ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DESARROLLO DE APLICATIVOS PARA GESTIÓN AUTOMÁTICA DE UNA RED BASADO EN PROTOCOLOS DE GESTIÓN.

DESARROLLO DE UN PROTOTIPO SOFTWARE (TIPO NMS) BASADO EN RESTCONF

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN

GUSTAVO XAVIER ACONDA TORRES

gustavo.aconda@epn.edu.ec

XAVIER ALEXANDER CALDERÓN HINOJOSA

xavier.calderon@epn.edu.ec

DMQ, AGOSTO 2023

CERTIFICACIONES

Yo, Gustavo Xavier Aconda Torres declaro que el trabajo de integración curricular aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

_____

Gustavo Xavier Aconda Torres

Certifico que el presente trabajo de integración curricular fue desarrollado por Gustavo Xavier Aconda Torres, bajo mi supervisión.

_____

Xavier Alexander Calderón Hinojosa

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el producto resultante del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Gustavo Xavier Aconda Torres

Xavier Alexander Calderón Hinojosa

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

Capítulo 1

Capítulo 2

Capítulo 3

ÍNDICE DE TABLAS

Capítulo 1

Capítulo 2

ÍNDICE DE CÓDIGOS

Capítulo 2

RESUMEN

En el presente documento se describe la implementación de una red que utiliza el protocolo RESTCONF para la administración de la red, esta estructura se montó con la ayuda de programas de emulación de redes y máquinas virtuales, para finalmente ser implementado con equipos físicos (CISCO C1111-4P Cisco 1100 Series y CISCO 4221 Integrated Services Routers).

Para las pruebas de funcionamiento de la NMS se utiliza el emulador GNS3 el cual permite implementar equipos de red como routers, switcheses, etc. Se utilizó VirtualBox y VMWARE para crear máquinas virtuales las cuales se utilizarán como clientes, esto con la finalidad de poder realizar pruebas de funcionamiento para su posterior implementación en los equipos CISCO. Además, se describen los protocolos de red complementarios como DHCP, NAT, OSPF entre otros. Esto para dar cumplimiento a los requerimientos de la NMS, la cual debe contar con 3 módulos (configuración, información y reportes), por último, el despliegue de la NMS se realiza mediante un servidor web alojado localmente para la construcción del código se utiliza la metodología Spring, la cual permite un desarrollo continuo de mejoras para la optimización del código, con lo cual se puede garantizar un crecimiento posterior del aplicativo.

La NMS se construirá en base a las bibliotecas del modelo YANG, por lo cual se utiliza Python como lenguaje base para esta tarea, el motivo es que las bibliotecas disponibles que permiten realizar consultas de tipo REST de forma dinámica, pues se adapta a las necesidades del programador, facilitando el crecimiento y la implementación de nuevos módulos de consulta y métodos de configuración.

Palabras Claves: NMS, YANG, RESTCONF, Python, GNS3, módulos, REST

ABSTRACT

This document describes the implementation of a network that uses the RESTCONF protocol for network administration, this structure was set up with the help of network emulation programs and virtual machines, to finally be implemented with physical equipment (CISCO C1111 -4P Cisco 1100 Series and CISCO 4221 Integrated Services Routers).

For the development of the NMS prototype, the GNS3 emulator is used, which allows the implementation of network equipment such as routers, switches, etc. VirtualBox and VMWARE were used to create virtual machines which will be used as clients, this in order to be able to carry out functional tests for their subsequent implementation in CISCO equipment. In addition, complementary network protocols such as DHCP, NAT, OSPF, among others, are described. This to comply with the requirements of the NMS, which must have 3 modules (configuration, information and reports), finally, the development of the NMS is done through a locally hosted web server for the construction of the code, the Spring methodology, which allows a continuous development of improvements for the optimization of the code, with which a subsequent growth of the application can be guaranteed.

The NMS will be built based on the libraries of the YANG model, for which Python is used as the base language for this task, the reason is the available libraries that allow REST-type queries to be made dynamically, as it adapts to the needs of the programmer, facilitating the growth and implementation of new query modules and configuration methods.

Keywords: NMS, YANG, RESTCONF, Python, GNS3, modules, REST

1. INTRODUCCIÓN

1.1. Objetivo general

Desarrollar una NMS basada en el uso del protocolo RESTCONF, mediante su implementación en equipos de red físico como los modelos Cisco C1111 y Cisco 4200, para entender cómo se puede monitorizar la red de forma remota.

1.2. Objetivos específicos

- Implementar el modelo de red en GNS3, para su posterior desarrollo con equipos físicos como El Cisco C1111 y Cisco 4200
- Desarrollar la NMS con el uso de Python y la Api de RESTCONF
- Evaluar el rendimiento de la configuración remota de equipos de red

1.3. Alcance

El proyecto define entender cómo funciona el protocolo RESTCONF sus utilidad e implementación en los equipos de red de características IOS-XE, todo con la finalidad de poder implementar un NMS capas de poder trabajar con las funciones de la API disponible en este protocolo.

Se estima que la implementación será una red simulada mediante el software GNS3 o a su vez con equipos Cisco Modelos C1111y 4200, los cuales permiten un trabajo sincronizado es decir mediante el simulador implementaremos un red virtual la cual posteriormente será desarrollada en los equipos Cisco mencionados anteriormente esto con el objetivo de visualizar el funcionamiento de la NMS en un ambiente controlado de pruebas y motivar con esto un desarrollo a futuro utilizando este software desarrollo para la monitorización de la red.

La primera fase del trabajo será el desarrollo de la red, configurando los equipos y asignando las direcciones IP de todos los equipos relacionados.

La segunda fase permitirá implementar los servicios del protocolo RESTCONF en el Router para poder comunicarnos desde la NMS

La tercera fase será la implementación la NMS con la API mediante el lenguaje de programación Python, se estimó un desarrollo amigable con una interfaz gráfica intuitiva para su uso.

1.4. Marco teórico

1.4.1. RESTCONF [1]

Es un protocolo que utiliza como base HTTPS y el lenguaje YANG, que es usado para definir la sintaxis de la estructura. La estructurada de datos XML, JSON y YANG provienen de API's del estilo REST. Los comandos HTTP como GET, POTS, PUT, PATCH y DELETE son redireccionadas a la API para acceder a los recursos de data representado por el modelo YANG, en la Figura 1. 1 se describe el funcionamiento de RESTCONF, donde el nivel más bajo utiliza el modelo YANG y sus bibliotecas, lo siguiente será definir como se trasporta la información, en el caso del protocolo con el que se trabaja es mediante HTTPS, lo siguiente es definir como se codifica la información en este caso se utilizan JSON, lo siguiente es el protocolo con el cual se desea trabajar para este desarrollo se utiliza RESTCONF, lo siguiente será seleccionar una (Interfaz e programación de aplicaciones) API capaz de soportar el uso del modelo YANG para realizar consultas, para el caso concreto de este trabajo utilizaremos bibliotecas de PYTHON las cuales permiten realizar consultas del tipo HTTP, además de que permite la codificación y decodificación JSON, la cual es útil para poder crear un cuerpo de mensaje con la capacidad de realizar cambios en los dispositivos CISCO en base a los requerimientos del usuario. Por último, se tiene la aplicación NMS la cual recoge todas las características mencionadas para que el uso de esta sea de la forma más transparente para el usuario, es decir no se preocupe de la escritura de las consultas sino de la información que debe proporcionar para realizar algún cambio en los dispositivos CISCO en este caso la conoceremos como NMS.

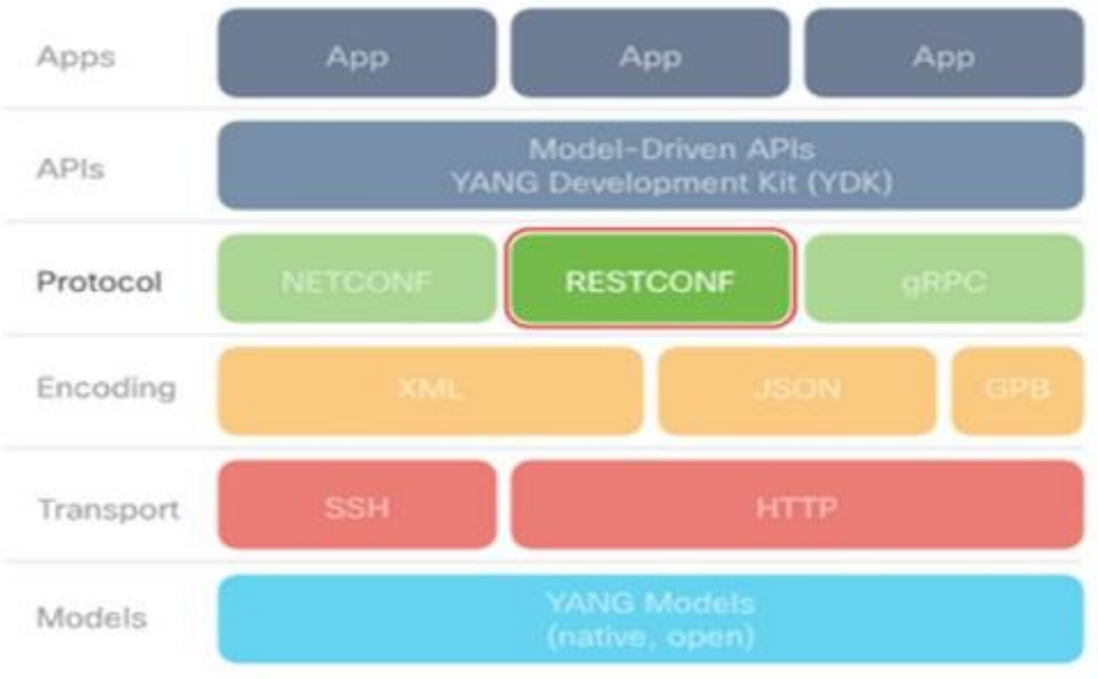

Figura 1. 1 Descripción del Modelo utilizado para la implementación de RESTCONF [1]

RESTCONF proporciona acceso a almacenes de datos específicos utilizando recursos de operación, sin embargo, estos no son obligatorios de acuerdo con el protocolo, esto por motivo de que cada recurso de operación determina el cómo accede a los almacenes de datos. En cuanto al formato JSON se debe aclarar que este tipo de notación es un formato abierto independiente del idioma, el cual es capaz de utilizar un texto legible por humanos para expresar objetos de datos, además, JSON a menudo envía los datos sin saltos de línea para ahorrar espacios.

1.4.2. Recursos de la API [2]

Contiene el recurso raíz de RESTCONF como el almacén de datos y recursos de operación como se aprecia en la Figura 1. 2.

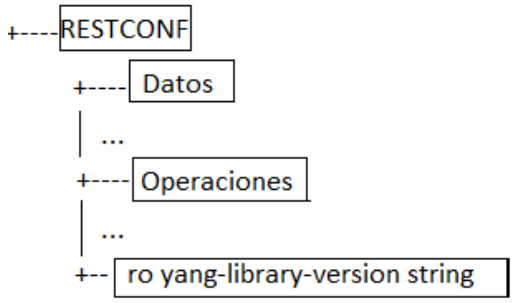

Figura 1. 2 Raíz de árboles YANG [2]

Del árbol de YANG podemos describir sus componentes de la siguiente manera:

Datos. – Recurso obligatorio el cual representa la configuración combinada y estado de los recursos de datos a los que puede acceder un cliente, no puede ser creado o eliminado por el cliente.

Operaciones. – Recurso opcional que proporciona acceso a las operaciones RPC específicas del modelo de datos admitidas por el servidor, el servidor puede omitir este recurso si no hay un RPC específico.

Versión-biblioteca-yang. – Identifica la fecha de revisión del módulo YANG implementado por el servidor.

1.4.3. Almacén de datos

Es una colección de datos de configuración y nodos de datos de estado, es un recurso de abstracción de datos, el cliente puede usarlo para editar y recuperar recursos de datos, como la raíz conceptual de toda la configuración y estado de datos presentes en el dispositivo.

1.4.4. REST Web API’s

Es una interfaz programable, la cual utiliza comunicación mediante HTTP mientras se adhiere a los principios de arquitectura REST.

Los conceptos del protocolo HTTP con los cuales trabaja el API son:

HTTP request/responses.

HTTP verbs.

HTTP status codes.

HTTP headers/body.

API's Request.

1.4.5. REST API Responses

Las respuestas HTTP comunican el resultado de la petición HTTP. La respuesta puede contener la data que fue requerida, El servidor que recibe la petición informa sus resultados, o podría informar un error en la petición de la data. Respuestas REST API son similares a los request, los cuales procesan tres factores:

- HTTP status.
- Header.
- Body.

1.4.6. Autenticación

La autenticación básica se utiliza para identificar a un usuario de la API RESTCONF. Las cuentas son las mismas que se utilizan para autenticarse en la interfaz de usuario del administrador. De manera predeterminada, las cuentas se administran en la base de datos de configuración (CDB), pero se pueden configurar para que sean datos admitidos externamente (p. ej., RDBMS, LDAP, etc.). Para usar esta funcionalidad, se debe configurar un administrador de credenciales. Cuando ese administrador de credenciales es uno que usa una fuente de datos LDAP (Protocolo ligero de acceso a datos), es posible devolver el conjunto de grupos en los que se encuentra el usuario. Esto debe hacerse incluyendo el atributo *memberOf* en la entrada LDAP del usuario autenticado. El nombre común (CN) de ese grupo es el valor que debe coincidir con el grupo de cualquier lista de reglas que se deba aplicar a ese usuario. Por ejemplo, si el atributo *memberOf* del usuario autenticado es *cn=my-good-group, o=example*, entonces el grupo será *my-good-group*.

También es posible autenticarse mediante *OAuth (autorización abierta)*. En tal caso, se deben configurar los clientes específicos que se pueden utilizar. Estos tienen ciertas restricciones que se aplicarán cuando se valide la configuración (por ejemplo, los ámbitos configurados, las audiencias, etc.). Si este acceso contiene un reclamo de grupo, estos valores se utilizarán para el control de acceso.

No es necesario definir ningún grupo proporcionado por el servidor LDAP o en el token de acceso *OAuth*. Sin embargo, la creación de reglas en la interfaz de

usuario requiere que se creen explícitamente. De lo contrario, la interfaz de usuario no los reconoce.

1.4.7. Consulta de datos

La API RESTCONF acepta varios parámetros en la cadena de consulta. Los parámetros permitidos y sus valores dependen del método HTTP que se utilice y del tipo de recurso. Los parámetros permitidos y una descripción del funcionamiento de las acciones que se pueden utilizar, todo esto se encuentra en el resumen de la Tabla 1. 1**¡Error! No se encuentra el origen de la referencia.** , para un mejor entendimiento de las posibles acciones a realizar con una solicitud HTTP:

Tabla 1. 1 Descripción de las opciones de RESTCONF

RESTCONF	Descripción
OPTIONS	Determina cuál método es soportado por el servidor.
GET	Recupera datos y metadatos sobre un recurso.
HEAD	Solo devuelve los encabezados de respuesta.
POST	Crea un recurso o invocar una operación RPC.
PUT	Crea o remplaza un recurso.
PATCH	Crea o actualiza (pero no elimina) varios recursos. Se admiten tanto el parche HTTP simple (RFC 7589) como el parche YANG (RFC 8072).
DELETE	Enviado por un cliente para eliminar un recurso de destino.

1.4.8. El lenguaje YANG

Es un lenguaje de modelado de datos extensible basado en estándares utilizado para modelar la configuración y los datos de estado operativo (como los comandos show), cuenta con las siguientes características:

• Humano legible y fácil de aprender.

• Modelos de datos de configuración jerárquicos.

• Tipos y agrupaciones reutilizables (tipos estructurados).

• Extensibilidad mediante aumento.

• Restricciones formales para la validación de la configuración.

• Modularidad de los datos a través de módulos y sub-módulos.

• Reglas de control de versiones bien definidas.

YANG cuenta con un conjunto de tipos de datos incorporados como los detallados a continuación:

String.
Binary.
Bits.
Boolean.
Empty.
Enumeration.
Identityref.
Instance-identifier
Int8, int16, int32, int64
Uint8, uint16, uint32, uint64.
Decimal64.
Leafret.
Unión.

1.4.9. Modelo de datos

Un modelo de datos es una descripción de cómo se deben codificar los datos para el intercambio de información entre dos entidades, se puede crear un modelo de datos para cualquier proceso de intercambio de información y se basa en pares clave-valor.

Por ejemplo, para representar una interfaz en un interruptor a través de un modelo de datos, el modelo podría estructurarse para incluir:

• Tipo: selección de una lista predefinida (Gigabit Ethernet IEEE 802.3z, FastEthernet IEEE 802.3, etc.)

• Descripción: cadena de texto de formato libre.

• Estado: habilitado/deshabilitado (booleano).

El lenguaje de modelado de datos YANG estándar impone una estructura definida para generar modelos de datos. YANG proporciona enfoques independientes del proveedor. Además, son estructuras del árbol jerárquico para la organización de datos.

1.4.10. Construcciones de módulos

Los módulos por lo general parten de uno o más contenedores, y el árbol de esquema se extiende a otro nivel de estructuras de datos. Una rama termina con hojas y/o listas de hojas.

Las principales afirmaciones en un módulo YANG son:

- Contenedor: Una agrupación de otras declaraciones (no tiene "Valor").
- Lista: Múltiples registros que contienen al menos una hoja "Clave" y una jerarquía arbitraria de otras declaraciones.
- Hoja: par único clave/valor.
- Leaf-list: Múltiples pares clave/valores del tipo.

1.4.11. Modelos de datos nativos

Los modelos de datos nativos de Cisco IOS XE específicos para IOS XE no son interoperables con otras plataformas. Reflejan de cerca la estructura de la

interfaz de línea de comandos (CLI) de IOS XE. El beneficio clave de los modelos de datos nativos es la amplitud de la cobertura de características.

Ejemplos incluyen:

• Cisco-IOS-XE-interfaces.yang: un módulo nativo para configurar los parámetros de la interfaz, como la dirección IP, la descripción y la velocidad. También contiene datos operativos sobre las interfaces, como el estado de administración.

• Cisco-IOS-XE-ospf.yang: un módulo nativo para configurar los procesos de enrutamiento OSPF (Open Shortest Path First).

1.4.12. Modelos de datos abiertos

Los modelos de datos abiertos proporcionan una interfaz común en múltiples plataformas. Cisco IOS XE es compatible con una cantidad de modelos de datos abiertos de los organismos de estándares IETF y OpenConfig.

1.4.13. Configuración RESTCONF

Para poder configurar RESTCONF en cualquier dispositivo de red como router o switches, lo principal es saber previamente si estos cuentan con bibliotecas YANG y generan los certificados Http y Https mediante sus archivos en la NVRAM del dispositivo a configurar. Para poder continuar con el proceso de configuración se realizan los pasos descritos en las Figura 1. 3 y Figura 1. 4.

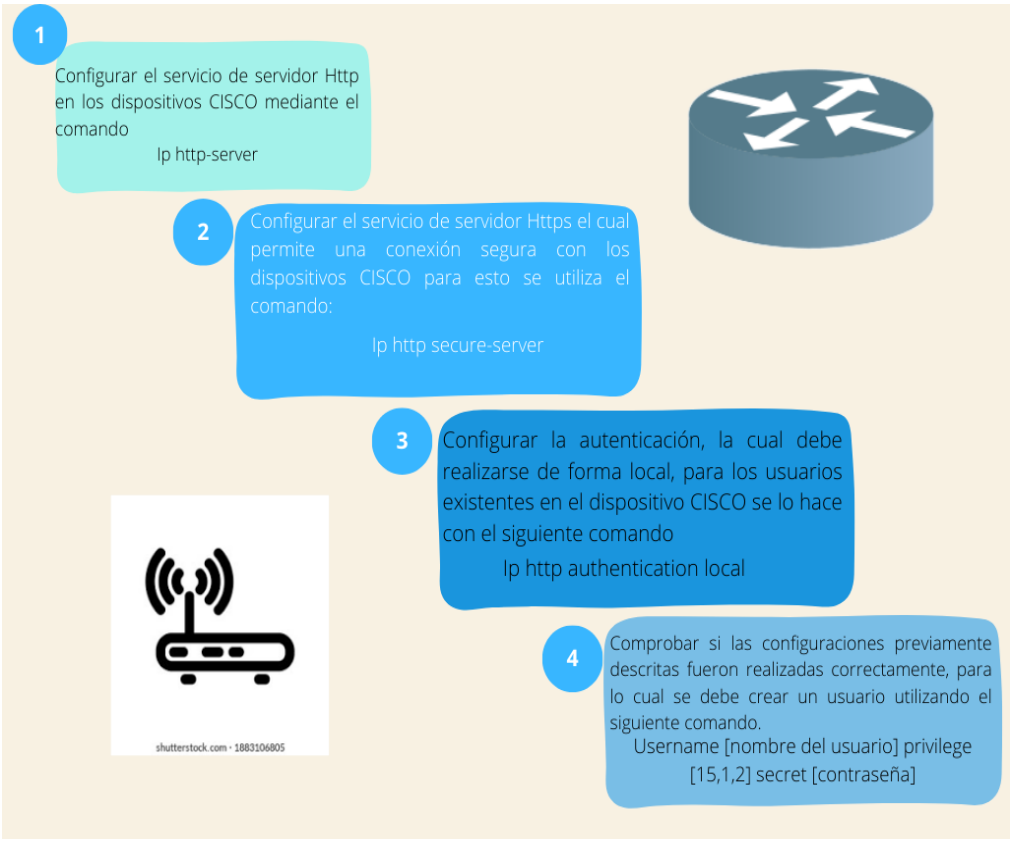

Figura 1. 3 Pasos para la configuración RESTCONF parte 1

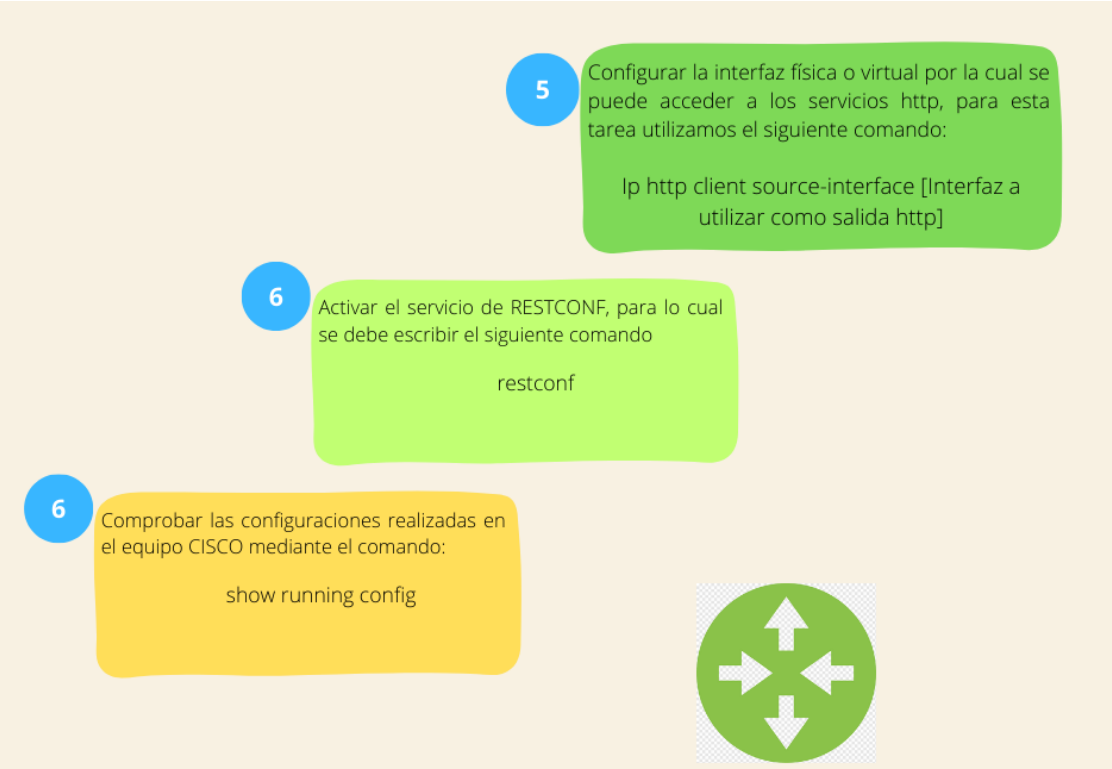

Figura 1. 4 Pasos para la configuración RESTCONF parte 2

En la Figura 1. 3, se aprecia en el paso 4 los niveles de seguridad con los que cuenta un usuario, esto con la finalidad de poder limitar el poder administrativo de los usuarios creados para lo cual se detalla a continuación los niveles que permiten los dispositivos CISCO para realizar estas tareas:

- 15 acceso completo a todos los comandos, para realizar cambios en la configuración.
- 1 solo lectura y acceso a comandos limitados.
- 2 solo lectura y la capacidad de hacer ping.

Una vez realizado todos los pasos descritos previamente es importante recordar que no se debe utilizar la configuración web esto con la finalidad de evitar sobrescribir las configuraciones ya realizadas, una herramienta para revisar si con todo lo hecho previamente, es posible acceder a los servicios de RESTCONF es mediante POSTMAN la cual cuanta con bibliotecas que permiten realizar pruebas.

1.4.14. CISCO C1111-4P - Cisco 1100 Series Integrated Services Routers

Para la implementación de la red este dispositivo será de importante utilidad porque soporta las características necesarias para el correcto funcionamiento del protocolo, por lo cual, se describen las características de este dispositivo en la Tabla 1. 2, donde se puede recalcar que este dispositivo soporta conexiones inalámbricas, permitiendo una administración Wireless.

Tabla 1. 2 Descripción del equipo CISCO C1111-4P

Red	
Ethernet	Si
Ethernet LAN, velocidad de transferencia de datos	10,100,100 Mbits/s
Tecnología de cableado	10/100/1000Base-T(X)
Tipo de interfaz de ethernet	Gigabit Ethernet
Algoritmos de seguridad soportados	3DES,802.1x RADIUS, AES, DES
Ruta estática	Si
Enrutamiento basado en políticas	Si
Puertos e Interfaces	
Ethernet LAN(RJ-45) cantidad de puertos	4
Puertos WAN	2
Versión USB	3.2 Gen 1 (3.1 Gen 1)
Características de administración	
Administración basada en web	Si
Calidad de servicio (QoS) soporte	Si
Multidifusión, soporte	Si
Protocolos de routing	BGP, EIGRP, HSRP, IS-IS, OSPF, RIP-1, RIP-2, VRRP
Seguridad	
Lista de Control de Acceso (ACL)	Si
Cortafuegos	Si
Antena	
Tipo de Antena	No compatible
Memoria	
Memoria Interna	4096 MB
Memoria Flash	4096 MB
Otras Características	
Redes Wifi	• Facilita la configuración y gestión de los servicios WLAN y LAN • Soporta Mobility Express para WLAN

1.4.15. CISCO ISR4221

Este equipo de la marca CISCO está diseñado para soportar altos niveles de convergencia y capacidades de red inteligente, además, permite una comunicación directa tanto con los centros de datos privados como con nubes públicas a través de diversos enlaces, incluidas las VPN de conmutación de equipos multiprotocolo (MPLS) e Internet. Con todas estas características podemos decir que el *router* CISCO ISR4221 es capaz de soportar el protocolo de administración RESTCONF como se aprecia en la Tabla 1. 3, donde se encuentran todas las características de este equipo.

Tabla 1. 3 Tabla de descripción del equipo CISCO ISR4221

Memoria Flash	
Memoria Instalada	8 GB
Máximo tamaño de memoria soportada	8 GB
Interfaces	
Interfaz	Ethernet 10 Base-T/100 Base-TX/ 1000 Base- T
Cantidad de puertos RJ45	2
Tipo de puerto USB	USB 2.0
Tipo de conector	SFP (mini-GBIC)
Red	
Tipo de dispositivo	Router
Protocolos de enlace	Ethernet, Fast Ethernet, Gigabit Ethernet
Protocolos de administración remota	RMON, SNMP, RESTCONF
Administración Web	Si
Características	Soporta Access Control List (ACL), Class-Based Weighted Fair Queuing (CBWFQ), IPFIX, IPv6, NetFlow, Quality of Service (QoS), RADIUS, Syslog, VLAN, VPN, Wall mountable, Weighted Random Early Detection (WRED)
Protocolos de transporte de Red	DHCP, IPSec, PPPoE
Protocolos de Enrutamiento	BGP, DVMRP, EIGRP, GRE, IGMPv3, IPv4-to-IPv6 Multicast, IS-IS, OSPF, PIM-SM, PIM-SSM, Policy-based routing (PBR), RIP-1, RIP-2, Static IPv4 routing, Static IPv6 routing
Puertos WAN	2
Memoria RAM	
Tamaño instalado	4 GB
Tecnología de la memoria	DDR3 SDRAM

1.4.16. Python

Es un lenguaje de programación utilizado para el desarrollo de aplicaciones web, ciencia de datos y machine learning. Entre sus características se destaca su facilidad de uso, que es eficiente y puede ser utilizado en múltiples plataformas, otra ventaja de usar este lenguaje es su capacidad de integrarse a todo tipo de sistema y que su uso es gratuito.

Al utilizar este lenguaje de programación podemos obtener los siguientes beneficios:

Una sintaxis básica similar a la utilizada en inglés.

Se utiliza un número reducido de líneas de código en comparación a otros lenguajes de programación.

Cuenta con un gran número de bibliotecas, lo que permite una reutilización para distintas tareas, además de su fácil implementación y uso.

Tiene una comunidad muy activa que trabaja por todo el mundo.

Existe diverso material en el internet, lo cual facilita el aprendizaje de este lenguaje de programación.

1.4.17. Flask

En un marco de aplicación web escrito en Python, el cual se basa en el kit de herramientas Werkzeg WSGI y el motor de plantillas Jinja2. Un marco de aplicación web representa una colección de bibliotecas y módulos que permiten a los desarrolladores de aplicaciones web escribir aplicaciones sin preocuparse por detalles de bajo nivel.

WSGI. - La interfaz de puerta de enlace del servidor web, es la especificación de una interfaz común entre servidores web y aplicaciones web.

Werkzeug. - Es un conjunto de herramientas WSGI que implementan solicitudes, objetos de respuesta y funciones de utilidad, esto permite la construcción de un marco web.

Jinja2. - Es un motor de plantillas para Python, combina una plantilla con una fuente de datos específica para representar una web dinámica.

Para poder utilizar esta biblioteca es necesario ejecutar el siguiente comando en el cmd (Interprete de comandos de Windows):

pip install Flask.

1.4.18. Request

Es una biblioteca estándar para realizar solicitudes HTTP en Python, abstrae las complejidades de realizar solicitudes detrás de una API para la interacción con los servicios y consumir datos de aplicación. Entre sus características detallan las siguientes:

Realizar solicitudes utilizando métodos HTTP más comunes.

Personalización de encabezados y datos de las solicitudes, utilizando la cadena de consulta y el cuerpo del mensaje.

Inspecciona los datos de solicitudes y respuestas.

Evita que la aplicación realice una copia de seguridad.

Para poder instalar la biblioteca, se debe ejecutar la siguiente línea de comando en el *shell* de Python:

pip install requests

Entre los métodos principales de HTTP, podemos describir los siguientes:

- Post.
- Put.
- Delete.
- Head.
- Patch.
- Options.

Los métodos HTTP determinan qué acción intenta realizar con el envío de peticiones, además permite revisar los códigos de estado donde el primer bit de información recopila el estado de la solicitud, por ejemplo, un 200 de OK, lo que significa una solicitud exitosa, por otro lado, un código 404 el cual representa la no existencia del recurso buscado, Python permite utilizar el comando *status_code* el cual se implemente como método de la biblioteca *request* al realizar consultas, lo cual facilita tomar acciones para estos mensajes. Algunos componentes útiles de esta biblioteca que facilitan la implementación de solicitudes con el protocolo RESTCONF, son el encabezado y contenido, estos componentes permiten personalizar o pasar un diccionario de encabezados HTTP los cuales indican qué tipos de contenido puede manejar la aplicación.

2. METODOLOGÍA

2.1. Topología de red

Para la implementación de la Figura 2. 1, se utilizan diferentes protocolos como NAT y DHCP, los cuales permiten utilizar internet, a su vez, permiten que sea posible la conexión de diferentes clientes (computadores, laptops y celulares), para conocer cómo se implementan estos protocolos podemos consultarlo en la sección de Anexos 5.1. Para describir la implementación de la topología se inicia con el acceso a internet, para lo cual se utiliza algún proveedor o se conectará a una fuente con este servicio, lo siguiente es utilizar un router el cual permita salir desde la red interna a Internet, además, este contará con una dirección IP la cual permite conectar otros dispositivos de red para este desarrollo se configuran dos switches capa 3, por otro lado tenemos que el dispositivo final será una NMS, para finalizar la descripción de la topología se considera que cada equipo conectado cuente con su pool DHCP, esto con la finalidad de poder identificar a cual dispositivo se conecta la NMS, esto también permite escalabilidad, porque, existe la posibilidad de seguir aumentando equipos los cuales solo deberán contar con la configuración RESTCONF descrita previamente para poder utilizar la NMS.

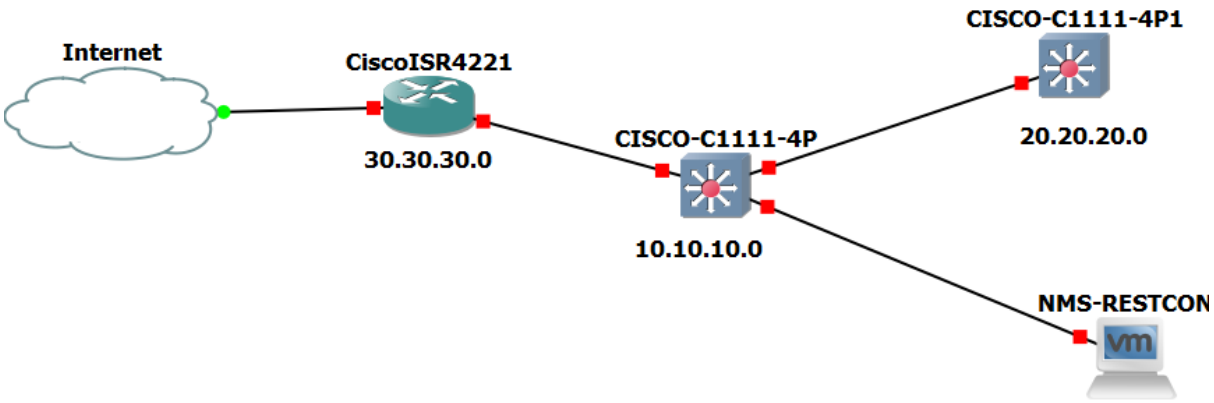

Figura 2. 1 Topología implementada para pruebas de funcionamiento de la NMS

2.2. Arquitectura de funcionamiento de la red

En la Figura 2. 2 se aprecia cómo se implementa la NMS y la interacción que debe realizar el usuario con la aplicación, esto con la finalidad de entender las funciones que puede realizar la NMS de las cuales son:

- Obtener información.
- Realizar configuraciones.
- Generar un reporte.

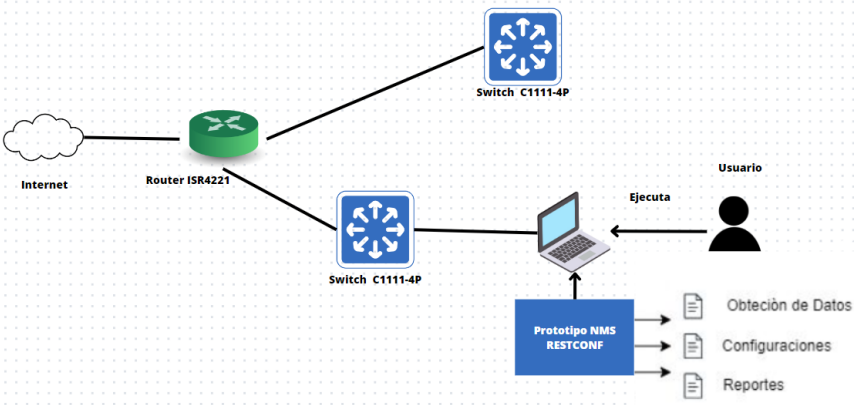

Figura 2. 2 Esquema a detalle de la arquitectura de funcionamiento de la NMS

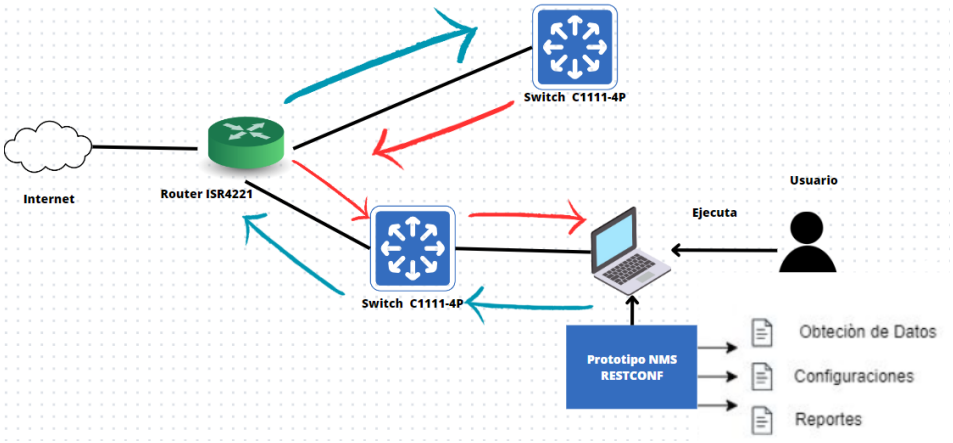

Figura 2. 3 Envió y recepción de solicitudes utilizando la arquitectura de funcionamiento

En la Figura 2. 3 se representa como el usuario puede realizar peticiones a cualquier equipo de red conectado a la topología previamente descrita, adicionalmente se aprecian flechas azules las cuales representan un mensaje de petición de configuración u obtención de información, por lo cual la NMS debe ser capaz de realizar cambios, además, se observan flechas rojas representando mensajes de envió de datos desde los dispositivos de red a la NMS, los mensajes enviados desde los equipos de red deben ser procesados para mostrar al usuario una información entendible y fácil de comprender entre algunas de las actividades previstas para el funcionamiento de la aplicación se consideran las siguientes:

Envío de solicitudes

- Cambiar el nombre del dispositivo de red.
- Crear VLANS.
- Asignar direcciones IP.

Recepción de solicitudes

- Mostrar las VLANS.
- Listar los usuarios.
- Visualizar reportes de protocolos.

2.3. Diagrama de procesos

Para poder entender la arquitectura de funcionamiento de la red lo representamos en un diagrama de procesos que ejemplifica las tareas que debe realizar el usuario, la NMS y los dispositivos de red. En la Figura 2. 4 se aprecia las tareas definidas para cada actor entre lo cual se destaca que para empezar la interacción entre los actores descritos debe existir la necesidad de gestionar un equipo de red por un usuario, una solicitud de configuración, información o reportes para que gestione la NMS y un dispositivo de red que genere una respuesta a la solicitud que le envíen.

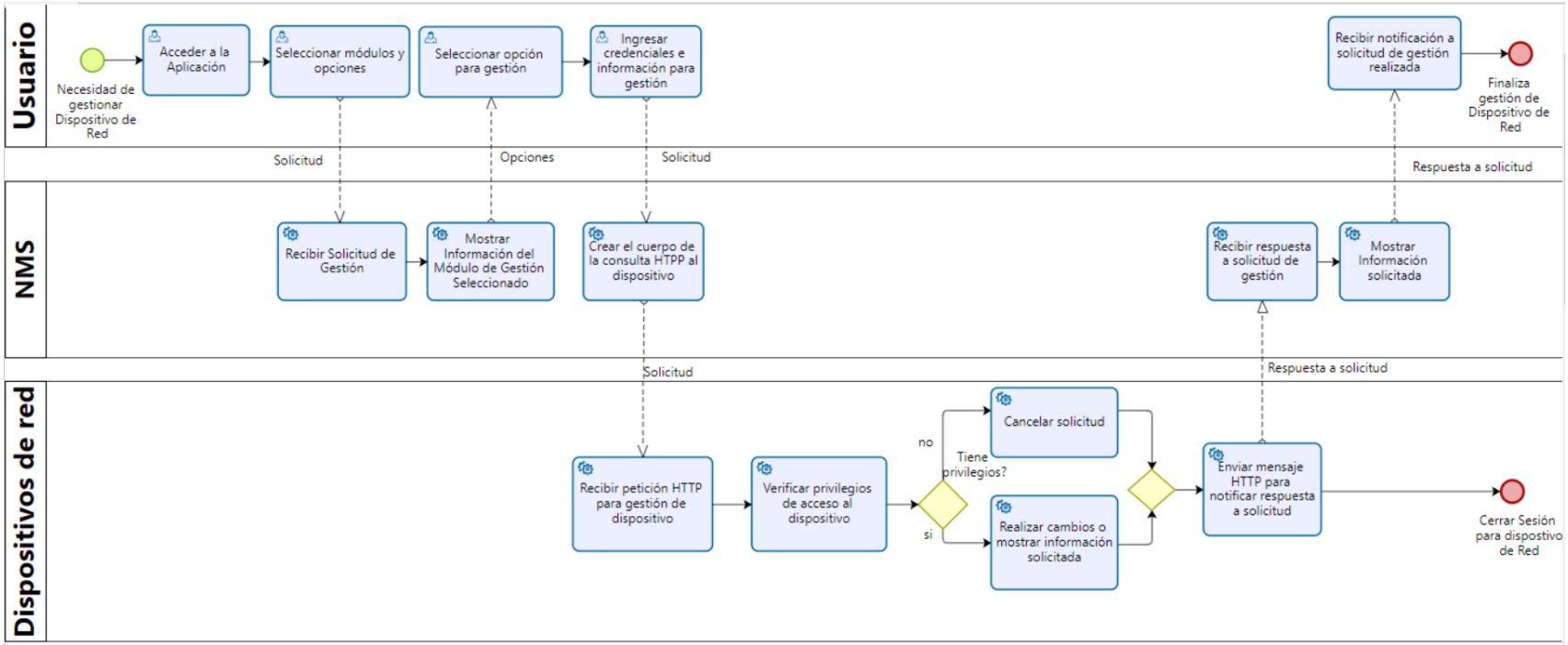

Figura 2. 4 Diagrama de procesos con todos los actores de la arquitectura de funcionamiento para la implementación de la NMS

2.4. Metodología para el desarrollo de software

Para el desarrollo de la NMS se optó por una aplicación WEB, donde el usuario pueda ingresar la dirección IP del equipo para realizar el monitoreo y los parámetros para las configuraciones deseadas, evitando mantener una sesión abierta todo el tiempo lo cual provoca un problema al intentar controlar diferentes equipos a la vez, con esta finalidad se utiliza la metodología SCRUM.

2.4.1. Descripción del proyecto

Para el desarrollo de la aplicación tenemos 3 tópicos importantes los cuales serán la base para las tareas que este debe realizar, las cuales se detallan de la siguiente manera:

- Módulo de configuraciones.
- Módulo de información.
- Módulo de reportes.

Con la finalidad de facilitar el entendimiento de cómo se desarrollará la metodología para la construcción de la NMS, con este fin, se dividirá la información de todo el proyecto en los módulos descritos previamente, para su posterior desarrollo, con esto, se debe contar con un diagrama de casos de uso y las tablas que describen las interacciones entre el usuario y el sistema.

2.4.1.1. Módulo de Configuraciones

El módulo de configuración nos permitirá realizar actividades que permitan modificar las interfaces físicas de los equipos de red o a su vez implementar protocolos que permitan añadir o identificar puntos de acceso a la red, para esto, se debe implementar las opciones necesarias que permitan realizar esta actividad como lo que podemos apreciar en la Figura 2. 5, en esta imagen se observa al usuario con todas las actividades que puede realizar en la NMS, además esto será el punto de inicio para comprobar el correcto funcionamiento de la aplicación, por motivo de que si no se configura encontraremos valores por defecto o a su vez información irrelevante para análisis, con lo cual, no se pueden generar propuestas de soluciones. Cabe recalcar que este diagrama de caso de uso describe las funciones que se implementarán en la ventana del módulo de configuraciones, dejando de esta manera un puente para implementar nuevas funciones en caso de ser requerido.

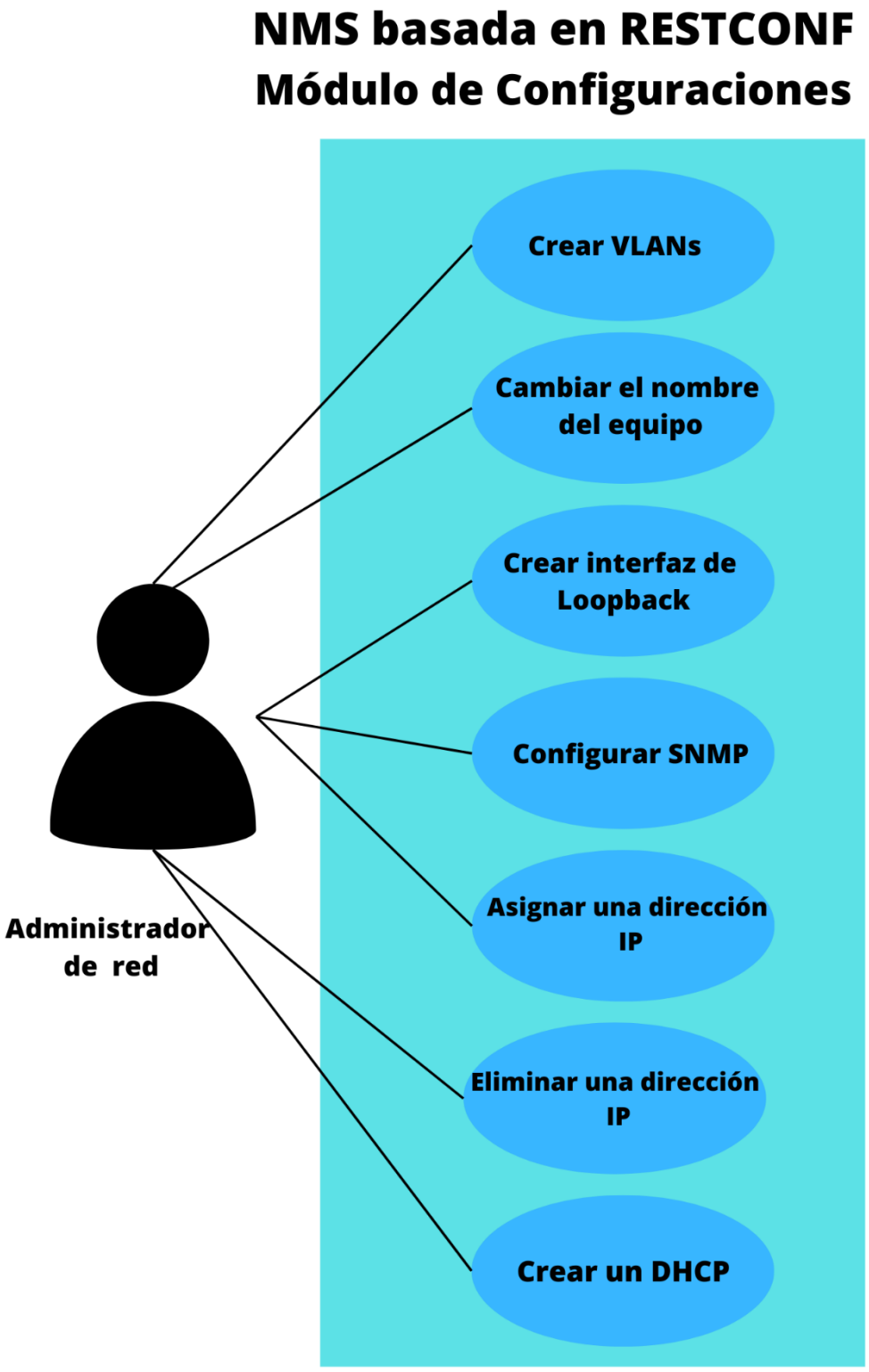

Figura 2. 5 Diagrama de caso de uso del módulo de configuraciones

Para continuar con la descripción de los requisitos para el funcionamiento de la NMS se tienen en la **¡Error! No se encuentra el origen de la referencia.** una d escripción de la opción crear VLANs en la cual se aprecia las características necesarias para la implementación de esta opción, además, de cómo se genera

la interacción del usuario con el sistema, para más ejemplo se puede consultar en anexos en el apartado 5.2 donde se encuentran las opciones que tiene el usuario para realizar configuraciones en la NMS.

Tabla 2. 1 Tabla de descripción de uso para la creación de VLANs

Caso de Uso	Crear VLANs
Actores	Administrador de red
Resumen	El Administrador de red creará una nueva VLAN, a la cual puede asignar una dirección IPv4 con su debida máscara, podrá asignarle un nombre como Vlan (número). El Dispositivo Cisco lo guardará en memoria y enviará un mensaje 204 en caso de crearla, con lo cual se asegura una respuesta de OK para el Administrador de red o en caso de error un mensaje 404 para advertir al Administrador de red que no se pudo crear.
Precondiciones	El Administrador de red debe haber seleccionado en el menú principal la opción configuraciones.
Postcondiciones	No aplica.
Flujo de Eventos	
Actor	Sistema
El Administrador de red selecciona la opción de Configuraciones. El Administrador de red selecciona la opción de Configurar VLANs. El Administrador de red ingresa la información solicitada en la ventana. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco genera una respuesta.	El sistema muestra una ventana con los datos requeridos (nombre, dirección IP, máscara de red, Administrador de red y contraseña) para crear una VLAN El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema construye el cuerpo de información y la URL. El sistema envía la información. El sistema recibe la respuesta y envía un mensaje de OK o error.

2.4.1.2. Módulo de información

El módulo de configuración nos permite visualizar la información que tiene nuestro dispositivo de red, para lo cual, es importante haber completado el primer paso de configuraciones esto para poder visualizar desde nuestra NMS los cambios que hemos realizado, la finalidad de este módulo es mostrar de una forma amigable y entendible los cambios hechos por el usuario, para esto se han definido opciones las cuales se encuentran representadas en la Figura 2. 6 donde se aprecia un diagrama de caso de uso para este módulo, todo esto con la finalidad de entender cuáles son las acciones que puede realizar el usuario al interactuar con la interfaz a desarrollar para este módulo. Para mejorar el entendimiento de cada una de las opciones se planea el desarrollo de tablas de

caso de uso que se describirán posteriormente. Por último, las opciones presentadas pueden aumentar en función de las necesidades presentadas por un usuario en específico, dando de esta manera escalabilidad a la NMS.

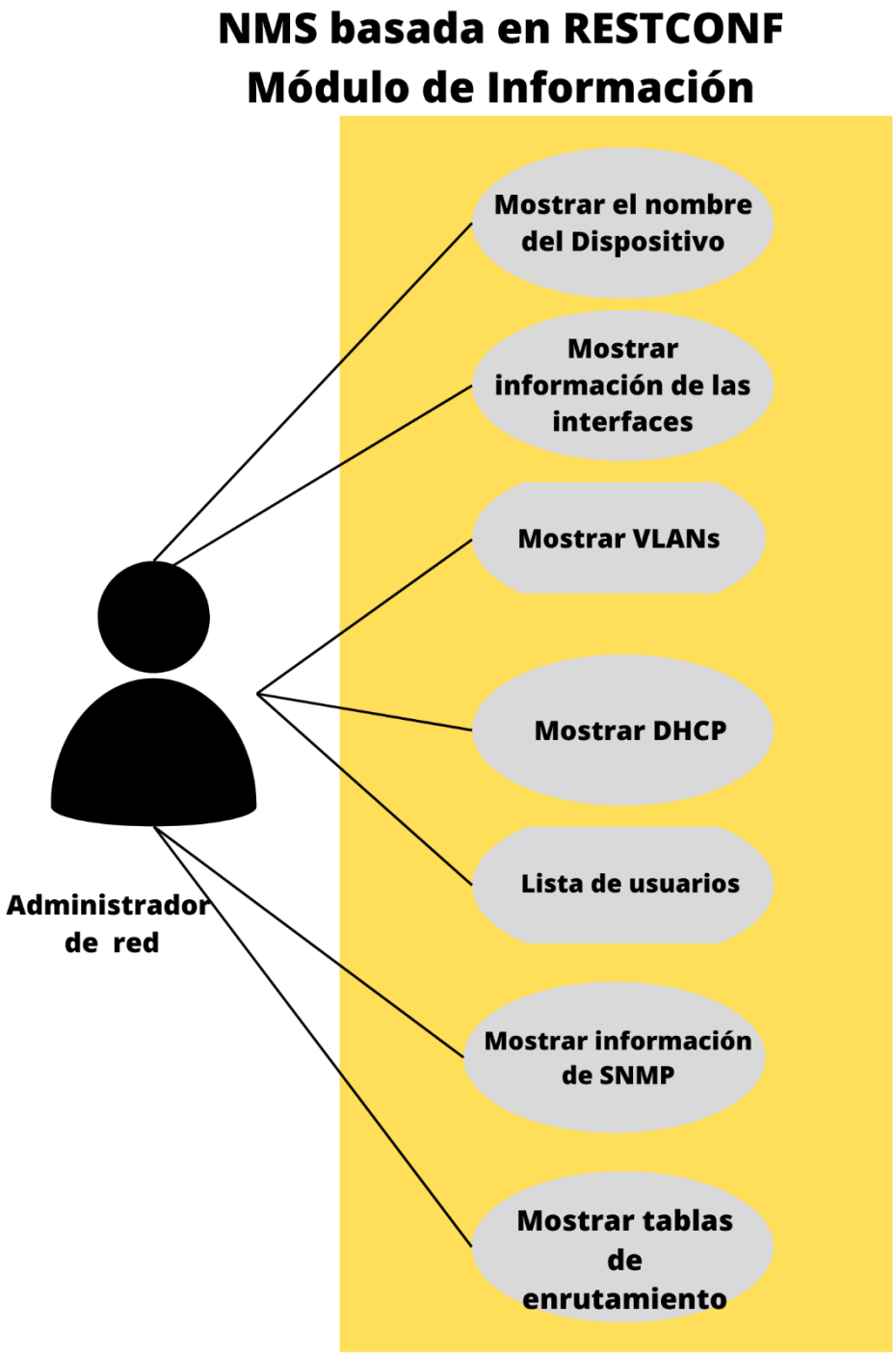

Figura 2. 6 Diagrama de caso de uso para el módulo de Información

Para conocer como es la interacción del usuario con el dispositivo NMS, se utiliza las tablas de casos de uso para cada una de las opciones propuestas para el desarrollo de la aplicación, en la Tabla 2. 2 se aprecia la interacción descrita y los pasos de cómo realiza la tarea el sistema. Para ver la descripción de las otras

tareas se debe consultar en la parte de anexos 5.2, en el apartado del módulo de información para conocer cómo deben funcionar las otras tareas requeridas para el funcionamiento de la NMS.

Tabla 2. 2 Tabla de caso de uso para mostrar el nombre del equipo

Caso de Uso	Mostrar el nombre del equipo
Actores	Administrador de red
Resumen	El administrador de red tendrá la capacidad de visualizar el nombre del equipo, con lo cual, se obtendrá el cambio realizado por el usuario si así lo hizo con las opciones del módulo de configuraciones. El dispositivo Cisco responderá a la consulta realizada por la NMS, la cual debe ser clara y entendible para el administrador de red, los procesos de transformación del formato de datos recibidos se realizarán de forma transparente para el administrador de red .
Precondiciones	El administrador de red debe haber seleccionado en el menú principal la opción información.
Postcondiciones	La ventana de información debe mostrar el nombre del equipo con el cual se está trabajando
Flujo de Eventos	
Actor	Sistema
El administrador de red selecciona la opción de información. El administrador de red ingresa el usuario, la contraseña y la dirección IP del equipo del que se obtendrá información. El usuario selecciona la opción de Mostrar nombre del dispositivo. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco envía la respuesta de lo solicitado.	El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema selecciona el módulo para realizar la consulta y construir la URL. El sistema envía la información. El sistema recibe la respuesta a la solicitud realizada. El sistema transforma los datos a formato JSON. El sistema guarda la información en una variable, El sistema muestra al usuario la información que ha procesado y almacenado.

2.4.1.3. Módulo de Reportes

Una vez descritos los módulos de configuración e información, el siguiente paso será definir el módulo de reportes en el cual se tiene como objetivo mostrar información de modo detallado, para lo cual, se piensa agruparlos por las características o configuraciones realizadas previamente. Para este módulo se

pensó en tres posibles soluciones para generar una diferencia clara y notoria entre los módulos de información y reportes, los reportes deberán ser generados mediante una plantilla que permita ver el nombre del equipo, seguido por las características como podrían ser protocoles o estado del equipo, para que el usuario pueda ver cómo está desempeñándose el dispositivo CISCO, o cuales son los protocolos configurados en este, además se piensa en la posibilidad de estimar el enrutamiento con los vecinos cercanos y la tablas que se generan para enviar paquetes construyendo tablas ARP, con este fin, tenemos una descripción de las acciones que el usuario puede hacer en la NMS como se muestra en la Figura 2. 7, en la cual se aprecia las opciones que debe tener este módulo todo esto mediante un diagrama de casos de uso para facilitar la visualización de las tareas que puede realizar el usuario.

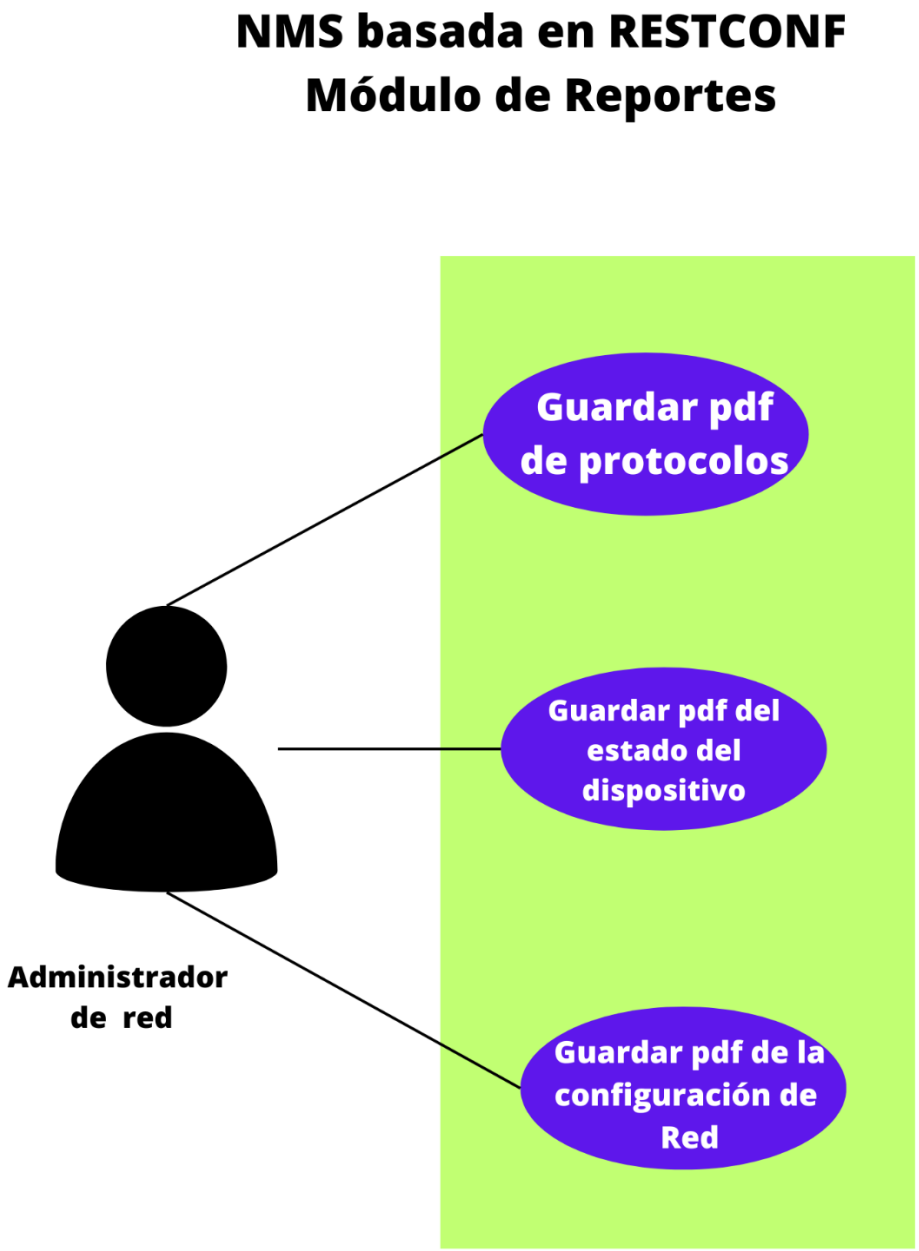

Figura 2. 7 Diagrama de caso de uso para el módulo de reportes

Para finalizar, la descripción de los requisitos a implementarse en la NMS se utilizan tablas da caso de uso con la descripción de cómo se debe generar un reporte y como están agrupados, las características de los dispositivos de red. Un punto adicional es que se tiene una descripción de cuál es la interacción del usuario con el dispositivo CISCO y a su vez las tareas que debe realizar el sistema para que la información está visible de una forma clara, todo esto con la finalidad de que sean transparentes los procesos realizados por el sistema, para que el usuario pueda leer y entender lo mostrado por la NMS

Tabla 2. 3 Tabla de caso de uso para ver el reporte de protocolos

Caso de Uso	Protocolos
Actores	Administrador de red
Resumen	El administrador de red podrá visualizar un reporte con los protocolos configurados en el dispositivo de red, el sistema debe mostrar en la pantalla un resumen del protocolo y como se encuentra configurado, esto incluirá direcciones IP y las interfaces a las cuales están vinculadas, entre otras, características. Para realizar esta tarea es importante las solicitudes enviadas a los dispositivos CISCO, el número de estas dependerá de cómo se estructure el reporte final, por lo cual se deberá generar una respuesta para todo esto.
Precondiciones	El administrador de red debe haber seleccionado en el menú principal la opción reportes.
Postcondiciones	Se debe poder visualizar un archivo pdf con toda la información generada por el reporte.
Flujo de Eventos	
Actor	Sistema
El usuario selecciona la opción de reportes. El usuario ingresa el usuario, la contraseña y la dirección IP del equipo del que se obtendrá información. El usuario selecciona la opción de *Guardar pdf de protocolos*. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco envía la respuesta de lo solicitado.	El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema selección el módulo para realizar las consultas y construye los URL. El sistema envía la información. El sistema recibe la respuesta a lo solicitud realizada. El sistema transforma los datos a formato JSON. El sistema guarda la información. El sistema almacena en un archivo PDF la información generada por el reporte.

Una vez descrito los requerimientos debemos considerar que los equipos CISCO soporten el protocolo y las tareas que se desea realizar en cada uno de los módulos, por lo cual, es importante considerar la versión de las bibliotecas YANG para la compatibilidad entre estos, además de la posibilidad de utilizar las mismas consultas realizadas desde la NMS.

Una vez definido los requerimientos para la construcción de la aplicación (NMS), lo siguiente será definir la metodología del desarrollo de la aplicación para lo cual se utiliza el Sprint del marco de trabajo de Scrum.

2.4.2. Sprint 1

2.4.2.1. Diseño de las ventanas para el servidor web.

Para el desarrollo del sprint, se esquematiza las ventanas en función a los módulos necesarios para la implementación y el desarrollo de la NMS, con este fin se tienen la Figura 2. 8, Figura 2. 9, Figura 2. 10 y la Figura 2. 11, las cuales representan las ventanas que se deben implementar para el desarrollo del servidor web.

Figura 2. 8 Resultado de la ventana Index

La Figura 2. 8 representa a la página de inicio que se desea implementar, donde se tiene una breve descripción del protocolo, además, de las características de los dispositivos CISCO, con los cuales, puede funcionar la NMS a desarrollar. Adicionalmente, se tiene el menú donde constan los módulos de *Configuraciones, Información y Reportes*, para garantizar el cumplimiento de lo planteado en los requerimientos descritos previamente. En la Figura 2. 9, que representa el módulo de *Configuraciones,* debe constar las opciones de las configuraciones que el administrador de red puede realizar, además de heredar el menú para permitir navegar a otro módulo, este mismo desarrollo se lo debe realizar en la Figura 2. 10 y Figura 2. 11, con sus respectivas opciones de configuración.

Figura 2. 9 Resultado del archivo Configuraciones

Figura 2. 10 Resultado del archivo Información

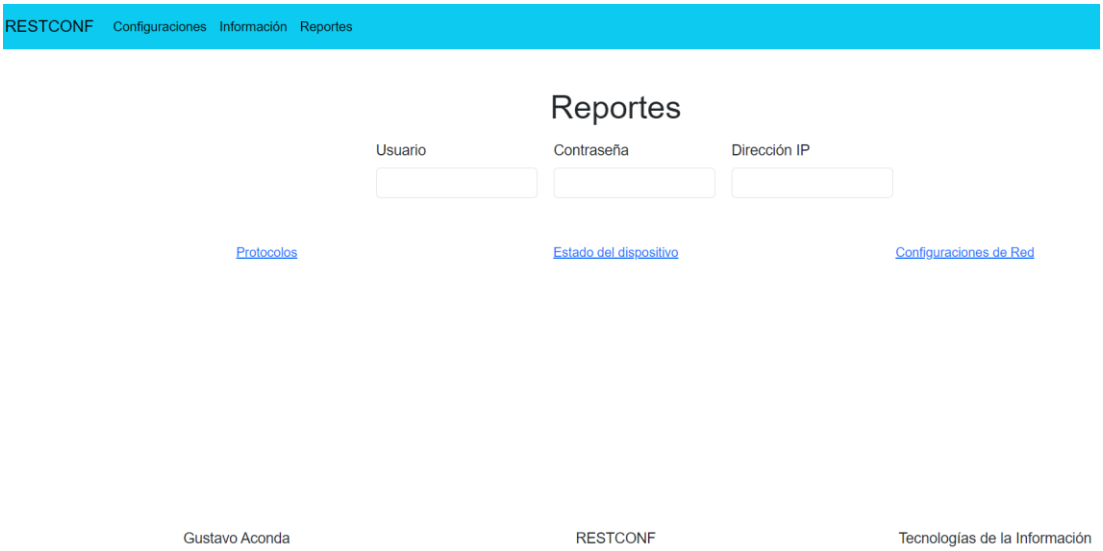

Figura 2. 11 Resultado del archivo Reportes

2.4.2.2. Diseño de ventanas emergentes.

Un complemento para el desarrollo del servidor web, son las ventanas emergentes las cuales facilitan la tarea de ingresar datos para realizar configuraciones, esto debido a que tienen diferentes parámetros por lo cual la Figura 2. 12 y Figura 2. 13, son un ejemplo del desarrollo pensado para cumplir las funcionalidades descritas en los requerimientos.

Con el fin de dar un mejor entendimiento de cuál es el formato para que el usuario ingrese información referente al cambio solicitado, se desea escribir en los cuadros de texto una guía, es decir la forma como se escribe una dirección *IP* o a su vez una *Máscara IP,* en el caso de no ser necesaria la ayuda el cuadro de texto se encuentra en blanco, esto con la finalidad de pasar mensajes con el tipo de variable *string* para su posterior procesamiento. Por último, se tiene como característica que el número de campos será ajustado al tipo de configuración y los parámetros que este necesite.

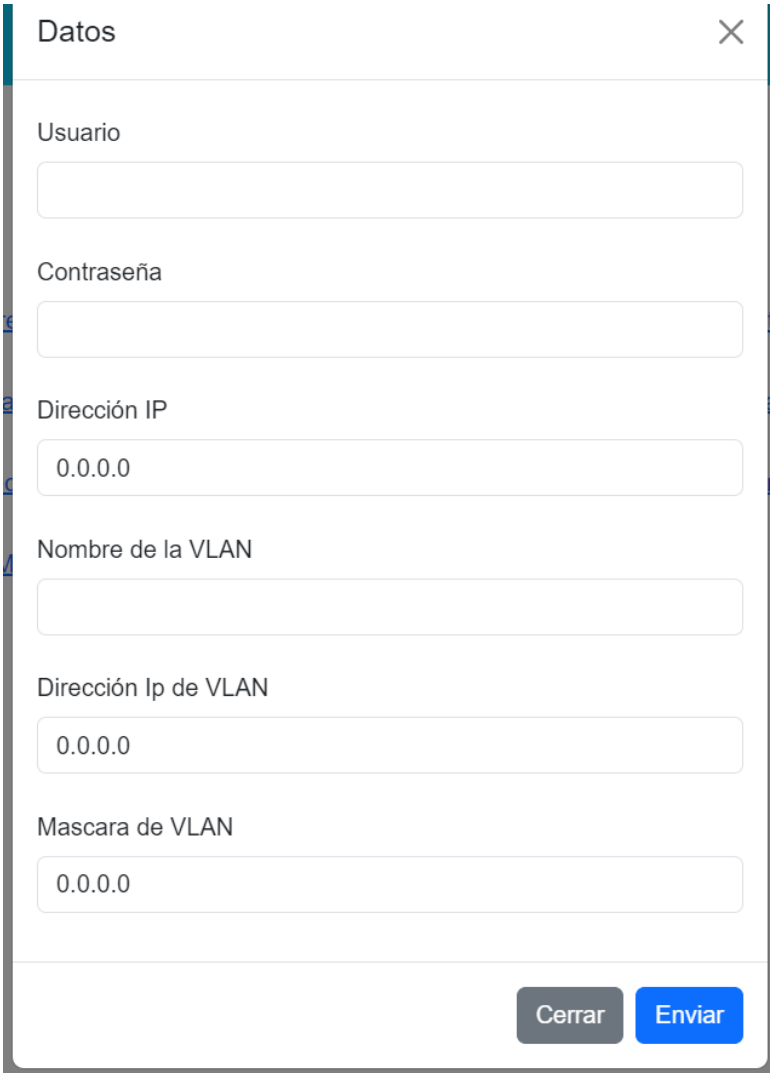

Figura 2. 12 Ventana de ingreso de datos para un VLAN

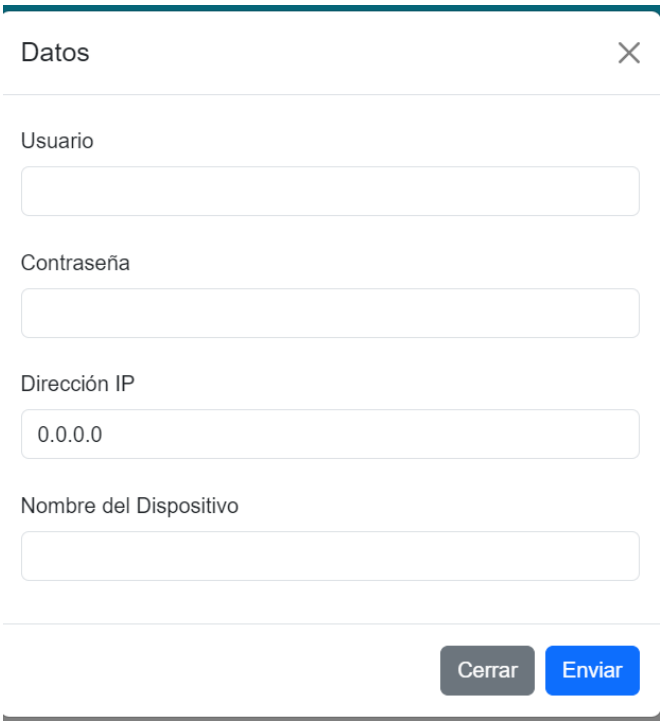

Figura 2. 13 Ventana de ingreso de datos para cambiar el nombre del equipo CISCO

2.4.3. Sprint 2

2.4.3.1. Códigos

El objetivo principal de esta sección es representar la estructura base de los códigos implementados para el desarrollo y funcionamiento de la página web la cual representa la NMS propuesta en el plan. Con la finalidad de tener un código que cumplas los principios de la programación orientada a objetos que son encapsulamiento, herencia, polimorfismo y abstracción, para lo cual se crea los siguientes archivos:

- informacion.html.
- index.html.
- configuraciones.html.
- reportes.html.
- base.html.
- servicio.py
- index.py

El Código 2. 1, representa la manera de cómo se debe alojar los archivos de extensión *.html,* a los cuales se hace referencia al navegar en la pantalla principal, como se aprecia en la imagen se utiliza la biblioteca *Flask,* la cual facilita el enrutamiento hacia los archivos necesarios para el desarrollo de la NMS, además, todo este código se encuentra en el archivo *index.py*

```python
@aplicacion.route('/')
def principalr():
    return render_template('index.html')
#Rutas de la aplicación web, pagina de configuraciones
@aplicacion.route('/configuracion')
def configuracion():
    return render_template('configuraciones.html')
#Rutas de la aplicación web, pagina de informacion
@aplicacion.route('/informacion')
def informacion():
    return render_template('informacion.html')
#Rutas de la aplicación web, pagina de informacion
@aplicacion.route('/reportes',methods=['GET','POST'])
def reportes():
    return render_template('reportes.html')
```

Código 2. 1 Código base para la locación de los archivos de los módulos de la NMS

En el Código 2. 2, se aprecia la creación del menú de navegación con lo cual se conectan los diferentes módulos de la NMS, su finalidad es permitir una interacción dinámica al usuario y el servidor web.

```
body>
    <nav class="navbar navbar-expand-lg bg-info">
        <div class="container-fluid">
          <a class="navbar-brand" href="/"  aria-setsize="" >RESTCONF</a>
          <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav"
aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarNav">
            <ul class="navbar-nav">
              <li class="nav-item">
                <a class="nav-link active" aria-current="page" href="{{
url_for('configuracion')}}">Configuraciones</a>
              </li>
              <li class="nav-item">
                <a class="nav-link active" aria-current="page" href="{{
url_for('informacion')}}">Información</a>
              </li>
              <li class="nav-item">
                <a class="nav-link active" aria-current="page" href="{{
url_for('reportes')}}">Reportes</a>
              </li>
```

Código 2. 2 Menú de navegación.

El Código 2. 3, representa la creación de un bloque donde se alojarán los códigos de las plantillas complementarias, es decir se heredan las características del archivo *base.html* en todos los archivos que tienen la extensión *.html*. Además, contiene un pie de página con información referente al autor, la carrera y el protocolo con el cual se está trabajando.

```
<div class="container text-align-midle">
        {% block body%}

        {% endblock %}
      </div>
<footer>
    <div class="col-md-11 mx-auto">
        <table   class="table table-borderless">
            <tbody>
                <tr>
                    <td >Gustavo Aconda</td>
                    <td >RESTCONF</td>
                    <td> Tecnologías de la Información</td>
```

Código 2. 3 Código para crear un bloque y el pie de página

En el Código 2. 4, se implementa las líneas de código para acceder a las características del archivo base.html, desde cualquier plantilla creada para el desarrollo de la NMS.

```
{%extends './base.html'%}
{%block title%}
```

Código 2. 4 Código para heredar características de un archivo de extensión .html

En el Código 2. 5 y Código 2. 6 , se implementa una nueva ventana emergente para la imagen mostrada se puede observar que tiene el identificativo para cambiar el nombre, además utilizad el método Post para enviar los datos del formulario.

```
<div class="modal fade" id="cambioNombre" tabindex="-1" aria-
labelledby="exampleModalLabel" aria-hidden="true">
    <div class="modal-dialog">
        <h1 class="modal-title fs-5" id="exampleModalLabel">Datos</h1>
 <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
        </div>
        <form action="/agregarnombre" method='POST' >
          <div class="modal-body">
            <div class="mb-3">
    <label for="recipient-name" class="col-form-label">Usuario</label>
         <input type="text" class="form-control" name="usuario" id="">
            </div>
```

Código 2. 5 Código para las ventanas emergentes parte 1

```
<input type="password" class="form-control" name="contrasenia">
            </div>
            <div class="mb-3">
              <label for="message-text" class="col-form-
label">Dirección IP</label>
              <input type="text" class="form-control"
name="direccionip" value="0.0.0.0">
            </div>
            <div class="mb-3">
              <label for="message-text" class="col-form-label">Nombre
del Dispositivo</label>
<input type="text" class="form-control" name="nombreDispositivo">

            <div class="modal-footer">
             <button type="button" class="btn btn-secondary" data-bs-
dismiss="modal">Cerrar</button>
            <button type="submit" class="btn btn-
primary">Enviar</button>
            </div>
          </form>
```

Código 2. 6 Código para las ventanas emergentes parte 2

Con estos códigos se termina el desarrollo de las ventanas del servidor web, por lo cual este se encuentra operativo, las pruebas de funcionamiento se las detallan en el apartado de *Resultados.* Para continuar con el desarrollo de la NMS se debe crear las funcionalidades necesarias para cumplir los requerimientos planteados previamente, se inicia con la creación del archivo de servicios el cual se encuentra diseñado para funcionar utilizando la programación orientada a objetos lo que facilita el llamado a métodos o el uso de constructores que contengan toda la información repetitiva al momento de realizar una consulta, el Código 2. 7 es el constructor desarrollado para realizar las peticiones su finalidad es definir el formato y las variables como el usuario, contraseña, dirección IP del equipo CISCO, el puerto y el encabezado, los cuales construyen el encabezado de lo solicitud que puede realizar el administrador de red.

```
class Servicio:
    Usuario = " "
    Contrasenia = " "
    Ip = " "
    def __init__(self,usuario,contrasenia,ip):
        self.Usuario = usuario
        self.Contrasenia = contrasenia
        self.Ip = ip
        self.port = "443"
        self.headers = {
                "Accept" : "application/yang-data+json",
                "Content-Type" : "application/yang-data+json",
```

Código 2. 7 Código para definir el constructor y los parámetros necesarios para una consulta

En el Código 2. 8, se crea un ejemplo de uno de los métodos disponibles en solicitudes de tipo REST como es el caso de PUT, con este código se puede realizar solicitudes implementado la url y el encabezado.

```
def Peticiones_put(self,url,payload):
        requests.packages.urllib3.disable_warnings()
        response = requests.put(url, headers=self.headers,
data=JSON.dumps(payload), auth=(self.Usuario, self.Contrasenia),
verify=False)
      if (response.status_code == 204 or response.status_code == 201):
            print("Successfully updated interface")
        else:
            print("Issue with updating interface")
```

Código 2. 8 Código para crear el método put

El Código 2. 9, es un ejemplo de cómo se construyen los métodos para realizar solicitudes a los dispositivos CISCO, las principales características para construir este método es definir el módulo en el cual se desea realizar los cambios y por ultimo los datos (*payload*), con estos parámetros se puede construir el url y utilizar el método de solicitud REST que necesite el cambio solicitado, pudiendo obtener información o mensajes de error u OK.

```
def Nombre_dispositivo (self, nombre):
  module = "Cisco-IOS-XE-native:native/hostname"
  url = f"https://{self.Ip}:{self.port}/restconf/data/{module}"
        payload = {
            "hostname":
                   nombre
        }
        self.Peticiones_put(url,payload)
```

Código 2. 9 Código para el método para cambiar el nombre del dispositivo CISCO

El Código 2. 10, es la forma como los datos ingresados por el usuario son receptados por el servidor web, en el ejemplo se aprecia que existe una comunicación de tipo POST, para enviar al archivo de servicios esa información y que se realice los cambios en el dispositivo CISCO.

```
@aplicacion.route('/agregarnombre',methods=['POST'])
def agregarnombre():
    if request.method == 'POST':
        usuario = request.form['usuario']
        contrasenia = request.form['contrasenia']
        direccionip = request.form['direccionip']
        nombreDispositivo = request.form['nombreDispositivo']
        nombre = ser.Servicio(usuario,contrasenia,direccionip)
        print(nombre.Nombre_dispositivo(nombreDispositivo))
        return 'recibido'
```

Código 2. 10 Código para conectar la interfaz del servidor web con el equipo CISCO

El Código 2. 11 es la forma como se realiza consultas desde las diferentes opciones con las cuales cuenta los módulos de información y reportes. Utilizando la biblioteca JQerry con lo cual se puede identificar la opción seleccionada ingresar los datos necesarios para traer la información y generar una respuesta que se visualiza en el servidor web.

```
<script type="text/javascript">
    var opcion;
    $('#nombre').click(function(){

      opcion=1;
    });
    $('#interfaces').click(function(){
      opcion=2;
    });
    $('#vlan').click(function(){
      opcion=3;
    });
    $('#dhcp').click(function(){
      opcion=4;
    });
    $('#direccionesip').click(function(){
      opcion=5;
    });
    $('#enrutamiento').click(function(){
      opcion=6;
    });

    $('#formulario').on('submit',function(event){
        event.preventDefault();
        console.log('ENTRE');
        console.log(opcion)
        $.ajax({
          type: "POST",
          url: '/consultar',
          data:{
            usuario:$('#usuario').val(),
            contrasenia:$('#contrasenia').val(),
            direccionip:$('#direccionip').val(),
            Opcion:opcion
          },
          success:function(response){
            console.log('Exitoso',opcion);
            console.log(response);
            imprimir(response);
          },
          error:function(response){
```

```
            alert('Error')
          }
        });

  })
  function imprimir(data){
    $('#encabezado').text(data['Respuesta'][0]);
    $('#texto').text(data['Respuesta'][1]);
  }

</script>
```

Código 2. 11 Código para los métodos de recepción de información

Para poder visualizar el código utilizado para la implementación de la NMS en los Anexos 5.3, donde se especifica el enlace al que se debe acceder además de como se puede poner en funcionamiento el servidor web.

3. RESULTADOS, CONCLUSIONES Y RECOMENDACIONES

3.2. Resultados

Para obtener los resultados deseados en la NMS se contó con la ayuda de POSTMAN, el cual es un software que permite realizar solicitudes de tipo REST como se aprecia en la Figura 3. 1, donde se visualiza las bibliotecas necesarias para trabajar con el protocolo RESTCONF.

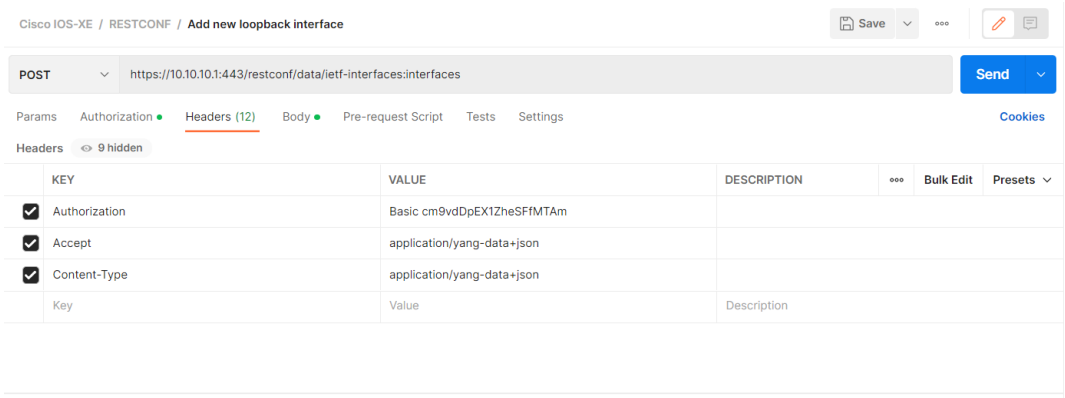

Figura 3. 1 Parámetros del encabezado utilizado para las peticiones con RESTCONF

En la Figura 3. 2, se tiene un ejemplo de una solicitud tipo GET, en la cual la URL cuenta con dos parámetros muy importantes, los cuales son el módulo YANG y los datos necesarios para el cambio, esto por la singularidad de que el módulo puede cambiar para obtener diferentes datos o a su vez podemos enviar parámetros mediante data el cual se encarga de enviar los requerimientos del usuario cuando se utiliza el método POST, PATCH o PUT.

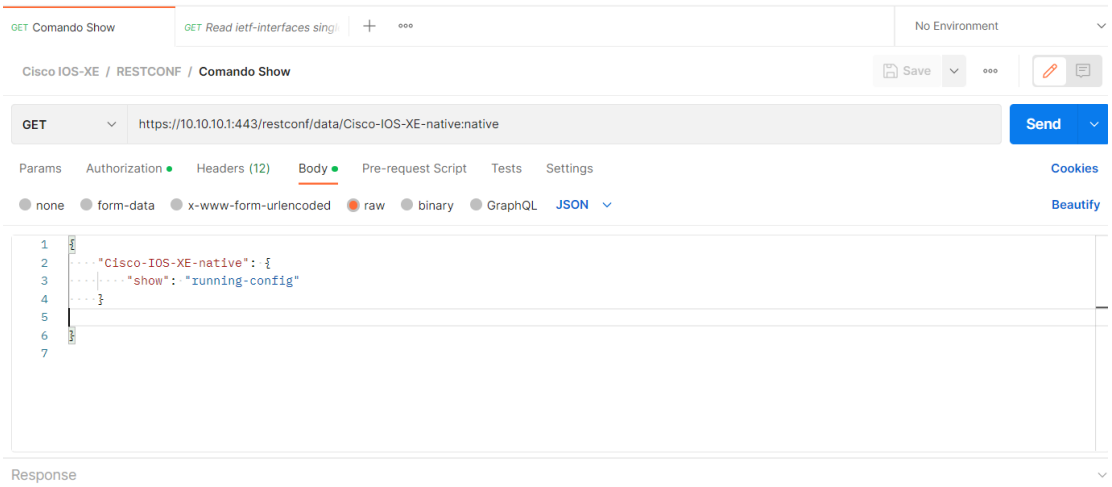

Figura 3. 2 Parámetros para enviar el comando show con sus diferentes complementos

Para la obtención de resultados en la NMS desarrollada, las pruebas se realizan enfatizando los módulos a los que pertenece, esto con el fin de dar cumplimiento a lo planteado en el desarrollo del aplicativo.

3.2.1. Módulo de Configuraciones.

Los resultados obtenidos en este módulo son principalmente las diferentes ventanas en las que podemos realizar configuraciones como es el caso de la Figura 3. 3 y Figura 3. 4, en las cuales se visualiza cuáles son los parámetros necesarios para realizar un cambio de nombre o a su vez crear una VLAN, como se ve en el ejemplo las configuraciones se adaptan a las necesidades del usuario.

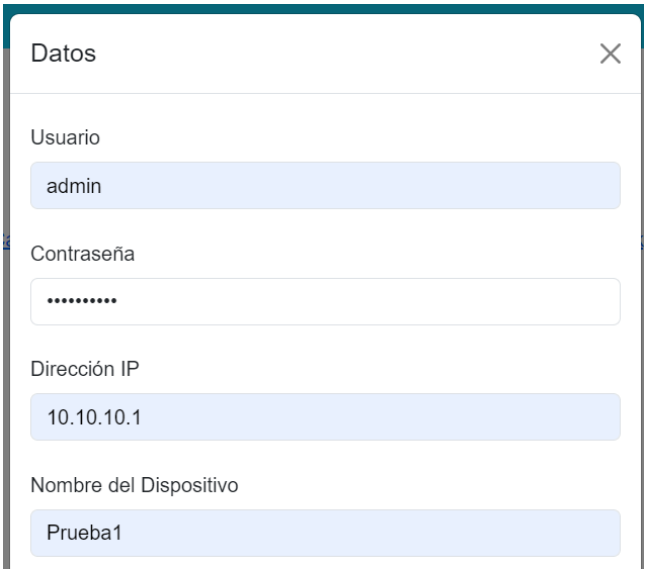

Figura 3. 3 Cambio del hostname

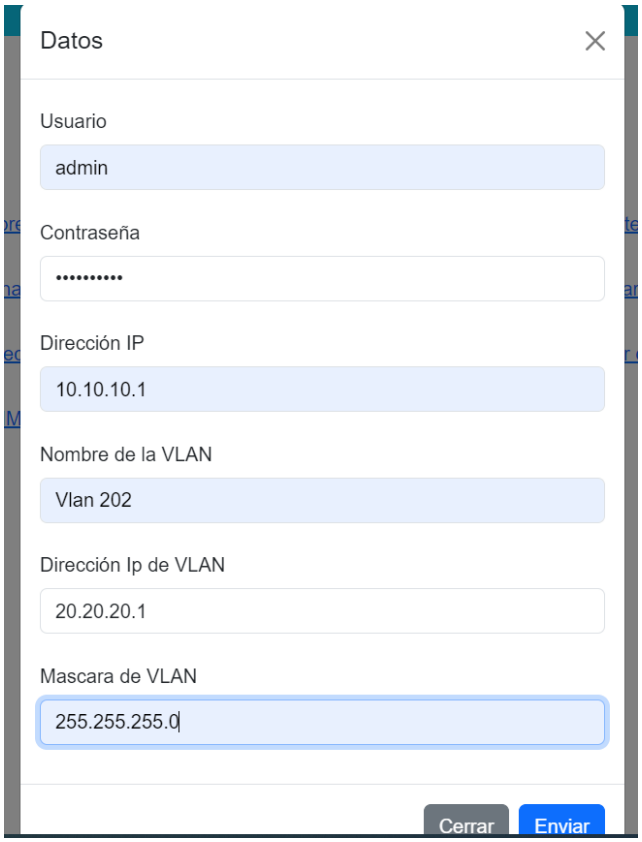

Figura 3. 4 Agregar una VLAN

En la Figura 3. 5 y Figura 3. 6, es un ejemplo de lo que el usuario observa al seleccionar una opción, como se ve la plantilla base cuenta con diferentes ayudas para facilitar el ingreso de datos, a su vez, también se visualiza como esa misma plantilla queda una vez editados cada uno de los campos.

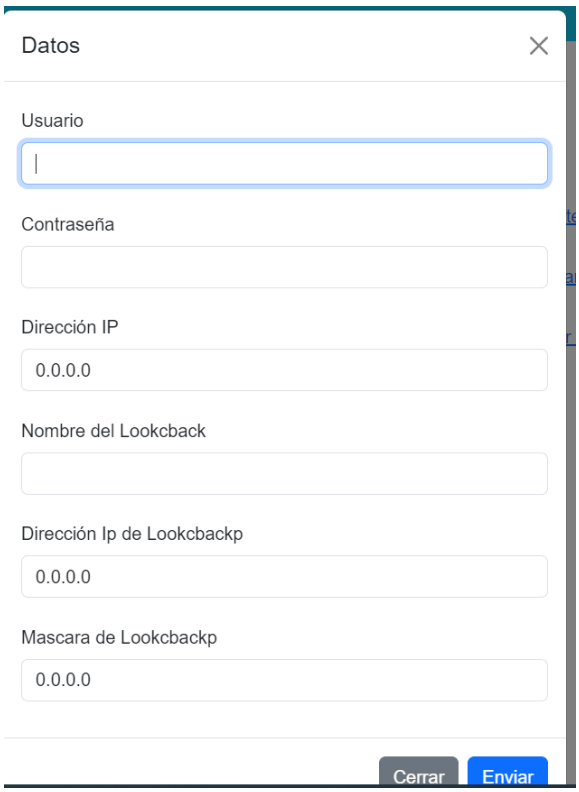

Figura 3. 5 Modelo para agregar una interfaz de Loopback

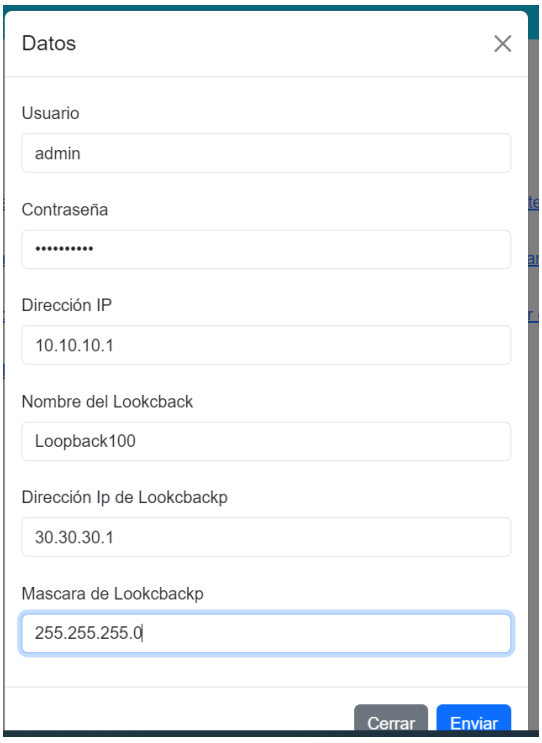

Figura 3. 6 Ingresar una interfaz de loopback

Por último, se tiene el resultado de los campos más extensos, como se aprecia en la Figura 3. 7 y Figura 3. 8, donde el número de parámetros ingresados por el usuario es alto, por lo cual, se dice que la ventana emergente que mira el usuario puede adaptarse a cualquier necesidad esto implica que crecerá a medida de los requerimientos de la configuración desea.

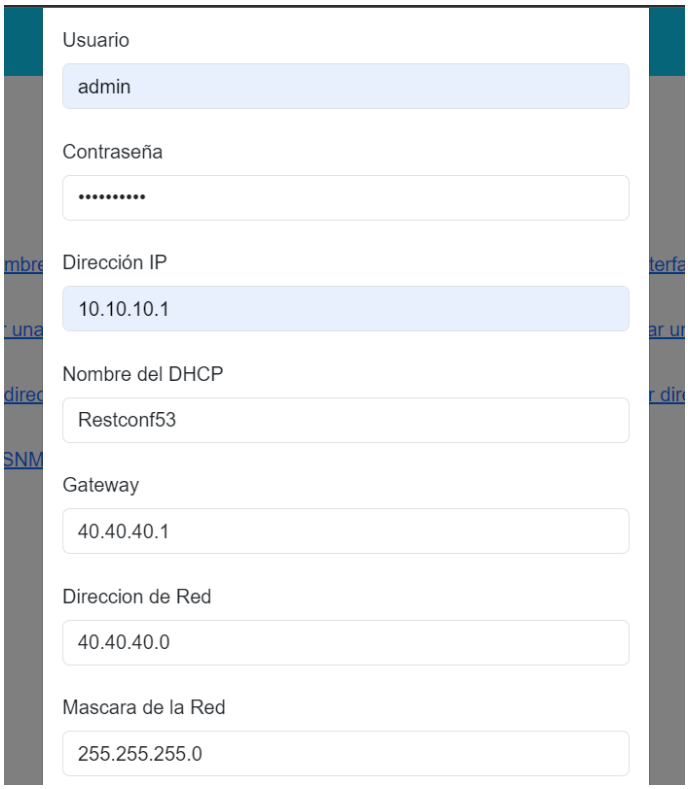

Figura 3. 7 Insertar DHCP parte 1

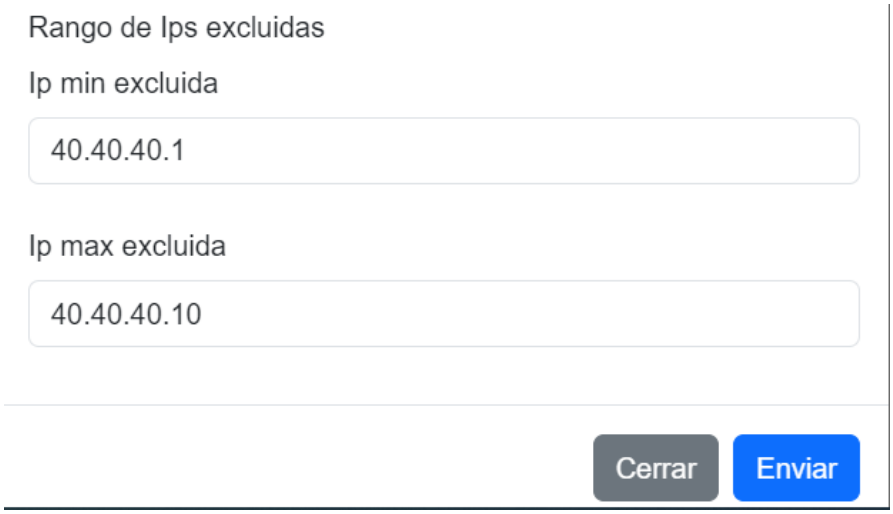

Figura 3. 8 Insertar DHCP parte 2

Como paso siguiente se tiene la visualización de las configuraciones realizadas por el usuario con lo cual se da inicio a un nuevo módulo que permite visualizar obtener información del usuario.

3.2.2. Módulo de Información

Este módulo permite obtener diferentes datos, la información obtenida se basa en la opción que seleccione el usuario, como se aprecia en la Figura 3. 9, se puede mostrar algo sencillo como el nombre, el formato que se visualiza es JSON.

Figura 3. 9 Mostrar nombre

La información que se visualiza puede cambiar como es el caso de interfaces donde se aprecia toda la información de estas. Como se aprecia el resultado obtenido en este módulo está limitado solo a las opciones que el usuario tiene como ejemplos de los diferentes tamaños de texto obtenidos mediante la compilación de este módulo se refleja en la Figura 3. 10 y Figura 3. 11.

Figura 3. 10 Mostrar Interfaces

Figura 3. 11 Mostrar DHCP

Por último, la información mostrada al usuario puede ser procesada para generar un reporte por lo cual se da paso al último enfoque del desarrollo de la NMS.

3.2.3. Módulo de Reportes

La finalidad de este módulo es poder exportar los datos informativos de los dispositivos de red a un pdf, el procedimiento se describirá de la siguiente manera en la Figura 3. 12, genera un reporte mediante el ingreso de los datos, la Figura 3. 13 se observa el pdf descargado en la carpeta del código, por último, la Figura 3. 14 se muestra la información generada para el reporte.

Figura 3. 12 Generar un reporte

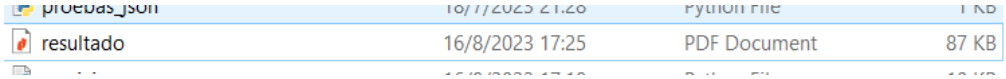

Figura 3. 13 Descarga del reporte

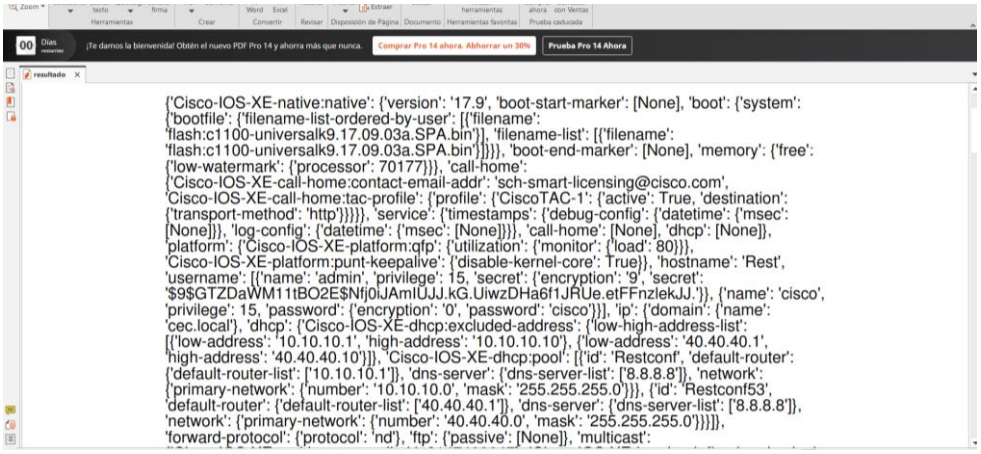

Figura 3. 14 Visualización del reporte

Para comprobar que los paquetes enviados por RESTCONF, se utiliza el software Wireshark, el cual permite interceptar los paquetes existentes en la comunicación entre la NMS y el dispositivo de red. En la Figura 3. 15 se observa la descripción de la comunicación mediante el protocolo de internet en lo cual se recalca las direcciones IP utilizadas para establecer él envió de los datos, se puede apreciar que no se fragmentó el mensaje, además de las direcciones MAC.

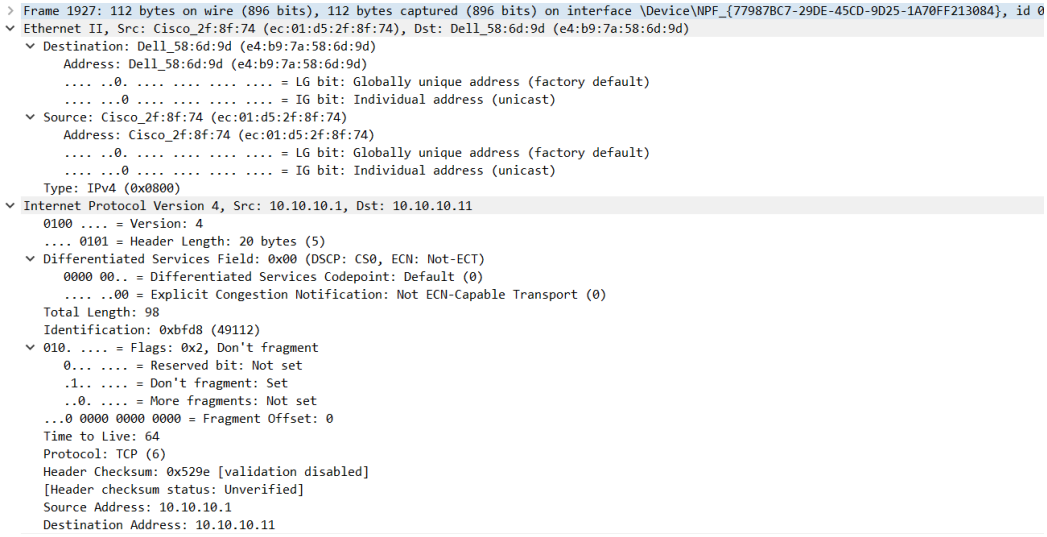

Figura 3. 15 Captura Wireshark parte 1

En la Figura 3. 16, el paquete capturado indica el puerto que utiliza la NMS para enviar el mensaje y el puerto de llegada, se observa las banderas, por último, tenemos el *Checksum* para recuperar el mensaje en caso de errores y el tamaño de ventana para enviar el paquete.

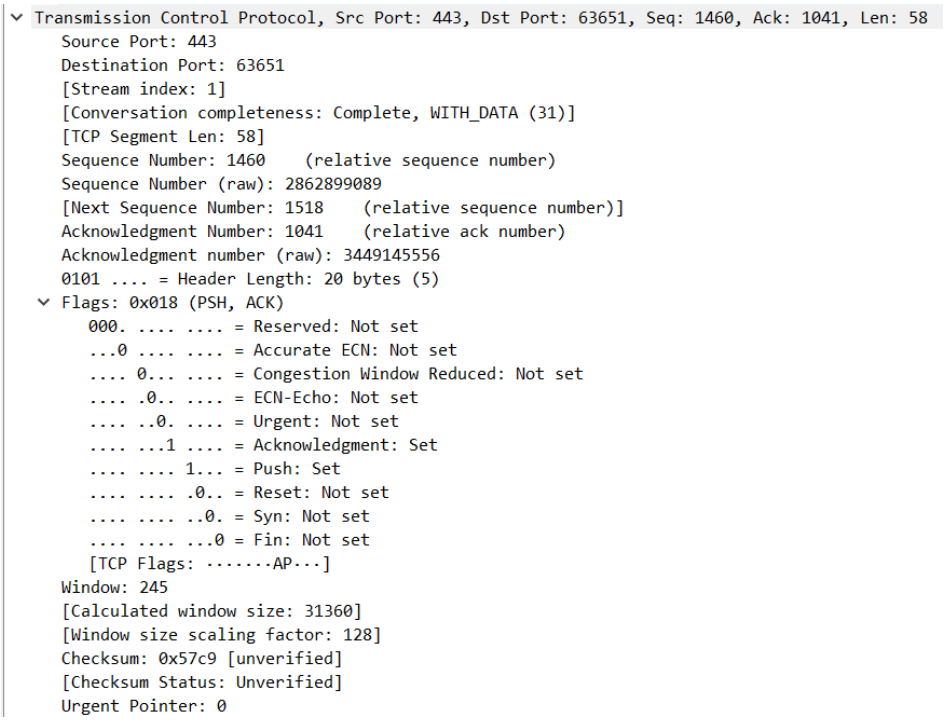

Figura 3. 16 Captura Wireshark parte 2

En la Figura 3. 17, se describe la cantidad de datos mediante el *payload*, una característica importante del paquete capturado es la Data TCP la cual se encuentra encriptada, mediante el protocolo *Hypertext Transfer Protocol*, el cual tiene la versión 1.2. Esto demuestra que RESTCONF utiliza la comunicación segura, se puede intentar descifrar los datos para poder ver en texto claro la información.

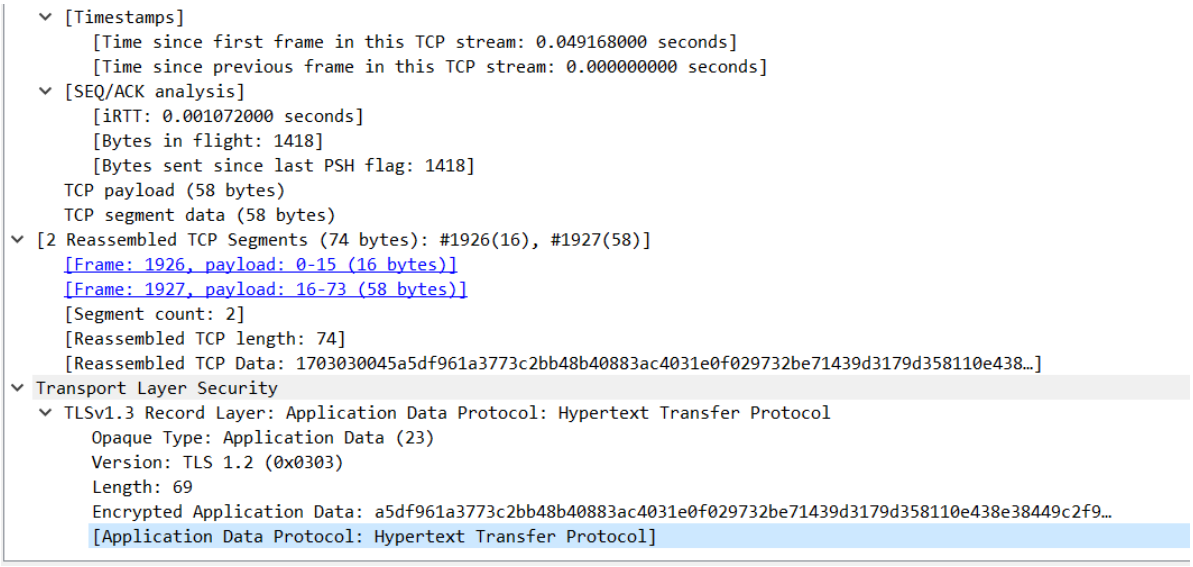

Figura 3. 17 Captura Wireshark parte 3

3.3. Conclusiones

Para trabajar con el protocolo RESTCONF es necesario saber si el dispositivo de red genera certificados para HTTP y HTTPS, como es el caso de los equipos de la marca CISCO. Esto con la finalidad de evitar errores al realizar solicitudes porque sin el certificado no se pueden realizar consultas del tipo REST.

Con RESTCONF es necesario entender el tipo de solicitud, que se envía, esto por motivo de que existen diversos tipos de peticiones como son GET, POST, PATCH entre otros, esto por motivo de que algunos módulos de las bibliotecas YANG no soportan todos los tipos de consulta REST.

El trabajar con servidores web, facilita la comunicación con el dispositivo de red, la principal razón es que RESTCONF no puede mantener sesiones abiertas por lo cual el envió de datos o la recepción de estos, solo sucede cuando se realiza una solicitud por lo cual no es necesario usar muchos recursos de memoria, lo cual disminuye la carga de trabajo de un servidor.

Las bibliotecas disponibles en el lenguaje de programación de Python permiten realizar solicitudes de tipo REST sin mayores complicaciones a la hora de cargar datos, pues la estructura se vuelve simple al combinarse con RESTCONF, entonces se debe definir el módulo del protocolo, la carga útil, es decir los parámetros de configuración y por último el tipo de solicitud.

El envío de datos mediante el método POST, garantiza una comunicación confiable y segura, por lo cual las bibliotecas Flask y JQuerry son óptimas para

mantener este tipo de comunicación porque permiten en su configuración permite establecer como único método de envió de datos a POST esto se aplicará a todas las solicitudes realizadas, con lo cual se garantiza que el usuario y el sistema se comuniquen de manera segura.

RESTCONF es un protocolo que trabaja mediante la comunicación segura utilizando el puerto 443, pero Wireshark no soporta el protocolo, por lo cual se puede utilizar el puerto como filtro, para poder verificar que se está cumpliendo el envío de información mediante HTTPS, además de poder observar los datos encriptados que son características de este modo de comunicación.

Al trabajar con la biblioteca REST, es importante considerar las repuestas cuando tenemos mensajes de OK o negación, esto es por la variación de los códigos que existe entre un método u otro para enviar mensajes relacionados al éxito o fracaso del cambio que se solicitó.

El trabajar los mensajes que proporciona el dispositivo de red como *string* es una buena práctica, para evitar errores por lo cual se debe tener ciertas consideraciones cuando se muestra los mensajes en la interfaz web, porque al mostrar un *string* lo que el usuario visualizara es una cadena que se expande a lo ancho de la pantalla y no una información ordenada que sea de mayor entendimiento para el usuario.

Para utilizar JQuery es importante entender el formato en el que se recepta la respuesta de una solicitud, pues este tipo de bibliotecas trabaja con formatos predefinidos, esto con la finalidad de evitar errores por que el intérprete no entiende lo que estamos enviando o recibiendo.

Las clases creadas con Python facilitan el desarrollo de constructores, y la herencia se debe realizar mediante el uso del comando *self,* el cual permite utilizar los métodos o variables globales de la clase.

Para construir objetos que sean reutilizables es importante identificar los datos que constantemente se repiten como es el caso de usuario, contraseña, la dirección IP del dispositivo CISCO y en el encabezado con la referencia al lenguaje YANG, esto permite el desarrollo de un constructor, pues son datos que estrictamente el usuario debe proporcionar para realizar una solicitud.

Los módulos YANG son únicos para cada solicitud, esto porque cuenta con características únicas como ejemplo si solicitamos cambiar el nombre el módulo utilizado solo requiere el nombre, pero si el usuario desea realizar cambios más complejos el módulo solicita más datos para evitar errores.

Las solicitudes REST son dependientes del módulo por lo cual se debe desarrollar un método para cada tipo de solicitud y llamarlo desde el método que lo necesite.

Para solicitudes tipo GET es importante considerar que el cambio de módulo es el único causante de tener diferentes resultados, por lo cual se pueden agrupar y definir cuales módulos YANG, son importantes para el desarrollo de la NMS,

esto porque la información o los reportes generados necesitan cambiar en función de lo requerido por el usuario.

La NMS en un servidor web permite un soporte más amplio, esto porque diferentes usuarios pueden acceder al sistema para realizar el monitoreo de los equipos de red, con lo cual se puede obtener una continuidad del negocio y prevenir incidentes, además de poder realizar cambios solicitados de manera urgente.

Al realizar múltiples consultas se tiene un problema en la memoria temporal del servidor por lo que es importante refrescarlo para evitar errores cuando se desea realizar algún otro cambio por parte del usuario.

POSTMAN es un software que permite trabajar con APIs, por lo cual es importante para realizar pruebas de los módulos YANG, debido a que no necesita una programación adicional, solo el usuario debe conocer como está estructurado el módulo YANG y que tipo de solicitud soporta, aunque se puede realizar diferentes pruebas para que la implementación de los módulos en la NMS de una manera fácil y sencilla.

3.4. Recomendaciones

La programación orientada a objetos es muy útil cuando tenemos datos de configuración que constantemente se repiten, por lo cual se recomienda utilizar constructores encargados de manejar este tipo de datos para optimizar y mejorar la calidad del código a presentar.

El uso de métodos que simplifiquen el envió de datos es una buena práctica, por lo cual se recomienda agrupar la información de forma ordenada, como ejemplo en el trabajo desarrollado todas las consultas de tipo GET se agrupan mediante la sentencia *case*, la cual permite seleccionar el módulo que se necesite y realiza la consulta al equipo de red.

La conversión de formatos es una consideración importante a la hora de mostrar resultados por lo cual se recomienda, convertir los datos a un solo formato para el trabajo y la transacción entre métodos, cuando se muestra el resultado al usuario se puede convertir a un formato de visualización como JSON.

Al utilizar *self* se recomienda diferencia lo que es una clase y lo que es un archivo común de Python, esto por motivo de que un archivo común no necesita esa palabra reservada solo las clases la utilizan.

Una desventaja de Flask es que siempre debe retornar una ruta o un mensaje, por lo cual se recomienda que los métodos configurados se direccionen a otra plantilla o a su vez retornar un mensaje, esto evita errores cuando se compile la solución.

Los módulos YANG son únicos, por lo cual se recomienda que en el método desarrollado se incluye una variable que almacene este módulo para evitar errores.

Los datos enviados en una solicitud de tipo REST necesita un formato JSON, por lo cual es importante considerar que algunos tipos de variables como *null* no están disponibles en Python por lo que se recomienda buscar módulos que no contengan este valor para evitar errores o a su vez utilizar opciones para generar la variable que se necesite.

El servidor web, permite tener diferentes usuarios, pero es importante considerar las características del equipo, por lo cual se recomienda que, para los módulos de información y reportes, una vez realizada la consulta se reinicie el servicio para liberar recursos, evitando errores al enviar una nueva solicitud.

4. Bibliografía

[1] Cisco, «YANG Data Modelling and NETCONF,» *Cisco Live,* nº 2032, pp. 42-50, 2020.

[2] Pallets, «Flask,» Sphinx 7.0.1., 2010. [En línea]. Available: https://flask.palletsprojects.com/en/2.3.x/. [Último acceso: 22 mayo 2023].

[3] B. Hadzhiev, «bobbyhadz blog,» 18 Febrero 2023. [En línea]. Available: https://bobbyhadz.com/blog/python-convert-json-null-to-none#:~:text=Use%20the%20json.,values%20to%20None%20in%20Python.. [Último acceso: 20 Junio 2023].

[4] Python, «Python,» [En línea]. Available: https://pythonprogramming.net/practical-flask-introduction/. [Último acceso: 22 Mayo 2023].

[5] V. Vázquez, «Automatización de redes: Un caso práctico,» Universidae da Coruña, Coruña, 2022.

[6] A. Prieto, A. Leung y K. Rockwell, «Automating the Testing of RESTCONF Agents,» Indiana, 2015.

[7] L. Domínguez, «Monitorización de Red usando Cisco IOS XE,» Universidad Politécnica de Madrid, Madrid, 2022.

[8] M. Granderath y J. Schonwalder, «A Resource Efficent Implementation of the RESTCONF Protocol for OpenWrt Systems,» IEEE, Bremen, 2020.

[9] B. YumaWorks y M. Bjorklund, «RESTCONF Protocol,» Enero 2017. [En línea]. Available: https://datatracker.ietf.org/doc/html/rfc8040. [Último acceso: 12 Junio 2023].

[10] C. Jiménez, UML Aplicaciones en Java y C++, Madrid: RA-MA, 2015.

[11] A. Downey, J. Elkner y C. Meyers, Aprenda a Pensar Como un Programador con Python, Boston: GreenTeaPress, 2002.

[12] S. Delgado, Aprende Python, CreativeCommonsReconocimiento4.0Internacional, 2023.

5. ANEXOS

5.1. Protocolos

5.1.1. Nat

Permite que redes de ordenadores utilicen un rango de direcciones especiales (IPs privadas) y se conecten a Internet usando una única dirección IP (IP pública), es decir permite traducir las direcciones para que sea posible una comunicación. Para nuestra practica es necesario activar este servicio porque el portal cautivo solo se activa cuando existe el servicio de internet activado, por lo cual seguimos el siguiente procedimiento para realizar su configuración.

- En la interfaz del router, por el cual debe ingresar todo el tráfico es decir la que nos conecta a la red interna creada para esto se debe ingresar el siguiente comando:
  - ip nat inside
- En la interfaz del router, por el cual debe salir todo el tráfico hacia Internet es decir aquí se traducirá las direcciones Ip para garantizar el acceso al servicio por lo cual debemos ingresar el siguiente comando:
  - Ip nat outside
- Lo siguiente a realizar es crear una lista de acceso, en esta lista ingresaremos las direcciones IP de las redes o red que están configuradas, cabe recalcar que la máscara de red se ingresara remplazando los 0 por 1, para ejemplificarlo usaremos la red 10.10.10.0 y mascara 24 esto lo realizamos con el siguiente comando:
  - access-list 1 permit [dirección Ip de red]10.10.10.0 [mascara]0.0.0.255
- Por último, debemos especificar cuál es la entrada y salida, esto con la finalidad de indicarle al router que realice la traducción y direccionamiento, para garantizar el servicio de Internet esto lo realizamos con el siguiente comando:
  - ip nat inside source list 1 interface GigabitEthernet0/0/1 overload

  Cabe recalcar que para utilizar el comando descrito previamente la entrada será especificada por la lista de acceso creada y la interfaz que seleccionamos como salida, el ultimo parámetro es indicarla la forma como debe trabajar para nuestro caso con sobrecarga porque utilizamos una sola dirección IP para la traducción.

5.1.2. DHCP

Este protocolo permite obtener direcciones IP de un rango, para esto es necesario definir una red y las direcciones Ip que excluiremos, esto con la finalidad de poder fijar direcciones en caso de ser necesario para realizar esta tarea se debe utilizar los siguientes comandos:

- Lo primero es decidir las direcciones a excluir esto con dos fines, el primero establecer una red fija para el Gateway o a su vez otras para servidores en caso de tenerlos, pues los clientes no necesitan una red fija esto se realiza mediante el siguiente comando:
    - Ip dhcp excluded-address 10.10.10.1 10.10.10.10

  Con esto conseguimos excluir 10 direcciones IP, para el ejercicio propuesto utilizamos una dirección para el Gateway.
- Lo siguiente a realizar es definir un nombre para nuestra piscina de direcciones para el ejercicio se seleccionó como nombre RESTCONF esto se realiza con el siguiente comando.
    - Ip dhcp pool RESTCONF
- Como siguiente paso se debe definir la red, una recomendación en este punto es que se debe considerar las direcciones IP excluidas anteriormente pues esta red debe escribirse con el siguiente comando:
    - Network 10.10.10.0 255.255.255.0

  Algo importante es el 0 al final de los 3 primero octetos este número define la red, así que se debe tener cuidado si realizamos subnetting por motivo de que la máscara definirá si tenemos más de una red, para la práctica se estableció trabajar con una red utilizando máscara 24.
- El siguiente paso es definir la dirección IP del Gateway por motivo de que nuestros equipos al conectarse hacia el router necesitan saber dónde llegar esto lo definimos mediante el siguiente comando.
    - Default-router 10.10.10.1

  Como buena práctica en el proyecto se utiliza el número 1 para definir el Gateway y las demás direcciones excluidas para servidores.
- El siguiente punto es opcional, pero para garantizar la navegación en internet del usuario y el equipo, utilizar un servidor de DNS ayudara a garantizar el servicio esto lo realizamos con el siguiente comando:
    - Dns-server 8.8.8.8

5.2. Tablas de caso de uso

5.2.1. Módulo de configuraciones

Caso de Uso	Cambiar el nombre del equipo
Actores	Usuario, Dispositivo Cisco
Resumen	El usuario cambiar el nombre del dispositivo Cisco, para lo cual el nuevo nombre será digitado por el usuario.
Precondiciones	El usuario debe haber seleccionado en el menú principal la opción configuraciones
Postcondiciones	
Incluye	
Extiende	
Hereda de	
Flujo de Eventos	
Actor	Sistema
El usuario selecciona la opción de Configuraciones. El usuario selecciona la opción de Cambiar nombre El usuario ingresa la información solicitada en la ventana. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco genera una respuesta.	El sistema muestra una ventana con los datos requeridos (nombre, dirección IP, usuario y contraseña) para crear una VLAN. El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema construye el cuerpo de información y la URL. El sistema envía la información. El sistema recibe la respuesta y envía un mensaje de OK o error

Caso de Uso	Crear interfaz de Loopback
Actores	Usuario, Dispositivo Cisco
Resumen	El usuario creará una nueva interfaz de Loopback, donde podrá asignarle una dirección IP v4, con su respectiva mascará, esos parámetros serán configurados por el usuario.
Precondiciones	El usuario debe haber seleccionado en el menú principal la opción configuraciones
Postcondiciones	
Flujo de Eventos	
Actor	Sistema
El usuario selecciona la opción de Configuraciones El usuario selecciona la opción de Configurar interfaz de Loopback El usuario ingresa la información solicitada en la ventana.	El sistema muestra una ventana con los datos requeridos (nombre, dirección IP, mascara de red, usuario y contraseña) para crear una Interfaz de Loopback. El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema construye el cuerpo de información y la URL.

El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco genera una respuesta.	El sistema envía la información. El sistema recibe la respuesta y envía un mensaje de OK o error.

Caso de Uso	Configurar SNMP
Actores	Usuario, Dispositivo Cisco
Resumen	El usuario creará una nueva comunidad SNMP a la cual se le puede otorgar permisos de lectura y escritura en función de la necesidad del usuario, con lo cual se puede realizar pruebas de MIBs, cabe recalca que para el prototipo se consideró la versión 2 del protocolo.
Precondiciones	El usuario debe haber seleccionado en el menú principal la opción configuraciones.
Postcondiciones	
Incluye	
Extiende	
Hereda de	
Flujo de Eventos	

Actor	Sistema
El usuario selecciona la opción de Configuraciones. El usuario selecciona la opción de Configurar SNMP. El usuario ingresa la información solicitada en la ventana. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco genera una respuesta.	El sistema muestra una ventana con los datos requeridos (comunidad, permisos, dirección IP, usuario y contraseña) para crear una comunidad SNMP. El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema construye el cuerpo de información y la URL. El sistema envía la información. El sistema recibe la respuesta y envía un mensaje de OK o error.

Caso de Uso	Asignar una dirección IP
Actores	Usuario, Dispositivo Cisco
Resumen	El usuario tendrá la capacidad de ingresar la dirección IP a una interfaz del dispositivo CISCO cabe recalcar que debido a las características de los equipos con lo que se trabaja no todas las interfaces permitirán el ingreso de direcciones IP, por lo cual la visualización de interfaces deberá ser limitada.
Precondiciones	El usuario debe haber seleccionado en el menú principal la opción configuraciones
Postcondiciones	
Flujo de Eventos	

Actor	Sistema
El usuario selecciona la opción de Configuraciones. El usuario selecciona la opción de Asignar una dirección IP. El usuario ingresa la información solicitada en la ventana. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco genera una respuesta.	El sistema muestra una ventana con los datos requeridos (nombre, dirección IP de la interfaz, mascara de red, dirección IP, usuario y contraseña) para asignar una dirección IP a una interfaz. El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema construye el cuerpo de información y la URL. El sistema envía la información. El sistema recibe la respuesta y envía un mensaje de OK o error

Caso de Uso	Crear un DHCP
Actores	Usuario, Dispositivo Cisco
Resumen	El usuario creará un nuevo pool DHCP, el cual podrá ser asignada a una interfaz o una VLAN en el caso de Wireless o a su vez en una interfaz determinada, el usuario tendrá la capacidad de limitar un rango de IPs las cuales se encontrarán excluidas del pool a su vez podrá configurar cual es la red de salida, la red y la máscara.
Precondiciones	El usuario debe haber seleccionado en el menú principal la opción configuraciones
Postcondiciones	
Incluye	
Extiende	
Hereda de	
Flujo de Eventos	

Actor	Sistema
El usuario selecciona la opción de Configuraciones El usuario selecciona la opción de Crear un DHCP. El usuario ingresa la información solicitada en la ventana. 9. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco genera una respuesta.	El sistema muestra una ventana con los datos requeridos (nombre DHCP, red, mascara de red, dirección mínima excluida, dirección máxima excluida, dirección por defecto, dirección IP usuario y contraseña) para crear un DHCP. El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema construye el cuerpo de información y la URL. El sistema envía la información. El sistema recibe la respuesta y envía un mensaje de OK o error

5.2.2. Módulo de información.

Caso de Uso	Mostrar información de las interfaces
Actores	Usuario, Dispositivo Cisco

Resumen	El usuario tendrá la capacidad de visualizar todas las interfaces del equipo Cisco, esto incluye las interfaces Loopback, con lo cual el usuario podrá visualizar las configuraciones que tengan estas.
Precondiciones	El usuario debe haber seleccionado en el menú principal la opción información.
Postcondiciones	La ventana de información debe mostrar la información de las interfaces con el cual está trabajando
Incluye	
Extiende	
Hereda de	
Flujo de Eventos	
Actor	Sistema
El usuario selecciona la opción de información. El usuario ingresa el usuario, la contraseña y la dirección IP del equipo del que se obtendrá información. El usuario selecciona la opción de Mostrar información de las interfaces. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco envía la respuesta de lo solicitado.	El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema selección el módulo para realizar la consulta y construye la URL. El sistema envía la información. El sistema recibe la respuesta a lo solicitud realizada. El sistema transforma los datos a formato JSON. El sistema guarda la información en una variable El sistema muestra al usuario la información que ha procesado y almacenado.

Caso de Uso	Mostrar VLANs
Actores	Usuario, Dispositivo Cisco
Resumen	El usuario tendrá la capacidad de visualizar las VLANs con las que cuente el equipo de red Cisco, además el usuario podrá ver si las configuraciones realizadas del módulo anterior se ingresaron con éxito.
Precondiciones	El usuario debe haber seleccionado en el menú principal la opción información.
Postcondiciones	La ventana de información debe mostrar las VLANs con las cuales está trabajando
Incluye	
Extiende	
Hereda de	
Flujo de Eventos	
Actor	Sistema

El usuario selecciona la opción de información. El usuario ingresa el usuario, la contraseña y la dirección IP del equipo del que se obtendrá información. El usuario selecciona la opción de Mostrar VLANs. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco envía la respuesta de lo solicitado.	El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema selección el módulo para realizar la consulta y construye la URL. El sistema envía la información. El sistema recibe la respuesta a lo solicitud realizada. El sistema transforma los datos a formato JSON. El sistema guarda la información en una variable El sistema muestra al usuario la información que ha procesado y almacenado.

Caso de Uso	Mostrar DHCP
Actores	Usuario, Dispositivo Cisco
Resumen	El usuario podrá visualizar los pools DHCP que tenga configurados, donde se puede ver características como la red la dirección IP por defecto, las direcciones excluidas, entre otras.
Precondiciones	El usuario debe haber seleccionado en el menú principal la opción información.
Postcondiciones	La ventana de información debe mostrar los pools DHCP con los cuales se encuentra trabajando
Incluye	
Flujo de Eventos	
Actor	Sistema
El usuario selecciona la opción de información. El usuario ingresa el usuario, la contraseña y la dirección IP del equipo del que se obtendrá información. El usuario selecciona la opción de Mostrar DHCP. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco envía la respuesta de lo solicitado.	El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema selección el módulo para realizar la consulta y construye la URL. El sistema envía la información. El sistema recibe la respuesta a lo solicitud realizada. El sistema transforma los datos a formato JSON. El sistema guarda la información en una variable El sistema muestra al usuario la información que ha procesado y almacenado.

Caso de Uso	Mostrar información de SNMP
Actores	Usuario, Dispositivo Cisco

Resumen	El usuario tendrá la capacidad de visualizar las comunidades snmp que se encuentran configuradas en el equipo para su posterior uso en monitoreo de las redes.
Precondiciones	El usuario debe haber seleccionado en el menú principal la opción información.
Postcondiciones	La ventana de información debe mostrar las comunidades con las cual puede trabajar el dispositivo Cisco.
Flujo de Eventos	
Actor	Sistema
El usuario selecciona la opción de información. El usuario ingresa el usuario, la contraseña y la dirección IP del equipo del que se obtendrá información. El usuario selecciona la opción de Mostrar información de SNMP. El dispositivo Cisco recibe la información y la procesa. El dispositivo Cisco envía la respuesta de lo solicitado.	El sistema recoge toda la información, comprueba que los campos no estén vacíos. El sistema selección el modulo para realizar la consulta y construye la URL. El sistema envía la información. El sistema recibe la respuesta a lo solicitud realizada. El sistema transforma los datos a formato JSON. El sistema guarda la información en una variable El sistema muestra al usuario la información que ha procesado y almacenado.

5.3. Código terminado

Para visualizar el código desarrollado se debe acceder al siguiente enlace.

https://github.com/gustavoxa/Program.git

Como el desarrollo de la NMS utilizo el lenguaje de programación Python por lo cual una vez se descargue el código fuente el equipo donde se desee ejecutar el programa se debe utilizar el siguiente comando en el cmd o powershell en el caso de Windows.

py index.py