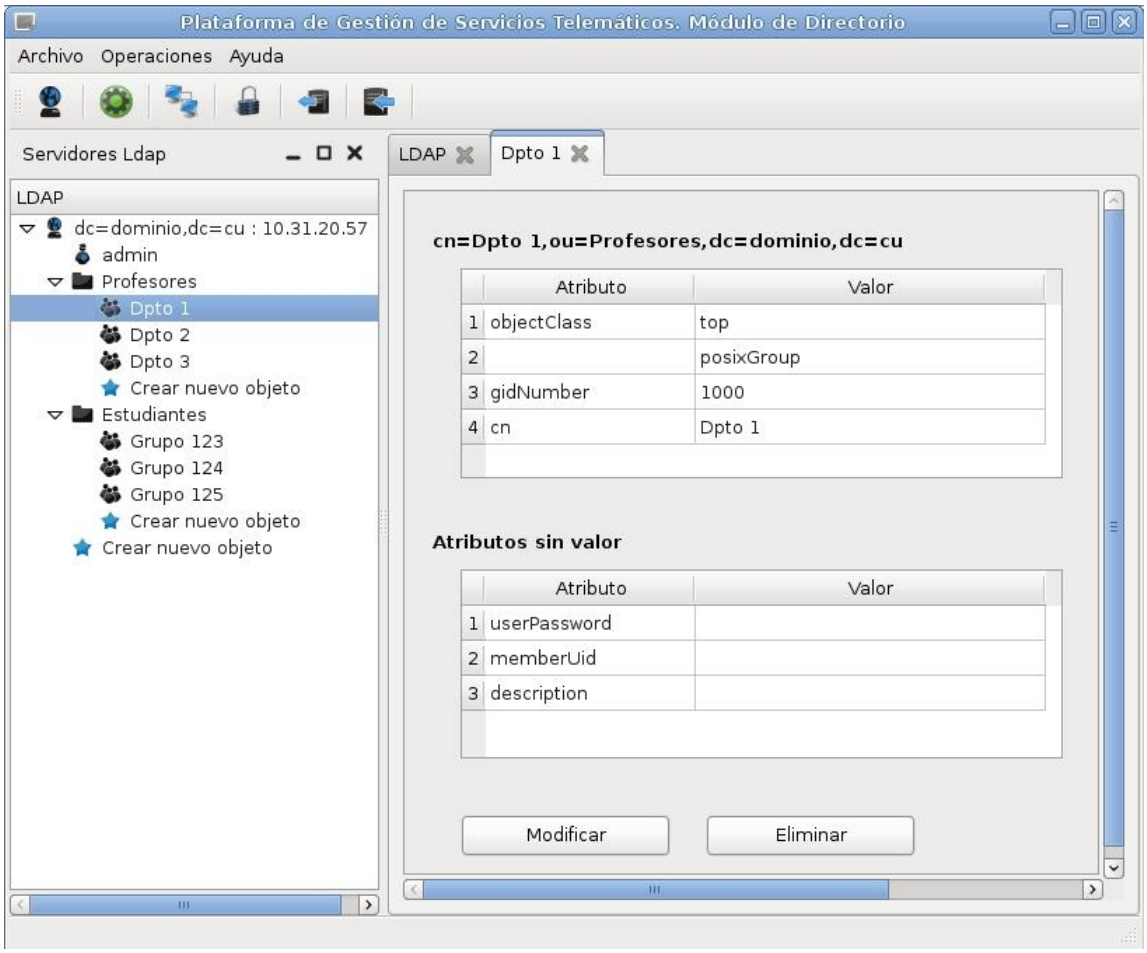


Anexo 4. Vista Módulo DNS para mostrar los registros de una zona

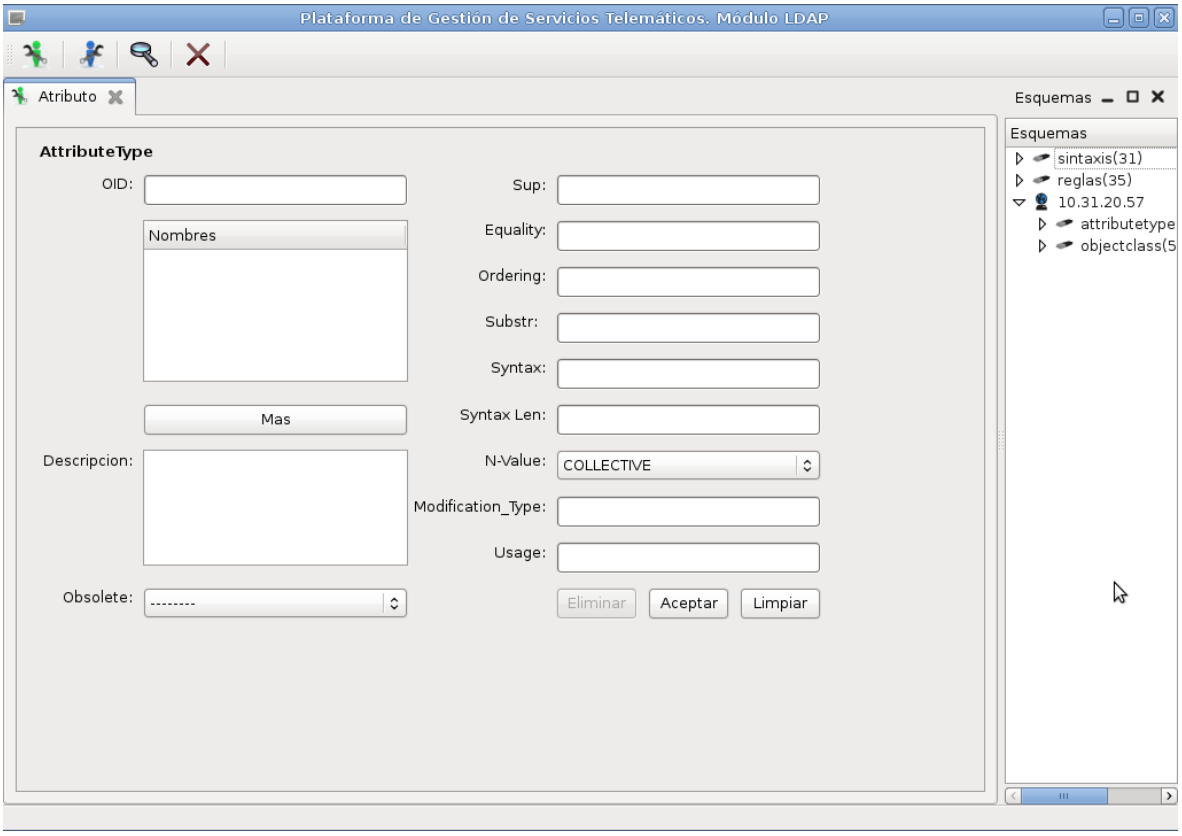


Anexo 5. Asistente de Compilación Bind para integración con Postgres.

Anexo 6. Vista Módulo Directorio para mostrar los datos de una entrada.



Anexo 7. Vista Módulo Directorio para gestionar un atributo de los esquemas.



Anexo 8. Vista Módulo Directorio para gestionar clases de objetos en los esquemas.