ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DESARROLLO DE UN SISTEMA DE TELEFONÍA IP DISTRIBUIDO MEDIANTE LA IMPLEMENTACIÓN DE UN MECANISMO DE DESCUBRIMIENTO DE RUTAS DE LLAMADAS, EN BASE AL SISTEMA OPERATIVO LINUX

PROYECTO PREVIO A LA OBTENCIÓN DEL TITULO DE INGENIERO EN ELECTRÓNICA Y REDES DE INFORMACIÓN

ANA FERNANDA RODRÍGUEZ HOYOS

ana.fernanda.rodriguez@gmail.com

DIRECTOR: ING. TARQUINO SÁNCHEZ, MBA

tsanchez@mailfie.epn.edu.ec

Quito, septiembre 2010

DECLARACIÓN

Yo, Ana Fernanda Rodríguez Hoyos, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Ana Fernanda Rodríguez Hoyos

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Ana Fernanda Rodríguez Hoyos, bajo mi supervisión.

Ing. Tarquino Sánchez, MBA

DIRECTOR DEL PROYECTO

PRESENTACIÓN

En la actualidad los Sistemas de Voz sobre IP son esenciales para la evolución del Sector de las Comunicaciones. A nivel empresarial VoIP ofrece varias ventajas como: reducción de costos en las llamadas y la implementación de servicios de telefonía adicionales. La implementación de dichos servicios utilizando la red de telefonía tradicional resultaría un proceso más costoso y complejo de realizar.

En el presente proyecto se estudia una solución de telefonía IP analizando los conceptos de redundancia, movilidad de los usuarios y confiabilidad, para prestar un servicio completo a una red WAN. A diferencia de la solución planteada, las opciones que existen en el mercado para brindar similares servicios de telefonía IP poseen un precio elevado, por ello se plantea el uso del Sistema Operativo Linux como alternativa de bajo costo al ser un software gratuito, además de la fiabilidad de sus aplicaciones, la flexibilidad en el desarrollo de sus módulos y la facilidad de integración con otras tecnologías.

En estos tiempos en los que la proliferación de las redes de datos ha avanzado con más rapidez en las empresas privadas y públicas, el uso de herramientas de virtualización, especialmente en el mundo *open source* resulta de gran utilidad para el estudio previo de nuevas tecnologías, servicios y protocolos sobre esquemas de red físicos, antes de ser implementados. En este sentido la virtualización es favorable para reproducir un entorno de pruebas que permita analizar los resultados del funcionamiento de un sistema, antes de ponerlos en producción con el fin de mejorarlo o encontrar sus falencias.

RESUMEN

El presente proyecto describe el desarrollo y la implementación de un prototipo para un sistema de telefonía IP distribuido sobre una red en malla formada por un total de ocho nodos, los cuales simulan estar localizados en distintas ciudades del Ecuador. Cada nodo en la red cumple con la función de router a nivel de red y de central de telefonía IP basada en Asterisk. El sistema de telefonía IP brinda la ventaja de ser redundante en caso de fallas en la red, además de proveer movilidad al usuario entre las diferentes centrales de telefonía IP. El contenido del proyecto se ha dividido en cinco capítulos.

En el capítulo I se realiza un estudio detallado de los conceptos básicos involucrados en el diseño del prototipo de sistema de telefonía IP, como protocolos de inicio de sesión, CODECS de manejo de voz, troncalización de PBX, protocolos de enrutamiento, etc. Además de realizar un análisis comparativo entre *ENUM* y el protocolo *DUNDi* como mejor alternativa para el descubrimiento de rutas de llamadas.

En el capítulo II se lleva a cabo el diseño del prototipo del sistema de telefonía IP, se analiza los requerimientos del sistema en función de la cantidad de centrales IP que se interconectarán para prestar servicio a los usuarios, y se detalla la solución que se va a implementar. Además de realizar un estudio comparativo entre varias herramientas de virtualización para implementar el prototipo de red en busca de la mejor opción. Al final se describe el funcionamiento detallado del prototipo.

En el capítulo III se realiza la implementación y pruebas de funcionamiento del prototipo. Se detalla el proceso de instalación del Sistema Operativo Linux que funcionará en la máquina física, la instalación y configuración de la herramienta de virtualización seleccionada, el desarrollo del script *XML* para implementar el

esquema de red propuesto, la configuración del enrutamiento entre los nodos de red, instalación y configuración básica de los servicios de telefonía IP en los nodos, registro dinámico de los usuarios en las centrales telefónicas y configuración del sistema de descubrimiento de rutas de llamada en base al protocolo *DUNDi* en Asterisk. Terminada la configuración se describe el esquema de pruebas que se realiza sobre el prototipo para probar su correcto funcionamiento, simulando escenarios de falla de red y empleando para la generación de tráfico IP la herramienta *SIPP*, la cual permite simular mensajes SIP.

En el capítulo IV se evalúa el costo referencial que tiene el implementar el prototipo del sistema de telefonía IP en un ambiente real, de acuerdo al esquema planteado con los equipos requeridos.

Finalmente en el capítulo V se analizan los resultados obtenidos de las pruebas de funcionamiento del prototipo, con el fin de formular las respectivas conclusiones y recomendaciones del mismo.

AGRADECIMIENTOS

Agradezco a Dios por brindarme vida y salud junto a mi familia, por su amor y protección incondicional.

A mis padres Inés Hoyos y Héctor Rodríguez por su dedicación, por su amor, apoyo y por la mejor herencia la educación. Gracias papis por inculcarme los mejores valores el respeto, la responsabilidad, la constancia, la honestidad y la tolerancia.

A mis profesores, por sus enseñanzas, consejos y la mejor herramienta de trabajo el conocimiento. En especial mis agradecimientos al Ing. Tarquino Sánchez y al Ing. José Antonio Estrada por su guía en la elaboración de este proyecto de titulación.

A mis amigos por compartir las mejores experiencias de nuestro paso por la Universidad, por estar conmigo en las pruebas, las amanecidas, los proyectos en fin la vida politécnica, gracias por todo: Gaby, Nelita, Danilo, Mony, Ramiro y Richard. A la vez mi gratitud a mis compañeros de trabajo Christian, Iván, Luis Felipe e Ivancito.

A mis tías Matilde, Clarita y Jovita por su apoyo y ayuda incondicional a la distancia, gracias queridas tías por estar pendientes de mis estudios y confiar en mí.

DEDICATORIA

Este proyecto va dedicado a la memoria de mi querido hermano Héctor Alejandro Rodríguez Hoyos, gracias ñañito por brindarme los mejores catorce años de mi vida, gracias por ser un ejemplo de constancia, por demostrarme con tu vida que no existe obstáculo que no se pueda superar de la mano de Dios. La confianza y amor que me brindaste son el mejor recuerdo en los momentos de debilidad, gracias por tus consejos, tu sonrisa, travesuras, por tu existencia que cambió nuestras vidas y le dio sentido a la familia Rodríguez Hoyos. Tu presencia siempre nos mantendrá unidos por el amor y la ternura que sembraste en nuestros corazones, gracias por ser el mejor regalo y lección de vida que Dios nos pudo dar.

CAPÍTULO I

FUNDAMENTOS TEÓRICOS

1.1 VOZ SOBRE IP (VoIP)

1.1.1 DEFINICIÓN

Voz sobre IP es una tecnología que permite la transmisión de voz sobre Internet, es decir la voz pasa a través de redes IP en forma digital como paquetes de datos hacia una dirección IP determinada. Al utilizar VoIP se puede hacer y recibir llamadas telefónicas ordinarias a través de una conexión a internet, aprovechando la infraestructura de la red datos.

En la actualidad los Sistemas de Voz sobre IP son esenciales para la evolución del Sector de las Comunicaciones. A nivel empresarial VoIP ofrece varias ventajas como: reducción de costos en las llamadas y la implementación de servicios de telefonía adicionales. La implementación de dichos servicios utilizando la red *PSTN*[1] tradicional resultaría un proceso más costoso y complejo de realizar. Se estima que para el año 2010 la cuarta parte de las llamadas mundiales se realizarán con esta tecnología.

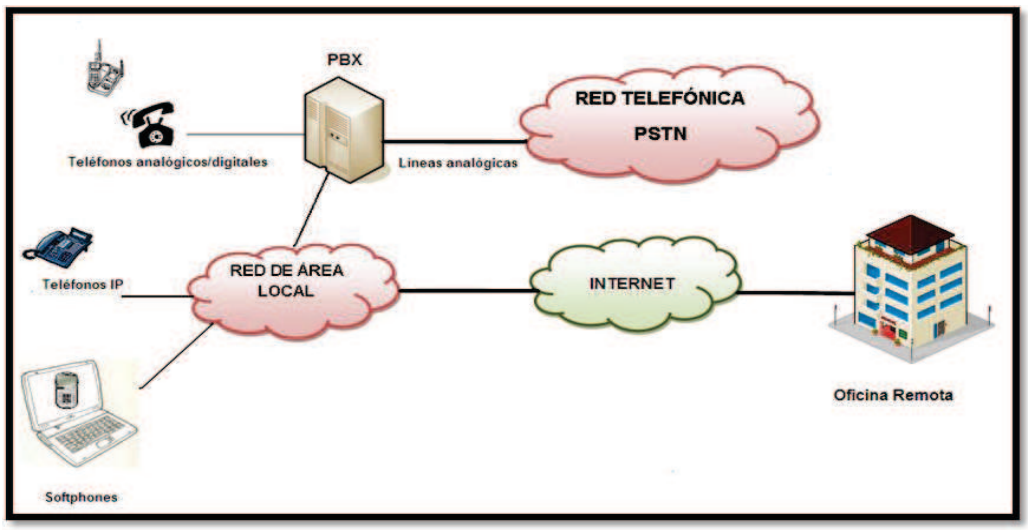


Figura 1.1 Tráfico de Voz sobre IP

1.1.2 CARACTERÍSTICAS

- Voz sobre IP no es un servicio al contrario es una tecnología, la cual emplea el Protocolo de Internet IP para la transmisión de voz. A través de esta tecnología se puede brindar servicios como: Telefonía IP y videoconferencia.

- VoIP es una tecnología desarrollada con estándares abiertos e internacionales, su ventaja es el ahorro de ancho de banda y el uso más efectivo de un canal de comunicación al aprovechar los intervalos entre ráfagas de datos, para el envío de paquetes de voz de forma gratuita.

- Permite convivir a la telefonía IP junto a la telefonía tradicional, a la vez de incorporar otros servicios especiales en la red como video y *TPV*[2].

- Permite controlar el tráfico de la red, para asegurar el óptimo rendimiento de la red, como en el caso de empresas que tienen conexiones intranet con otras sucursales.

1.2 TELEFONÍA IP

1.2.1 DEFINICIÓN

La telefonía IP es un servicio que permite al usuario tener comunicaciones de voz por medio de redes IP. La telefonía IP se desarrolla como alternativa tecnológica frente a la telefonía tradicional a nivel corporativo y empresarial, con el fin de aprovechar los recursos de red y obtener beneficios económicos.

PSTN [1](*Public Switched Telephone Network*) La red telefónica pública conmutada, es una red con conmutación de circuitos tradicional optimizada para comunicaciones de voz en tiempo real.

TPV [2](*Terminal Punto de Venta*) Hace referencia al dispositivo y tecnologías que ayudan en las tareas de gestión de un establecimiento comercial de venta al público.

1.2.2 FUNCIONAMIENTO DE LA TELEFONÍA IP

La telefonía IP cumple las siguientes fases:

Digitalización de la voz: Conversión de la señal de voz analógica a digital. Este proceso involucra las fases de muestreo, cuantificación y codificación de la señal.Los dispositivos utilizados en esta conversión son llamados CODECS los cuales además de realizar la conversión de la señal se encargan de comprimir la información y proporcionan la cancelación de eco. La utilización de mayor o menor ancho de banda en la transmisión depende del CODEC que se utilice.Los teléfonos IP y centrales IP soportan unaserie de CODECS cada uno. Cuando hablan entre sí negocian un CODEC común. Entre los CODECS más utilizados actualmente se encuentran:

NOMBRE	DESCRIPCIÓN	BIT RATE (Kbps)	TASA DE MUESTREO (Khz)	Ancho de Banda (Khz)
G711	Tiene dos versiones Ley μ (USA, Japón) y Ley A(Europa) para muestrear la señal.	64	8	4
G722	SBADPCM (*Sub-Band Adaptive Differential Pulse Code Modulation*).	48/56 64	16	7
G723.1	Multi-tasa Coder. Utilizado para transmisión de comunicaciones multimedia	5.3/6.3	8	21.9
G728	LD-CELP (*Low-Delay Code Excited Linear Prediction*)	16	8	3.4
G729	Se usa mayoritariamente en aplicaciones de Voz sobre IP VoIP por sus bajos requerimientos en ancho de banda	8/13	16	31.2

Tabla 1.1 Tipos de CODECS para VoIP

Paquetización de la voz: Encapsular la señal de voz al formato de un paquete IP para su transmisión**.**

Enrutamiento de los paquetes: Varios usuarios pueden utilizar simultáneamente la misma línea o realizar varias conversaciones al mismo tiempo.

Estas tres fases son cumplidas por el emisor en recepción una vez recibido el paquete IP se procede a desempaquetarlo, analizar la información y transformar la señal de digital a análoga.

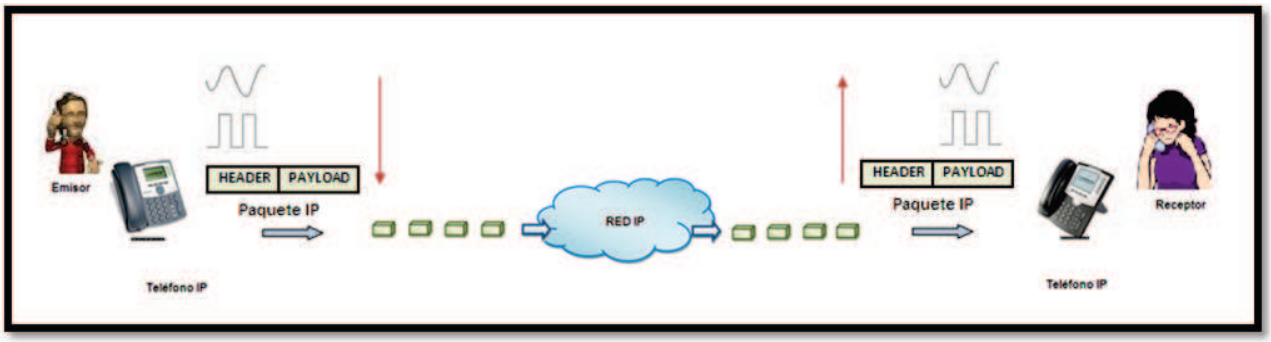


Figura 1.2 Fases de la telefonía IP

1.2.3 VENTAJAS DE LA TELEFONÍA IP

- Inter-operatividad entre redes telefónicas actuales con redes IP por medio de una central telefónica IP. La transmisión de datos y voz se realiza sobre la misma red, con la ventaja de disminución de costos por llamada. Una llamada entre dos teléfonos IP es gratuita, mientras que llamadas desde teléfonos IP a convencionales tienen un pequeño costo ligado al consumo de internet.

- Integración de servicios de valor agregado por los cuales la telefonía convencional cobra tarifas adicionales mensuales, como:

  a) Identificación de llamadas.

  b) Servicio de llamadas en espera

  c) Servicio de transferencia de llamadas

  d) Llamada de 3 líneas (*three-way calling*).

e) Desviar la llamada a un teléfono particular

f) Filtro de llamadas.

g) Enviar la llamada directamente al buzón de voz en el e-mail e integración con la agenda del gestor de correo electrónico.

- La telefonía IP brinda movilidad al usuario, éste puede acceder al servicio de telefonía desde cualquier punto donde exista conexión a Internet con el hardware y software necesario.

1.2.4 DESVENTAJAS DE LA TELEFONÍA IP

- La telefonía IP necesita una conexión de banda ancha con el fin de mantener una conversación fluida con VoIP. Las conexiones por modem limitan el rendimiento de este servicio.

- Tanto los teléfonos IP como la Central IP requieren conexión eléctrica para su funcionamiento. La suspensión del servicio eléctrico involucra una caída del sistema de telefonía IP. Esta limitación no está presente en la telefonía tradicional y es una desventaja frente a ella.

- La telefonía IP utiliza una dirección IP para identificar a un usuario en la red. En caso de emergencia una llamada al 911 no permite al operador ubicar geográficamente la zona del usuario que solicita ayuda.

- El teléfono IP que usa el usuario puede ser implementado por hardware o software. En el caso de instalar un *softphone*[3] en el computador, este se puede ver comprometido en su funcionamiento por el rendimiento del computador.Por esta razón los computadores que actúancomo*softphones* deben tener buenas características para procesar todas las tareas del usuario.

- La calidad de servicio que brinda la telefonía IP está ligada a la conexión que se establece en la red. Si esta conexión tiene problemas de latencia o pérdida de paquetes las conversaciones telefónicas pueden ser distorsionadas o en casos más graves no se podrá establecer la llamada.

1.2.5 ARQUITECTURA DEL SISTEMA DE TELEFONÍA IP

El estándar VoIP establece tres elementos básicos en su estructura:

- **Terminales:** Son los dispositivos finales empleados por el usuario. Estos pueden ser implementados a nivel de hardware o software (*softphone)*. Los dispositivos hardware tienen más opciones en relación a un teléfono convencional, e incluso puede ser configurado vía web o telnet.

- **Gatekeepers:** Se encarga de realizar dos funciones de control para el procesamiento de la llamada, con el fin de garantizar la integridad de la Red corporativa de datos. Primero da al sistema la traducción de direcciones y el control de acceso a la red de los terminales y *gateways*. Segundo gestiona el ancho de banda para determinar el número de llamadas simultáneas en la red cuyo fin es evitar sobresaturar al canal con peticiones que excedan al nivel establecido.

- **Gateways:** Constituyen el enlace entre la telefonía IP con la red telefónica tradicional. Su función consiste en emular la interfaz FXO/FXS(*Foreing Exchange Station/Office*) con el fin de adaptar una central de telefonía convencional *PBX* **(***Private Branch Exchange***)** o la *PSTN* con la red de telefonía IP pública.

  La centraltelefónica debe tener interfaces FXO para conectarse con la red de telefonía pública conmutada. Las interfaces FXS se utilizan para conectar extensiones análogas al sistema.

Softphone[3] Es un software que hace una simulación de teléfono convencional por computadora.

1.3 PROTOCOLOS

1.3.1 PROTOCOLOS DE TRANSPORTE

1.3.1.1 RTP (Real-Time Transport Protocol)

1.3.1.1.1 Definición:

Es un protocolo desarrollado por la *IETF*[4], el cual es empleado para transmitir información de audio y video a través del Internet en tiempo real. Este protocolo funciona sobre el protocolo de transporte *UDP*[5], es decir se encapsula dentro de datagramas *UDP*. Al trabajar bajo *UDP* no garantiza la entrega de todos los paquetes al igual que su llegada al destino en el instante adecuado. La capa de aplicación se encarga de superar estos fallos en la transmisión de información.

RTP no trabaja con un puerto predefinido por lo general escoge un número par elegido al azar.

1.3.1.1.2 Funciones:

- En telefonía IP *RTP* se encarga de la digitalización y compresión de la voz.
- Reconocimiento: Identifica el tipo de información transmitida.
- Secuenciación: *RTP* implementa números de secuencia y marcadores temporales a los paquetes IP para rearmar la información en el emisor y receptor, al igual que indicar el instante donde se generó el paquete.

IETF [4]*(Internet Engineering Task Force) E*s una organización internacional abierta de normalización, que tiene como objetivos el contribuir a la ingeniería de Internet**.**

UDP [5] (User Datagram Protocol) Es unprotocolo del nivel de transporte basado en el intercambio de datagramas. Permite el envío de datagramas a través de la red sin que se haya establecido previamente una conexión

- Multicasting: Los paquetes de difusión múltiple utilizan *RTP* para enrutar conversaciones a múltiples usuarios.

- Monitorización: Controla la llegada de los paquetes al destino.

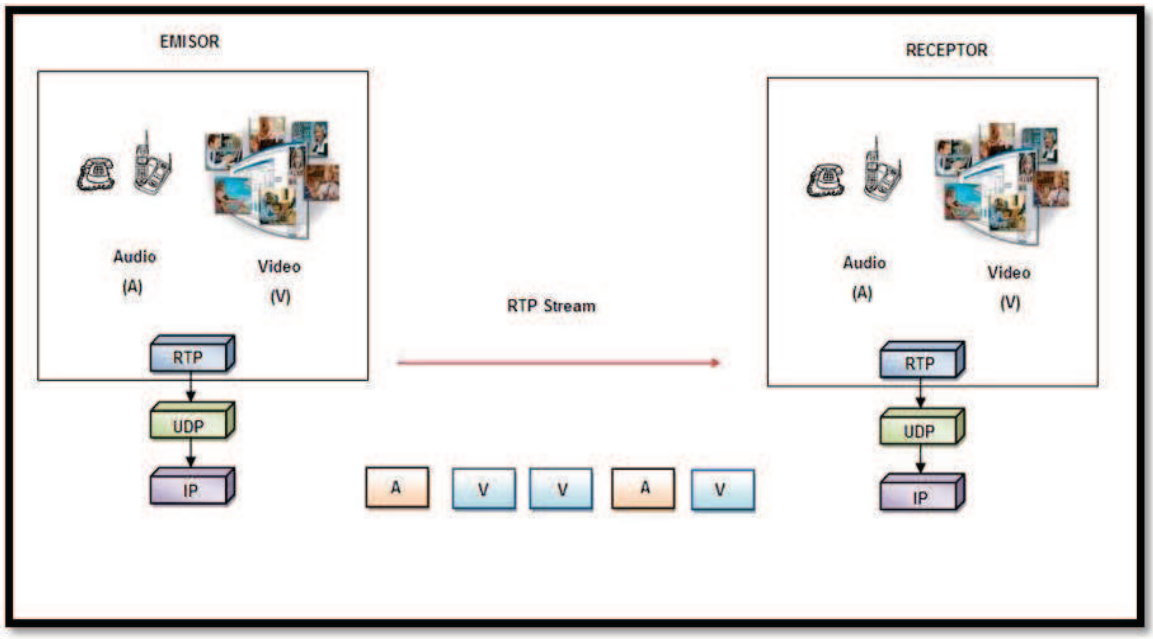


Figura 1.3 Funcionamiento del protocolo RTP[ref [1]]

1.3.1.2 RTCP (RTP Control Protocol)

1.3.1.2.1 Definición

Protocolo usado para enviar datos de control y de mediciones realizadas durante la transmisión. Para su transmisión se encapsulan dentro de mensajes *RTP*. La frecuencia de envío es aproximadamente cada cinco segundos. El puerto con el cual trabaja *RTCP* al igual que *RTP* no es definido, por ello escoge un número de puerto impar consecutivo en relación al seleccionado por RTP. RTCP no da ninguna clase de cifrado de flujo o de autenticación.

1.3.1.2.2 Funciones

RTCP realiza el control de flujo *RTP*, es decir permite obtener información básica sobre los participantes de la sesión y la calidad de servicio. El emisor y receptor de la comunicación intercambian estadísticas sobre paquetes perdidos y recibidos.

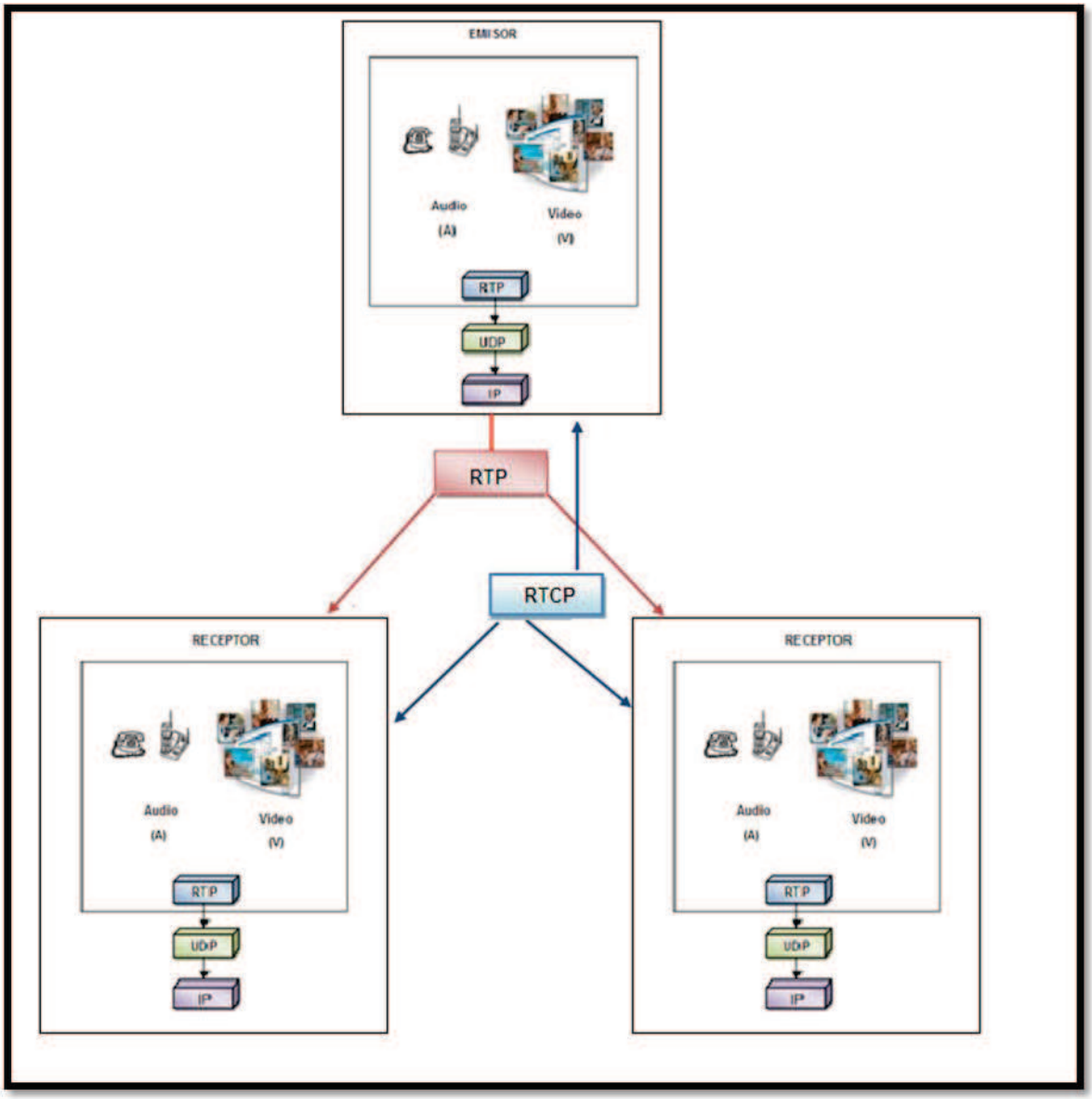


Figura 1.4 Funcionamiento del protocolo RTCP ref[2]

1.3.2 PROTOCOLOS DE SEÑALIZACIÓN

1.3.2.1 SIP (Protocolo de Inicio de Sesión)

1.3.2.1.1 Definición

Es un Protocolo de Inicio de Sesiones creado por el *IETF*. Este protocolo de señalización para voz sobre IP es utilizado para iniciar, modificar y terminar Sesiones Interactivas de Comunicación Multimedia entre usuarios tales como: video, voz, mensajería instantánea y realidad virtual.

La sintaxis de las operaciones que realiza *SIP*son equivalentes a las realizadas por los protocolos de servicio de páginas web (HTTP) y distribución de e-mails (SMTP).

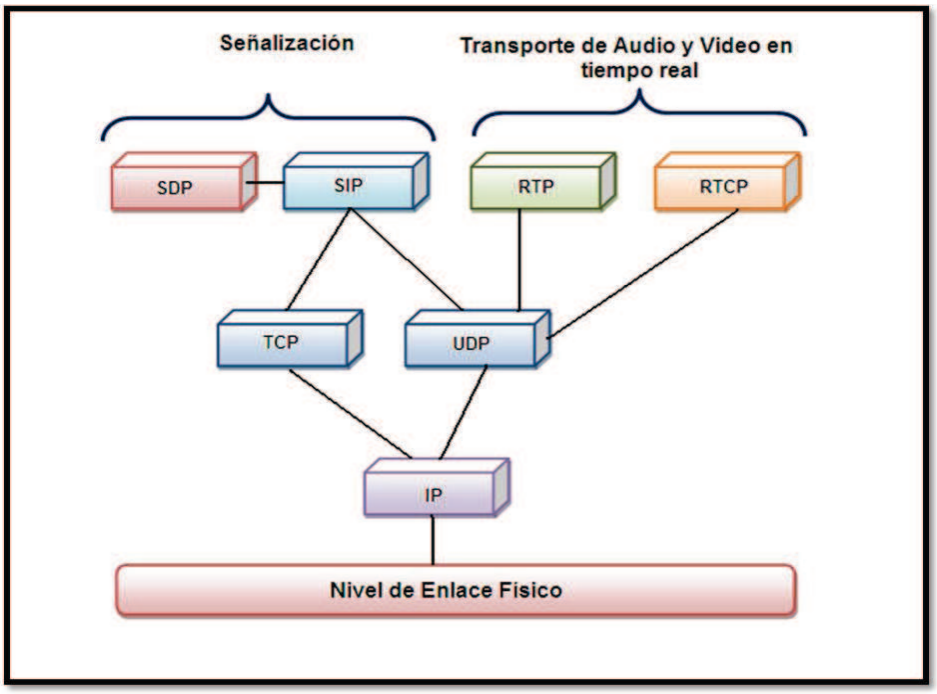


Figura 1.5 Pila de Protocolos de SIP[ref[3]]

SIP para su funcionamiento trabaja en conjunto con los protocolos *SDP* y *RTP*. *SDP* (*Session Description Protocol*) se encarga de describir los parámetros de inicialización de los flujos multimedia en una sesión, es decir su función es invitar, anunciar y negociar las capacidades de una sesión multimedia en internet. *RTP* (*Real-Time Transport Protocol*) cumple con la función de transportar en tiempo real el contenido de voz y video entre los usuarios una vez establecida la sesión.

1.3.2.1.2 Componentes

a) Agentes de Usuario

Los Agentes de usuario son los puntos extremos de la Sesión, estos se pueden comportar de dos modos:

- Clientes: UAC (*User Agent Client*)

- Servidores: UAS (*User Agent Servers*)

Un *User Agent Client* es el encargado de realizar las peticiones de servicio, mientras que el *User Agent Server* es el encargado de procesar la petición recibida.

b) Servidores Proxy o de Redirección

Los servidores tienen como función encaminar un mensaje entre el *User Agent Client* y el *User Agent Server*. Un servidor de acuerdo a la situación puede funcionar de dos modos:

Servidor Proxy.- Este servidor realiza una petición a nombre de un Agente de Usuario, dirigida a otro Proxy u otro Agente de Usuario. Su función principal es encaminar las invitaciones de sesión hacia el Agente de Usuario llamado. Para llegar al destino una invitación de sesión puede atravesar en el camino varios proxies hasta llegar al proxy que le indique la localización exacta del Agente de Usuario llamado.

Los Servidores Proxy pueden ser:

- ***Stateful Proxy*.-** Crea el estado de petición y lo mantiene hasta que termine la transacción.
- ***Stateless Proxy*.-** Únicamente reenvía los mensajes SIP.

Servidor de Redirección.- Este servidor escucha peticiones y retorna respuestas con la localización actualizada de un usuario. La búsqueda del usuario la realiza en la Base de Datos creada por el Servidor de Registro.

Ref [1]: http://memnon.ii.uam.es/~eloy/media/REDES/Tema7.4-rtp.pdf

Ref [2]: http://www.uco.es/~i62gicaj/RTP.pdf

c) Servidores de Registro

Los Servidores de registro se emplean para establecer la ubicación física de un usuario. Para identificar una entidad SIP se emplea la forma denominada URI (*Uniform Resource Identifier*). Esta forma es similar a una cuenta de correo electrónico. Cada usuario tiene por consiguiente una dirección lógica (URI) usuario@dominio independiente de su ubicación física y una dirección física llamada dirección de contacto la cual es su dirección IP. Un Servidor de registro realiza un proceso denominado *Bindind*, para asociar la dirección física con la dirección lógica mediante el método *REGISTER* cuando se inicializa un terminal de usuario. Esta asociación tiene un tiempo de vida que debe ser renovado para no caducar. Para finalizar la asociación también se puede emplear el método *DESREGISTER.*

1.3.2.1.3 Funciones

a) Resolución de Direcciones.

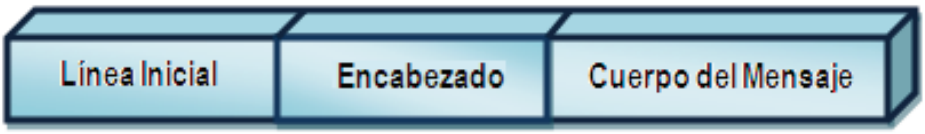
b) Funciones de sesión :

- Establecimiento
- Negociación de medios
- Modificación
- Terminación
- Cancelación
- Señalización en llamada
- Control de llamada
- Configuración de QoS

c) No relacionadas con la sesión

- Movilidad
- Transporte de Mensajes
- Suscripción a eventos
- Autenticación

1.3.2.1.4 Mensaje SIP



Línea Inicial	Request-Line	Versión del Protocolo, método, direcciones involucradas.
Encabezado	Header Message	Origen de la petición, destino de la petición, Identificador de llamada.
Cuerpo del Mensaje	Message Body	Carga útil o payload

Figura 1.6 Estructura del mensaje SIP[ref [3]]

Ref [3]:http://www.cudi.edu.mx/primavera_2005/presentaciones/rodolfo_castaneda.pdf

1.3.2.1.5 Tipos de Mensajes

MÉTODOS	FUNCIÓN
INVITE	Inicio de Sesión
ACK	Reconocimiento de *INVITE*
BYE	Terminación de sesión
CANCEL	Cancelación de *INVITE*
REGISTER	Registro de URL[6]
OPTIONS	Preguntar por opciones y capacidades
INFO	Transporte de información en llamada
PR*ACK*	Reconocimiento Provisional
COMET	Notificación de precondición
REFER	Transferencia a otra URL
SUSCRIBE	Requerir notificación de Evento
UNSUSCRIBE	Cancelar notificación de Evento
NOTIFY	Notificación de Evento
MESSAGE	Mensaje Instantáneo

Tabla 1.2 Mensajes SIP ref [3]

1.3.2.1.6 Códigos de respuesta

CLASE	DESCRIPCIÓN
1XX	Información provisional, requerimiento en progreso pero no terminado.
2XX	Respuesta final completada satisfactoriamente.
3XX	Redirección: Petición debería re-direccionarse
4XX	Error en cliente (error en la petición)
5XX	Error de servidor
6XX	Falla Global

Tabla 1.3 Códigos de respuesta SIP ref [3]

URL [6] (*Uniform Resource Locator*) Localizador de Recurso Uniforme la dirección global de documentos y de otros recursos en la *World Wide Web*.

1.3.2.1.7 Proceso de Registro

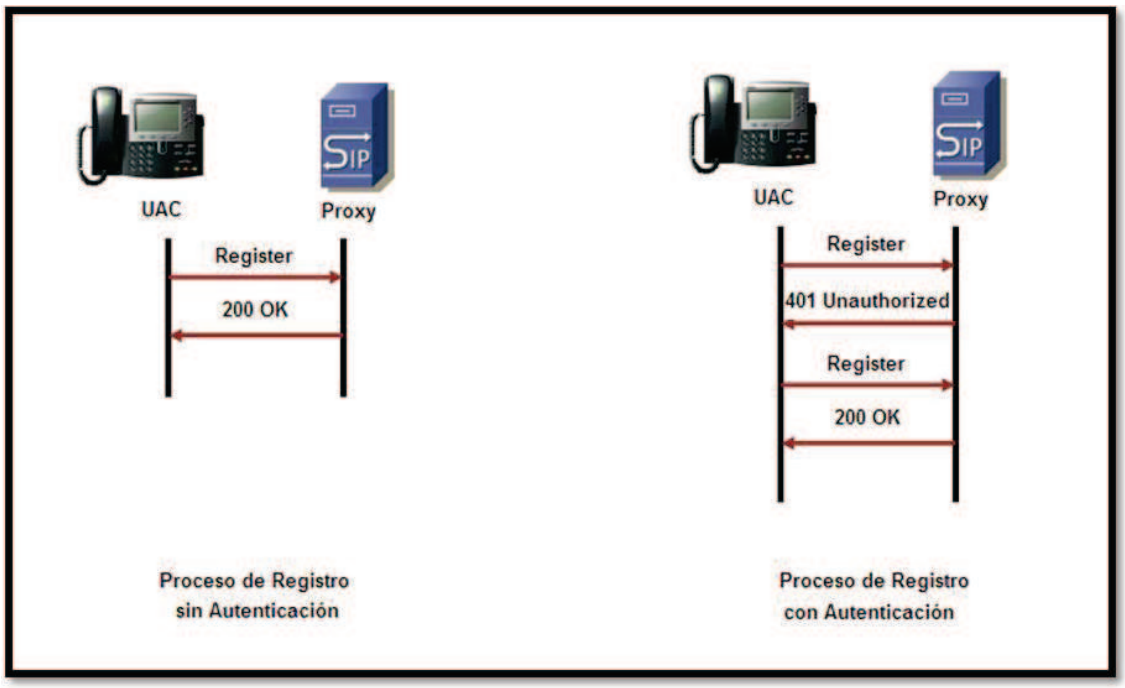


Figura 1.7 Registro SIP[ref [3]]

1.3.2.1.8 Establecimiento de Sesión

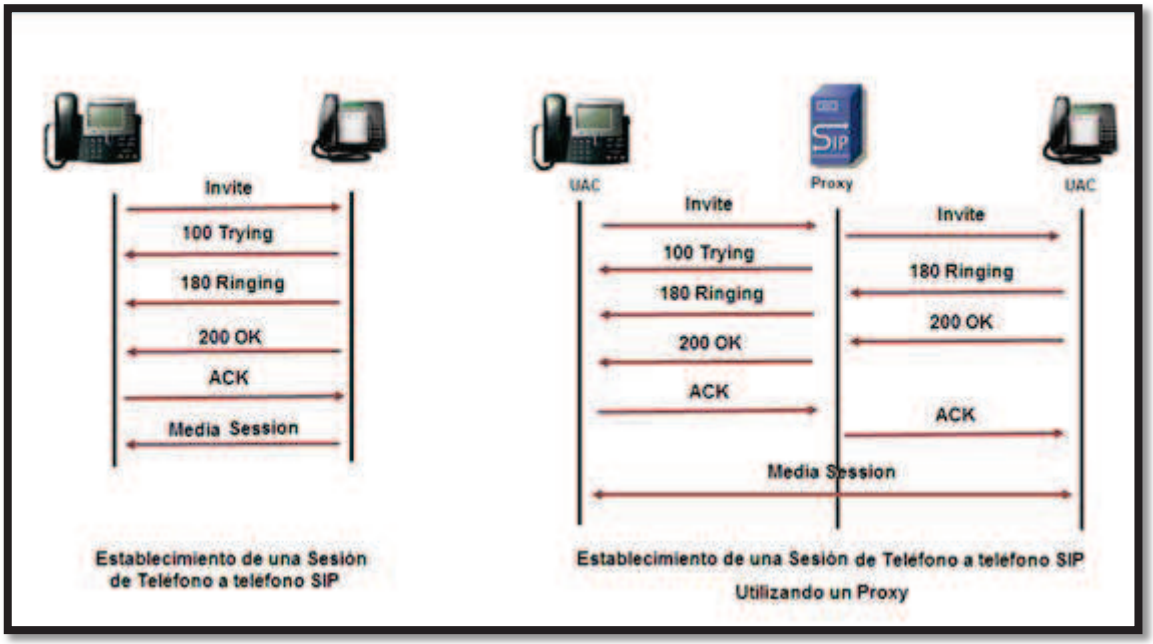


Figura 1.8 Establecimiento de una Sesión SIP[ref [3]]

1.3.2.1.9 Terminación y cancelación de Sesión

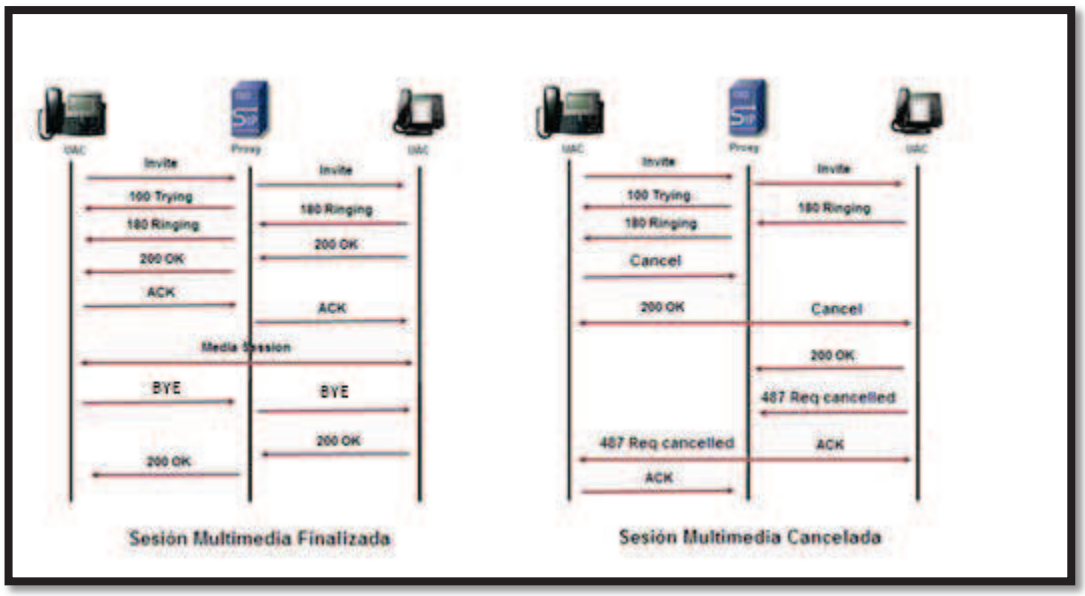


Figura 1.9 Finalización de una Sesión SIP [ref [3]]

1.3.2.2 IAX(Inter-Asterisk eXchange Protocol)

1.3.2.2.1 Definición

Es un protocolo creado para la señalización de VoIP en Asterisk. Este protocolo es abierto para su libre desarrollo. Para control y tráfico de datos usa el puerto *UDP* 4569. *IAX* se basa en muchos estándares de transmisión de datos como *SIP* y *RTP*.

1.3.2.2.2 Funciones

- *IAX* proporciona control y transmisión de flujos de datos multimedia sobre redes IP.
- El objetivo de este protocolo es optimizar el consumo de ancho de banda, por lo cual permite mayor número de canales entre terminales y empaqueta múltiples sesiones dentro de un flujo de datos.
- Maneja conexiones de VoIP entre servidores Asterisk. La configuración de este protocolo se especifica en el archivo *iax.conf*.
- Permite la autenticación pero no hay cifrado.

1.3.3 PROTOCOLOS DE ENRUTAMIENTO DINÁMICO

1.3.3.1 OSPF (Open Shortest Path First)

1.3.3.1.1 Definición

Es un protocolo de enrutamiento jerárquico de pasarela interior, o *IGP* (Interior Gateway Protocol)pararedes TCP/IP, basadas en el RFC 2328.*OSPF* es un estándar abierto, razón por la cual está disponible en múltiples sistemas operativos como: Windows 2003 Server, Linux, Cisco IOS, etc.

1.3.3.1.2 Funcionamiento

OSPF funciona dividiendo una Intranet o un sistema autónomo en unidades jerárquicas de menor tamaño. Cada una de estas áreas se enlaza a un área *backbone* mediante un router fronterizo. Todos los paquetes enviados desde una dirección de una estación de trabajo de un área a otra de un área diferente atraviesan el área *backbone*, independientemente de la existencia de una conexión directa entre las dos áreas. Aunque es posible el funcionamiento de una red OSPF únicamente con el área *backbone*, OSPF escala bien cuando la red se subdivide en un número de áreas más pequeñas.[ref [4]]

OSPF opera como protocolo de estado de enlace, e implementa un algoritmo para calcular la ruta más corta a cada red de destino, basándose en el ancho de banda. *OSPF* es un protocolo apto para su implementación en redes de todo tipo y tamaño.[ref [5]]

El fundamento principal en el cual se basa un protocolo de estado de enlace, es en la existencia de un mapa de la red,al cual acceden todos los nodos y que regularmente es actualizado.La creación del mapa de red local en cada router de la red se realiza a través de una tabla de enrutamiento.

1.3.3.1.3 Características

- Mayor velocidad de convergencia al surgir algún cambio en el estado de los enlaces de la red.

- Actualizaciones pequeñasya que no envía toda la tabla de enrutamiento.

- Sin bucles de enrutamiento.
- Escalabilidad en redes grandes.
- Métricas basadas en anchos de banda.

1.3.3.2 RIP (Routing Information Protocol)

1.3.3.2.1 Definición

El protocolo *RIP* es un protocolo de encaminamiento dinámico de tipo *IGP* (*Internal Gateway Protocol*), utilizado por routers y equipos, para intercambiar información acerca de redes IP.

1.3.3.2.2 Funcionamiento

RIP calcula el camino más corto hacia la red de destino usando el algoritmo del vector de distancia. La distancia o métrica está determinada por el número de saltos del router hasta alcanzar la red de destino.

RIP no es capaz de detectar bucles, por lo que necesita limitar el tamaño de la red a 15 saltos. Cuando la métrica de un destino alcanza el valor de 16, se considera como inalcanzable.

La métrica de un destino se calcula como el número de saltos comunicado por un vecino más la distancia en alcanzar a ése vecino. La tabla de rutas hacia ese destino se actualiza sólo en el caso de que la métrica anunciada más el coste en alcanzar sea estrictamente menor a la de la ruta almacenada. Sólo se actualizará a una métrica mayor si proviene del enrutador que anunció esa ruta.[ref [6]]

Ref[4] http://aprenderedes.com/2006/10/configuracion-de-ospf/

Ref [5]http://www.giga-optics.com/portal/index.php/ospf

Ref [6]http://es.wikipedia.org/wiki/Routing_Information_Protocol

1.3.3.2.3 Características

- *RIP* es más fácil de configurar en comparación con otros protocolos de enrutamiento.
- Es un protocolo abierto, soportado por muchos fabricantes.
- Para determinar la mejor métrica, únicamente toma en cuenta el número de saltos, descartando otros aspectos fundamentales tales como: Ancho de banda y congestión de rutas.

1.4 SOFTWARE

1.4.1 ASTERISK

1.4.1.1 DEFINICIÓN

Es software *open source*, es decir de código abierto, utilizado para la implementación de una Central Telefónica Privada. Este software es desarrollado principalmente por la empresa americana *DIGIUM*. La empresa *DIGIUM*, fundada por Mark Spencer, administra y mantiene el código fuente de Asterisk, y lo ofrece bajo dos licencias: *GPL* y licencia comercial. *GPL* da la libertad de ejecutar el programa, con cualquier propósito y de redistribuir copias mientras que la licencia comercial tiene un coste económico en función de las características requeridas como número de canales simultáneos, aplicaciones de software, etc. Asterisk es en sus inicios se desarrolló para cualquiera de las distribuciones de Linux y en la actualidad se puede acceder a versiones para otros sistemas operativos como: *MAC*, *Solaris, Windows y BSD*[7].

1.4.1.2 CONCEPTOS RELACIONADOS

- **Dial-plan:** Constituye la configuración de la central Asterisk, es decir su comportamiento lógico.El dialplan o plan de marcado, por tanto indica el itinerario que sigue una llamada desde que entrao sale del sistema hasta que llega a su destino.
- **Extension:** En Asterisk una extensión es una lista de comandos a ejecutar. Las extensiones se acceden cuando:

– Se recibe una llamada entrante por un canal dado.

– El usuario que ha llamado marca la extensión.

– Se ejecuta un salto de extensiones desde el Dialplan deAsterisk.

- **Contexto (*Context*):** El plan de marcado se divide en uno o varios contextos. Un contexto es una colección de extensiones. Los contextos existen para programar las funcionalidades de marcado deacuerdo a los requerimientos del usuario, tales como: diferenciar donde se encuentra una llamada, aplicar políticas de seguridad y acceso, procesos de registro de los teléfonos IP, etc.

- **Aplicación (*Application*):** Asterisk ejecuta secuencialmente los comandos asociados a cada extensión. Esos comandos son realmente aplicaciones que controlan el comportamiento de la llamada y del sistema en sí.

1.4.1.3 FUNCIONAMIENTO

Asterisk es un demonio que se ejecuta en segundo plano. Al igual que el resto de servidores tales como: *Apache, openssh, proftpd, bind*, etc.

La configuración normalmente se almacena en varios ficheros de texto los cuales son editables para su configuración.Estos archivos son:

1.4.1.3.1 *extensions.conf:*

Es el propio Dial-plan de Asterisk.Le permite a la central telefónica determinar las acciones a tomar en función, básicamente, de lo que ha marcado el usuario desde el cliente de telefonía IP. Se le denomina el “corazón” de Asterisk, pues gracias a este archivo es que esta solución de telefonía IP goza de tanta flexibilidad y funcionalidad, sin olvidar el gran aporte que le da la existencia de aplicaciones y funciones que pueden utilizarse dentro del plan de marcado.

BSD[7] (*Berkeley Software Distribution*) Sistema operativo derivado del sistema Unix nacido a partir de los aportes realizados a ese sistema por la Universidad de California en Berkeley.

Estructura.- La estructura para definir las extensiones es la siguiente:

```
exten => EXTENSION, PRIORIDAD, Aplicación
```

- Las extensiones son los dígitos, el destino de llamada que ha marcado el usuario cuando llama.Algunas extensiones especiales:
  - ***s:*** Extensión por defecto cuando una llamada entra en uncontexto sin número destino asociado.
  - ***i:*** Cuando el usuario marca una extensión incorrecta.
  - ***t:*** Cuando se produce un *timeout* (Tiempo fuera).
- La prioridad puede ser asignada con números por ejemplo 1,2,3 etc. Sin embargo lo más recomendable es utilizar el parámetro “n”. La n es equivalente a sumar uno a la prioridad anterior. Esto permite editar y modificar rápidamente eldialplan.
- Aplicaciones:
  - ***Wait (n):*** Espera n segundos, ignorando los dígitos marcados durante.
  - ***WaitExten (n):*** Espera n segundos, pero gestionando los dígitos marcados.
  - ***Answer():*** Acepta la llamada entrante por el canal.
  - ***Busy():*** Envía la señal de ocupado al origen.
  - ***Hangup():***Cuelga la llamada.
  - ***Ringing():*** Envía la señal de tono de llamada.
  - ***Dial:***(tipo/identificador,timeout,opciones,url)
  - ***Goto*** (contexto, extensión, prioridad): Salta al contexto, extensión y prioridad del argumento.
  - ***GotoIf*** (condición ? prioridad1 : prioridad2)

- Salta a la prioridad1 si la condición se cumple.
- Salta a la prioridad2 si la condición no se cumple.

Aspectos de Programación

Para simplificar la programación del archivo se utilizan patrones de coincidencia. Para indicar patrones, se utiliza el carácter: "_". Se pueden utilizar:

– ***X:*** Indica un dígito del 0 al 9

– ***Z:*** Indica un dígito del 1 al 9

– ***N:*** Indica un dígito del 2 al 9

– ***[129]*** Indica el 1, 2 o 9

– **.** Indica uno o más caracteres.

En el dialplan de Asterisk existen variables, que pueden sermodificadas por el propio Asterisk en su ejecución lógica opor comandos expresos del dialplan.

Los tipos de variables son:

– **Globales:** Declaradas en extensions.conf (o por comando).

– **Canal:** Son propias a cada canal.

– **Entorno:** Variables de entorno (UNIX Like).

La sintaxis de una variable es:

```
${variable}
```

Listado de variables más importantes:

- **${CALLERID}:** Caller ID actual, nombre y número.
- **${CONTEXT}:** Contexto actual.
- **${EXTEN}:** Extensión actual.

- **${CHANNEL}:** Canal actual.

- **${DIALSTATUS}:** Estado de la llamada: unavailable,congestion, busy, noanswer, answer, cancel, hangup.

- **${DATETIME}:** Hora actual.

Las expresiones se utilizan por lo general al programar las aplicaciones.

Syntaxis:

```
s[expr1 operador expr2]
```

- Operadores Lógicos: `|(or) , &(AND)`
- Operadores de Comparación: `=, !=, <, >, <=, >=`
- Operadores Aritméticos: `+, -, *, /, %`

1.4.1.3.2 *sip.conf:*

En este fichero se definen las variables globales de SIP.

Sección General: Contiene la configuración por defecto de todos los usuarios que se conectarán a la central, contiene la definición de las variables globales y aspectos por defecto para todos los canales SIP.

La sintaxis es la siguiente:

```
[general]

variable1=valor1

variable2=valor2
```

Las variables generales más importantes son:

–***allow y disallow:*** indican los CODECS permitidos / no permitidos.

–***context***: Contexto por defecto donde entraran las llamadas entrantes por SIP.

–***port:*** Puerto en el que se escucha (5060).

–***Insecure***: autentificará al enviar al usuario llamadas.

Clientes y Servidores:

– ***user:***Envía llamadas a Asterisk

– ***peer:*** Recibe llamadas de Asterisk (proveedor).

– ***friend:*** Recibe y Envía llamadas (usuario).

La syntaxis para definir un friend o un peer es:

```
[nombre]

type = friend / peer

variable = valor

viarable2 = valor
```

Las variables más importantes que deben ser configuradasinicialmente son:

– ***type:***peer / friend

– ***context:*** Contexto donde entraran las llamadas generadas.

– ***nat:*** Indica si el usuario o peer se encuentran tras un nat.

– ***host:*** IP remota o dynamic.

– ***username:*** nombre de usuario.

– ***secret***: contraseña de acceso.

– ***allow y disallow:*** Configuraciones de CODECS específicas paracada friend/peer.

1.4.1.3.3 *iax.conf*:

En este archivo se definen todos los enlaces que se realizaránusando el protocolo *IAX*. Al igual que en SIP, se pueden definir:

- Variables generales de *IAX*
- Clientes *IAX*
- Servidores *IAX*

IAX soporta nativamente autenticación con clave pública/privada, al definir un peer se puede autenticar de la siguiente forma:

```
auth=rsa

inkeys=nombredelaclave
```

IAX soporta nativamente "trunking", es decir, enviar por unmismo flujo varias llamadas. Para activarlo:

```
trunk=yes
```

1.4.1.4 VENTAJAS

- Reducción de costos: Asterisk es una aplicación de código abierto por ello elimina los costos de licencia para su uso, constituyéndose en una solución económica. Sin embargo su gran ventaja radica en integrar y aprovechar la infraestructura de red presente en las empresas, permitiendo implementar sobre ella telefonía IP.

- Compatibilidad: Esta aplicación brinda una amplia compatibilidad con protocolos de VoIP, facilitando su implementación en la red y el uso de varios CODECS.

- Asterisk no solo limita al sistema al uso de teléfonos IP. Teléfonos tradicionales es decir analógicos pueden formar parte de la red y operar sin ningún inconveniente al utilizar tarjetas electrónicas FXS o FXO aparte del *modem*.

1.4.2 *SIPP*

1.4.2.1 DEFINICIÓN

SIPP es un software de código abierto. Esta herramienta de prueba gratuita permite simular y generar tráfico del protocolo SIP. Con *SIPP* se pude simular cada comportamiento de un teléfono SIP incluyendo los escenarios de agente de usuario servidor y cliente. Para llevar a cabo la simulación el funcionamiento de *SIPP* se basa en la programación de scripts *XML*. Los scripts *XML* contienen la secuencia de mensajes de envío y recepción.

1.4.2.2 VENTAJAS

- *SIPP* presenta estadísticas sobre la ejecución de las pruebas de tráfico SIP, obteniendo información del coste de una llamada, retraso de ida y vuelta y mensajes estadísticos. ref [7]

- *SIPP* da soporte para IPV6, autenticación SIP, *TLS*[8], retransmisiones *UDP* y robustez de error (tiempo de espera de llamada) ref[7]

1.4.3 *DUNDi*

1.4.3.1 DEFINICION

DUNDi (*Distributed Universal Number Discovery*) es un *draft*[9]de Internet creado por Mark spencer y tiene como función el descubrimiento de un número entre diferentes *PBXs* en una red punto a punto. Para ello busca y comparte los planes de marcación (dialplan) entre servidores IP-*PBX*. *DUNDi* por consiguiente es un protocolo de enrutamiento de VoIP que brinda servicios de directorio similar a los proporcionados por *ENUM* pero sin la restricción de ser un sistema centralizado.

Ref[7] http://sipp.sourceforge.net/

Cada servidor *PBX* debe tener su plan de marcación y para la configuración del protocolo se emplean los archivos iax.conf, extensions.confy dundi.conf.

1.4.3.2 VENTAJAS

- *DUNDi* permite configurar una red redundante al eliminar puntos centrales de falla. En el caso que un Servidor *PBX* deje de funcionar la llamada será enrutada por otras rutas para llegar a su destino sin ser descartada.

- *DUNDi* da la posibilidad de añadir nuevas extensiones a los planes de marcación en forma dinámica, función requerida para dar movilidad a los usuarios a través de la red.

- *DUNDi* se puede utilizar en Internet para formar una web E.164 de confianza que permita a los proveedores de servicios hacer disponibles los números de teléfonos reales en Internet. Esto permite el *baypass* de cierto número sin recargo adicional de la publicación de la información. ref[8]

- *DUNDi* no consume mayor ancho de banda para proporcionar seguridad y privacidad. *DUNDi* utiliza RSA y AES para el cifrado y la autenticación de la información.

1.4.3.3 FUNCIONAMIENTO

Cada cliente *DUNDi* tiene que conocer al menos a otro cliente *DUNDi* o par dentro de la red.

En la configuración de *DUNDi* el número de búsquedas consecutivas entre pares en la red para llegar al destino deseado se puede limitar utilizando el TTL (*Time To Live*).

- El TTL=1 significa que sólo se puede pedir búsquedas a los compañeros conocidos.

- El TTL=n significa que al compañero al cual se pide la búsqueda puede redirigir la búsqueda a otros compañeros conocidos por él, solo con un TTL de n-1. ref [9]

Cada intento se debe aplicar para reducir al mínimo el número de consultas en todo el sistema de confianza, sin sacrificar la exactitud de las respuestas aportadas. Algunas consultas se puede reducir con el almacenamiento en caché.

Cada nodo en *DUNDi* se identifica por una entidad a nivel mundial único de seis bytes identificador, cuyo valor normalmente es una dirección MAC de Ethernet para una de las interfaces en el sistema. ref [10]

- **Transacción:**

  Una transacción *DUNDi* es un diálogo completo en el protocolo *DUNDi* de una *PBX* (nodo) a otra. Cada mensaje contiene dos números de transacción un origen y un destino para que un mensaje recibido puede ser identificado únicamente por el número de transacción de manera aislada.

- **Retransmisiones**

  Una solicitud *DUNDi* puede ser retransmitidos hasta 10 veces a intervalos de tiempo que serán decididas por el implementador, pero no más de 1 segundo. Siun mensaje no se reconoce después de 10 retransmisiones o 10 segundos, la operación deberá ser considerada cerrada sin mensajes adicionales que se transmiten o se esperan. ref [10]

- **Registro**

  El registro se utiliza para un par *DUNDi*, por lo general con una ubicación dinámica o una dirección traducida, para registrar su presencia a otros pares *DUNDi*. ref [10]

TLS [8] *(Transport Layer Security*) Seguridad de la Capa de Transporte

Draft [9] Borradores de Internet, es una serie de documentos de trabajo publicados por la IETF.

Ref [8] http://www.dundi.com/

Ref [9] http://en.wikipedia.org/wiki/Distributed_Universal_Number_Discovery

Ref [10] http://www.dundi.com/dundi.txt

1.4.4 ENUM

1.4.4.1 DEFINICIÓN

ENUM (RFC 3761-*Electronic Number Mapping System*) es un protocolo que utiliza el sistema *DNS* de Internet para traducir números de teléfono convencionales E.164 a esquemas de dirección de IP (como las de SIP, H323 o email).[ref[11]]

E.164 es el nombre de la normativa que define el plan de numeración telefónica internacional que administra la Unión Internacional de Telecomunicaciones (UIT). La UIT concede códigos de país a las naciones con soberanía y la administración de los números de teléfono de cada país los efectúa el regulador del país correspondiente. Un número E.164 está compuesto por el código de país, código de zona o ciudad y un número telefónico. Sin embargo, en algunos países, la marcación dentro de una zona o de una ciudad puede ser abreviada, sin necesidad de tener que marcar el código de zona o ciudad. [ref [12]]

ENUM entonces traducirá números de teléfono tales como +12025332600 a *URIs* como por ejemplouser@host.com y viceversa, facilitando así la interconexión de estos sistemas, es decir las comunicaciones terminan en el identificador correcto sea este un número E.164 o un URI.

1.4.4.2 FUNCIONAMIENTO

Para utilizar el servicio se requiere de tres elementos:

- *Uniform Resource Identifier* (*URI*) personal, que se utilizará en la zona IP de las redes.
- Un número E.164 ENUM, es decir un número asignado al usuario de telefonía pública o su teléfono móvil.
- Un permiso o autorización para poder escribir las preferencias de desvío y terminación de llamadas en el registro *NAPTR* (Autoridad de Nombres de Puntero).El registro *NAPTR* es un registro de recursos usado en *DNS* para guardar información acerca del usuario.

El conjunto funciona del modo siguiente:

1. Se proporciona al usuario un nombre de dominio es decir el *URI* que se utilizará para acceder al servidor *DNS* y al registro *NAPTR*.
2. El usuario registra su número telefónico personal (el número *ENUM*), el nombre de dominio se asocia biunívocamente al número *ENUM*.
3. Finalmente el registro *NAPTR* que se obtiene mediante dicho *URI* contiene las preferencias de desvío y terminación de llamadas del usuario. Entre estas preferencias el usuario decide cómo va a ser contactado por ejemplo si se le comunica con un email, *sms*, telefax o llamada de voz.

Se parte del número telefónico ENUM.	+59322073613
Se retira el símbolo + y se añade puntos entre los dígitos.	5.9.3.0.2.2.0.7.3.6.1.3
Se invierte la secuencia para seguir con la lógica de los DNS.El motivo de este proceso de cambio se debe principalmente a la distribución actual de las URL, dónde el nivel de jerarquía disminuye de derecha a izquierda, sumergiéndose cada vez más dentro de un dominio.	3.1.6.3.7.0.2.2.0.3.9.5
Se agrega el dominio establecido para ENUM es decir el dominio e164.arpa.	3.1.6.3.7.0.2.2.0.3.9.5.e164.arpa 3.1.6.3.7.0.2. = Número de usuario 2.0.= Código nacional 3.9.5= Código de país E164.arpa= Dominio

Tabla 1.4 Funcionamiento de ENUM

4. Realizado este registro, un usuario que desee contactarse con otros usuarios *ENUM*marcará dentro de una aplicación habilitada para hacer

5. búsquedas *ENUM*. El servidor *DNS* le contestará las diferentes formas de contacto asociadas al número solicitado.

Este es el servicio que nos brinda *ENUM*poder acceder a las direcciones de correo electrónico, *URL*, teléfonos IP y a los números de fax de un usuario con tan sólo al marcar su número telefónico, y, viceversa, recibir un email, un documento fax o una visita a una dirección *URL*, además de llamadas de voz, de alguien que conozca el número telefónico del usuario.

La ITU ha asignado una zona específica o rango *URI* para *ENUM*, éste es el E164.arpa. Para que *ENUM* se implemente en un país se requiere que la ITU delegue el dominio correspondiente al país de acuerdo al formato x.y.z.e164.arpa donde zyx es el código de país correspondiente.El organismo que se encarga de examinar las solicitudes de delegación para el dominio e164.arpa es el *TSB* (Oficina de Normalización de las Telecomunicaciones). Los países a los cuales ya se les ha delegado su dominio, administrarán de acuerdo a sus propias normas los númerostelefónicos que sean para uso de *ENUM*. Cabe señalar que los dominios de los códigos de país son almacenados en el *DNS* e164.arpa quedando estos bajo la completa administración del *DNS*.

Para ampliar las aplicaciones de *ENUM* a un servicio distribuido en el cual se posea un directorio de todos los teléfonos, faxes, emails, módems y demás posibilidades de comunicación sería necesario desarrollar una troncal *ENUM* que soporte un sin número de peticiones sin tener problemas de seguridad y rendimiento. En este caso los elementos clave de todo el sistema *ENUM*, serían los servidores raíz los cuales deberían ser implementados a nivel mundialhaciendo posible así que todos los países compartan los gastos y la responsabilidad de mantenerlos.Estos servidores darán lugar a lo que se conoce como capa o Tier 0. Éste es el primer nivel de *ENUM*, en el que se guarda registro de todos los prefijos nacionales de los países asociados al sistema. [ref[13]]

Ref [11] http://www.rfc-archive.org/getrfc.php?rfc=3953

Ref [12] http://es.wikitel.info/wiki/E.164

Una vez la petición ha pasado por el Tier 0, éste la redirige al Tier 1, que corresponde a la entidad responsable en cada país de manejar este tipo de infraestructuras.

El Tier 1 mantiene un registro completo de todos los números correspondientes al país que opera bajo ese código nacional. Aunque es este nivel el que se encarga en última instancia de redirigir las llamadas a uno u otro país, existe también la figura del Tier 2, correspondiente a los distintos operados de telecomunicaciones. ref[13]

1.4.4.3 APLICACIONES

- Voz sobre IP
- Mensajería unificada
- Mensajería Instantánea
- IP Fax
- Páginas personales WEB.

1.4.4.4 TIPOS DE ENUM

1.4.4.4.1 ENUM PÚBLICO O DE USUARIO

Los elementos que intervienen son:

- Abonado o registrante
- Registrador
- Registro
- Regulador.

Cuando un usuario de la RTPC marca un número *ENUM*, el *gateway* RTCP/IP cambiará ese número E.164 por el *URI* correspondiente. A través del URI se accederá al servidor *DNS* de *ENUM* y traerá un registro *NAPTR* que se conoce

como “la información de acceso”, el cual tendrá la prioridad de los servicios de la persona llamada y así completar la comunicación de acuerdo a esa prioridad. Los servicios pueden ser dirección de e-mail, número de fax, número telefónico, página web, *URI* de VoIP, coordenadas *GPS* o direcciones de mensajería instantánea. De cualquier manera, si se inicia una llamada desde la RTPC o por una red IP se accede al mismo registro *NAPTR*.[ref [13]]

1.4.4.4.2 ENUM PRIVADO

Las empresas pueden utilizar las técnicas de *ENUM* en sus redes y en las de sus clientes o proveedores para tener planes de numeración privados, de manera que puedan permanecer perfectamente comunicados para todo tipo de información, sea voz, datos, e-mail e imágenes.

El servicio de *ENUM* privado no está obligado a sujetarse a las normas o estándares públicos en su totalidad porque está en un entorno privado y controlado, por lo que puede no emplear el sistema *DNS* público.

Para que la implementación de *ENUM* sea apropiada, los registrantes deben tener la seguridad que el funcionamiento del sistema no afectará la privacidad de su información personal. La participación y elección de hacer uso del servicio de *ENUM* es decisión del usuario. De esta manera no se registra ningún número telefónico sin el consentimiento del propietario autorizado de ese número.

Los registrantes controlan la prioridad y la información que se agrega a los registros *NAPTR* asociada con su número telefónico. Para poder utilizar los servicios de comunicaciones el usuario debe estar dispuesto a revelar cierta información personal, como es correo electrónico y número telefónico.[ref [13]]

1.4.4.4.3 DE INFRAESTRUCTURA O DE OPERADOR

En esta clasificación de *ENUM*, el Operador es el que administra los registros *NAPTR*. La información contenida en la base de datos pública o privada, es manejada por el Operador, de manera que el *ENUM* de Infraestructura les permite a los Operadores encaminar sus llamadas. Se utiliza básicamente para la interconexión entre operadores de telefonía, a diferencia del *ENUM* de Usuario

que se puede ver como una tarjeta de presentación electrónica. Dentro de la literatura se nomina a este tipo de *ENUM* como Carrier *ENUM*.

Sin embargo, todos comparten las mismas ventajas, las cuales son permitir que las llamadas que se generan se mantengan y/o terminen en Internet así como portabilidad numérica.[ref [13]]

1.4.4.5 *ENUM* VS *DUNDi*

1.4.4.5.1 MODO DE OPERACIÓN

- En un ambiente empresarial con *ENUM* varios servidores ubicados en distintas áreas geográficas deberán compartir un repositorio centralizado para localizar otros recursos. En este caso el servidor *ENUM,* para ser redundante deberá usar *switching* especializado de capa cuatro. Cuando un cliente solicite un número o extensión el servidor deberá preguntar al repositorio *ENUM* por una lista de *gateways* de salida. Todo esto debido a *ENUM* sólo cumple con la función de mapear la numeración E.164 tradicional a *URI*. De este modo este sistema resulta no redundante al tener un único punto de falla. Por el contrario en el modelo *DUNDi* no existe repositorio central. En lugar de este los nodos participan en un sistema confiable en el cual cada nodo está relacionado al menos con otro nodo en el sistema. Cuando el cliente solicita un número o extensión desconocido para este nodo preguntará a los nodos directamente conectados a él. Esos nodos a su vez preguntarán a los nodos conectados directamente a ellos. Como resultado las respuestas de los nodos son cotejadas en caché con cada nodo involucrado hasta encontrar la ubicación del número solicitado. [ref[14]]

- Para preservar la exactitud de los datos y prevenir el spam *ENUM* crea dos entidades, una entidad para supervisar estas rutas a través del modelo de seguridad SIP, y la otra encargada de emitir los certificados y verificar la autenticidad.Con *DUNDi* se reemplaza estas dos entidades con un documento el *GPA*(General Peering Agreement). Con el *GPA* en lugar de buscar la exactitud de los datos y rendición de cuentas a través de una

empresa u organización se establece las normas de interconexión con respecto a la publicación y uso de las rutas dentro del contexto E.164 del protocolo *DUNDi*. El *GPA* es ejecutado por todos los miembros del sistema bajo las siguientes responsabilidades:

- Los participantes solo publicarán números autorizados por el representante de acuerdo con una política específica.
- Los participantes y sus clientes sólo podrán utilizar las rutas aceptadas por la política de seguridad.
- Los participantes sólo deben rellenar el campo de peso conforme al acuerdo, de modo que la prioridad de respuestas puede ser establecida. De 0-99 para suscriptores y de 100-199 utilizado por el proveedor de suscriptor de servicios directos.
- Los participantes sólo deben proporcionar rutas que no requieren una cuota para conectar con el servicio anunciado.
- Los participantes deben proveer una identificación de llamadas para validar y limpiar.
- Los participantes están de acuerdo con dar a todos los miembros del Sistema la capacidad procesal respecto a la ejecución del GPA con cualquiera de los demás miembros.
- Los participantes deben proporcionar información de contacto válida a través del protocolo *DUNDi*.

ENUM no puede manejar un plan de marcado complejo para el servicio telefónico analógico tradicional, es decir *ENUM* no puede reconocer códigos adicionales que digite el usuario los cuales representen una funcionalidad para el sistema tales como registro de una extensión y redirección de una llamada. Por el contrario *DUNDi* contiene banderas las cuales permiten analizar la estructura del número digitado con el fin de incrementar los servicios de la telefonía IP.

1.4.4.5.2 PRIVACIDAD

- *ENUM* no provee privacidad para los números que se solicitan. Cualquiera puede observar que números están siendo localizados por otros usuarios, además no provee información sobre la preferencia del usuario con respecto a una llamada no solicitada.*DUNDi*como Asterisk incluyen el concepto de contextos para las extensiones. Contexto es grupo de extensiones que comparten determinadas funciones, políticas de seguridad y configuraciones, de esta manera los nodos pueden limitar el acceso a las consultas paraextensiones específicas o recursos de otros nodos determinados.

Ref[13]
http://www.sepi.escom.ipn.mx/wps/wcm/connect/2142D9804F816642911DB1724CBBC24/RISCE_N347EE.PDF?MOD=AJPERES&CACHEID=2142d9804f816642911db1724cbbc243

Ref [14]http://www.dundi.com/dundi.pdf

CAPÍTULO II

ANÁLISIS DE LOS REQUERIMIENTOS Y DESARROLLO DEL SISTEMA DE TELEFONÍA IP

2.1 DISEÑO DEL SISTEMA DE TELEFONÍA IP

Previo a realizar el diseño del Prototipo de Sistema de Telefonía IP, se analizarán los requerimientos de mayor demanda, que tienen las empresas en la actualidad al momento que deciden contratar el Servicio de Telefonía IP. Para ello se propone simular un ambiente empresarial compuesto por ocho sucursales, Estas sucursales simulan estar ubicadas en diferentes ciudades del Ecuador.

2.1.1 REQUERIMIENTOS DEL SISTEMA

Brindar un Servicio de Telefonía IP confiable y de bajo costo, a una red WAN compuesta por un total de ocho sucursales. Este sistema debe permitir:

- Realizar llamadas internas entre usuarios de la red, y llamadas hacia el medio externo.
- Movilidad de los usuarios entre las diferentes sucursales. El usuario en forma transparente, y sin mayor dificultad, deberá poder trasladar y registrar su número telefónico, en cualquier sucursal donde él se encuentre. Este caso es común, cuando un usuario es transferido a otra sucursal, o cuando temporalmente, por motivos de trabajo tiene que viajar a otra dependencia de la empresa.
- Disminuir los problemas de fallas en el servicio, considerando las pérdidas que este problema ocasiona a nivel empresarial.
- Garantizar la optimización de recursos en la implementación del sistema, y en el posterior funcionamiento del mismo.

2.1.1 SOLUCIÓN

De acuerdo a los requerimientos, se plantea:

- Implementar y configurar un Sistema de Telefonía IP distribuido, aprovechando la topología en malla de la red. Un sistema distribuido, evita tener un único punto de falla en la red.

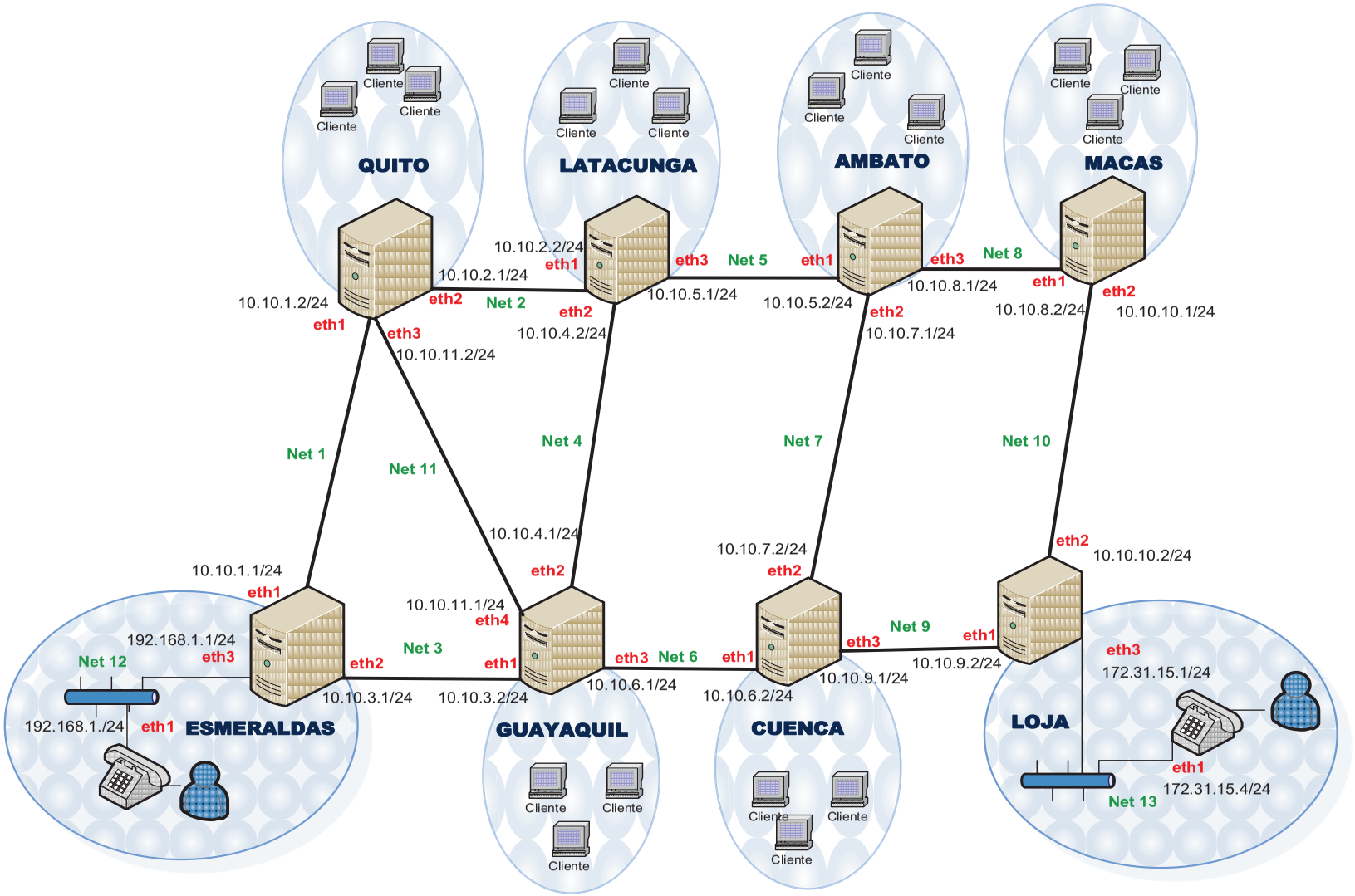
- Utilizar la plataforma operativa Linux para el funcionamiento de cada nodo en la Red. Al emplear software libre, como es el caso de Linux, se elimina el alto costo de licenciamiento que imponen los sistemas operativos o soluciones propietarias, además se elimina la dependencia en el uso de hardware específico para el sistema de telefonía IP, brindando a la empresa la posibilidad de adquirir hardware de distintas marcas y precios de acuerdo a sus necesidades

- Implementar y configurar en cada Central de Telefonía IP para su funcionamiento la solución basada en open source Asterisk de Digium. Para el manejo de sesiones de VoIP en el sistema, emplear el protocolo SIP para los usuarios locales de cada nodo, y el protocolo IAX para la comunicación entre centrales de telefonía IP.

- Configurar en cada Central telefónica el protocolo *DUNDi*, para el descubrimiento de rutas de llamadas. *DUNDi* y la topología en malla permitirán una comunicación punto a punto entre las ocho centrales telefónicas, integrándolas en un solo dominio para facilitar el intercambio de información en todo el sistema de telefonía IP.

- Proveer movilidad a los usuarios dentro de la Red, permitiéndoles realizar un enrolamiento dinámico a través de la Central de telefonía IP a la cual ellos se conectan para usar el servicio.Para identificar a cada usuario proporcionar un número de extensión personal único. Este proceso involucra configurar en cada Central de Telefonía IP un servidor de redirección mediante la herramienta *SIPP*, además de la programación del plan de marcado.

2.1.2 ESQUEMA DE RED



- **Explicación:**

Para realizar el esquema de red se tomó como referencia la localización de las ciudades de acuerdo a la geografía del Ecuador. La topología que se describe en el esquema de red es en malla, con el fin de garantizar el funcionamiento de un sistema distribuido y redundante tal como se expone en la solución del sistema de telefonía IP, mencionado en el literal 2.1.1.

Adicionalmente para el diseño de red se toma en cuenta el optimizar los enlaces WAN entre los nodos que la conforman, debido al alto costo operativo que los enlaces representan en un ambiente empresarial real. Para definir el número exacto de enlaces WAN a crearse se establecen los nodos críticos dentro del sistema, es decir aquellas ciudades que deben tener una comunicación 100 % garantizada, libre de errores a causa de la información vital que transmiten. Las ciudades que se asume entran en este caso son: Quito y Guayaquil, por esta razón entre los nodos de Quito y Guayaquil se crea un enlace dedicado. Los seis nodos restantes en la red se asume que no entran en el caso de ser nodos críticos, por lo cual no poseen enlaces WAN dedicados garantizándose únicamente dos caminos alternos para la comunicación de cada nodo. Este análisis se realiza en base a las consideraciones previas que se estudian en Empresas e Instituciones antes de plantear un esquema de red funcional a bajo costo operativo.

2.1.3 DIRECCIONAMIENTO IP

El direccionamiento IP que se aplica a la red es con direcciones IP estáticas, es decir con direcciones IP fijas por cada interfaz de red en los nodos y teléfonos IP de los usuarios. Se emplean en la red Direcciones IP privadas:

- Clase A: (10.0.0.0-10.255.255.255 /24) Para los enlaces WAN entre los ocho nodos de la red.
- Clase B: (172.31.15.0 /24) Para la red LAN de la ciudad de Loja.
- Clase B: (192.168.1.0 /24) Para la red LAN de la ciudad de Esmeraldas.

UBICACIÓN	INTERFAZ DE RED	DIRECCIONES IP	MÁSCARA
Esmeraldas	eth1	10.10.1.1	255.255.255.0
	eth2	10.10.3.1	255.255.255.0
	eth3	192.168.1.1	255.255.255.0
Quito	eth1	10.10.1.2	255.255.255.0
	eth2	10.10.2.1	255.255.255.0
	eth3	10.10.11.2	255.255.255.0
Latacunga	eth1	10.10.2.2	255.255.255.0
	eth2	10.10.4.2	255.255.255.0
	eth3	10.10.5.1	255.255.255.0
Guayaquil	eth1	10.10.3.2	255.255.255.0
	eth2	10.10.4.1	255.255.255.0
	eth3	10.10.6.1	255.255.255.0
	eth4	10.10.11.1	255.255.255.0
Ambato	eth1	10.10.5.2	255.255.255.0
	eth2	10.10.7.1	255.255.255.0
	eth3	10.10.8.1	255.255.255.0
Cuenca	eth1	10.10.6.2	255.255.255.0
	eth2	10.10.7.2	255.255.255.0
	eth3	10.10.9.1	255.255.255.0
Macas	eth1	10.10.8.2	255.255.255.0
	eth2	10.10.10.1	255.255.255.0
Loja	eth1	10.10.9.2	255.255.255.0
	eth2	10.10.10.2	255.255.255.0
	eth3	172.31.15.1	255.255.255.0
TLOJ	eth1	172.31.15.4	255.255.255.0
TESM	eth1	192.168.1.4	255.255.255.0

Tabla2.1 Direccionamiento IP de la red

2.2 SELECCIÓN DE LA HERRAMIENTA DE VIRTUALIZACIÓN

Para implementar el Prototipo del Sistema de Telefonía IP, se utilizará un software de virtualización. Este software permite al sistema de hardwareejecutar, múltiples instancias de diferentes sistemas operativos instalados en máquinas virtuales, deforma concurrente. Lossistemas operativos no interfieren entre sí, ni alas aplicaciones.

Una máquina virtual es una máquina que crea un entorno virtualizado sobre la plataforma de computadora para que el usuario final pueda operar software en un ambiente controlado.

La herramienta de virtualización que se seleccione deberá permitir:

1. Interconectar una red integrada por un total de ocho centrales de telefonía IP, optimizando los recursos del computador (host anfitrión) sobre el cual se implemente la simulación.

2. Configurar en cada central IP Asterisk, *DUNDi*y *SIPP*.

3. Conectar al menos dos usuarios a la red para realizar las pruebas de funcionamiento del Sistema de Telefonía IP en general.

2.2.1 COMPARACIÓN DE SOFTWARE DE VIRTUALIZACIÓN

2.2.1.1 Virtual Network User Mode Linux (*VNUML*)

2.2.1.1.1 Definición

Virtual Network User Mode Linux (*VNUML*) es una herramienta de virtualización creada para Linux. *VNUML* funciona sobre su propia interfaz de llamadas al sistema con el fin de obtener varias consolas Linux invitadas operando simultáneamente en una sola máquina física llamada anfitrión. Al emplear *VNUML* se aprovecha el hardware existente sin causar daños al sistema operativo Linux que corre en el host anfitrión.

VNUML corre como un proceso normal de Linux para solicitar al anfitrión recursos de: memoria, red, salida a internet entre otros. Los procesos

*VNUML*generados por cada máquina invitada utilizan la memoria de la máquina anfitrión por slots de tiempo para ejecutar sus propios procesos a la vez de manejar también llamadas al sistema.

2.2.1.1.2 Aplicaciones

- *Hosting* para servidores virtuales. Esta aplicación es útil, para las empresas dedicadas a la provisión de servicios de hospedaje(*hosting*), puesto que implica optimizar los recursos, al emplear un único equipo anfitrión sobre el que implementar un conjunto de servidores virtuales.

- Creación de *honeypots* (sistemas para probar la seguridad de una máquina sin comprometerla).

- Desarrollo de *kernels* (Núcleo del sistema operativo Linux).

- Prueba de aplicaciones de red. *VNUML* permite establecer un ambiente de pruebas, con el fin de estudiar la funcionalidad de aplicaciones de red tales como: servidores web, *DNS*, mail, etc.

2.2.1.1.3 Ventajas

- *VNUML*además de permitir crear máquinas virtuales, provee mecanismos para la creación de redes virtuales. Estas redes pueden ser utilizadas para interconectar máquinas virtuales entre sí formando diferentes topologías deacuerdo a los requerimientos del sistema que se desee simular. Para ello se hace uso de las interfaces físicas del equipo anfitrión.

- Mejor aprovechamiento de los recursos de hardware de las máquinas. Muchas empresas en la actualidad emplean la consolidación de servidores, para agrupar todos los servidores en una sola máquina, a la vez que se facilita la realización de copias de seguridad ya que solo se debe almacenar el *backup* de un sólo fichero.

- Seguridad en la ejecución de servicios de red. Los servicios de red de las máquinas anfitrionas se ejecutan en diferentes procesos *VNUML* en la

máquina nativa sin comprometer la seguridad o estabilidad mutua entre las máquinas.

- El consumo de disco duro no es directamente proporcional al número de máquinas virtuales que se simulen. *VNUML* brinda optimización en el uso de disco duro, al utilizar un mismo sistema de ficheros (*filesystem*), el cual va a dar soporte a todas las máquinas virtuales. Este *filesystem* es denominado de escritura-en-copia (*Copy-on-write*) o*COW*, es compartido por todas las máquinas, pero a la vez se encarga de almacenar los cambios que cada una hace al mismo en forma independiente. Este proceso es totalmente transparente para el usuario

2.2.1.1.4 Desventajas

- *VNUML* sólo permite crear máquinas virtuales basadas en Linux. No es posible crear máquinas virtuales basadas en otros sistemas operativos por ejemplo Windows.

- *VNUML* no ofrece una interfaz gráfica de usuario, únicamente presenta un terminal de comandos, lo cual demanda un conocimiento sólido de manejo de Linux.

- Al instalar nuevas aplicaciones en una máquina *VNUML* muchas de las veces se requiere la instalación de herramientas de desarrollo extras. Estas herramientas no pueden ser seleccionadas previamente a la instalación y ellos puede originar problemas de dependencias no instaladas.

- Al ser *VNUML* un producto de código abierto, no proporciona soporte oficial ni comercial.

2.2.1.1.5 Funcionamiento

*VNUML*está formada por dos componentes:

- Un lenguajesimple y descriptivo basado en *XML* [10](*eXtended Markup Language*).*XML* es el lenguaje base para la descripción de escenarios de simulación, utilizando este lenguaje se debe definir el archivo .*XML* para emular el sistema de prueba.

- Un parser(escrito en Perl[11]), el cual se encarga de procesar el archivo *.XML*, construyendo la simulación ygestionándola. Además se encarga de ocultar los detalles complejos al usuario.El procesado se realiza ejecutando el comando *vnumlparser.pl* en el equipo anfitrión. El programa trabaja en dos etapas:

  1. Primero un parser *XMLDOM* (*Dinamic Object Model*) procesa el archivo *.XML*, construyendo en la memoria del programa un árbol DOM con la información del archivo. En caso de existir errores en la descripción del archivo, el programa interrumpe la ejecución usando un parser validador basado en *DTD* (*Document Type Definition*).

  2. Una vez creado el árbol *DOM* con los datos de la simulación, la lógica de control del programa se encarga de acceder a esos datos e invocar determinados comandos en el equipo anfitrión (relacionados con *VNUML* y la gestión de redes virtuales mediante bridges virtuales), que conducen a la creación, gestión o eliminación de la simulación. La lógica de control, implica tres modos de funcionamiento:

     - Construcción de topología (-t).
     - Ejecutar comandos (-x).
     - Parar la simulación (-d).

2.2.1.2 *XEN*

2.2.1.2.1 Definición

XEN es una solución *open source*desarrollada por la Universidad de *Cambridge* su propósito es ejecutar varios sistemas operativos clientes junto con un sistema operativo que tiene acceso directo a todos los dispositivos de hardware. Con esta herramienta seobtiene para-virtualización a nivel de sistema operativo, llegando a conseguir un rendimiento próximo a la ejecución nativa. Las arquitecturas sobre las que funcionan son *x86, x86_64, IA64, AMD64*y *PowerPC*.

2.2.1.2.2 Ventajas

- *XEN* brinda la posibilidad de migrar máquinas virtuales de un host a otro en caliente, esto permite manejar y distribuir la carga del host. Por ello es muy útil en el mantenimiento de servidores y hosts.

- Aprovecha las capacidades de virtualización del*VT* de *Intel* y el *AMD* Virtualization.

2.2.1.2.3 Desventajas

- *XEN* requiere realizar modificaciones en el sistema operativo alojado por ello solo pueden virtualizarse en *XEN* sistemas operativos parchados. En el caso de Linux al ser open source no hay inconveniente pero desde el punto de vista de otros sistemas operativos que no sean de fuente abierta como Windows el proceso de parchar es más complicado. Es posible ejecutar Windows como S.O. alojado en *XEN*, pero solo en sistemas hardware que soporten la tecnología *Vanderpool* de *Intel* o Pacifica de *AMD*.

- *XEN* no soporta ACPI[12] o APM[13] para un correcto funcionamiento en portátiles.

2.2.1.3 Vmware Workstation(VMWARE)

2.2.1.3.1 DEFINICION

Es un sistema de virtualización completa. Esta solución es comercial y puede funcionar en Windows, Linux, y en la plataforma *Mac OS X* que corre en procesadores *INTEL*, bajo el nombre de *VmwareFusion*.

2.2.1.3.2 Funcionamiento

Se basan en un *hypervisor* que mediante complicadas técnicas como traps y análisis del código antes de su ejecución detecta el código que no puede ejecutarse directamente porque afectaría a todo el sistema, modificando dinámicamente las instrucciones conflictivas. El rendimiento varía mucho según lo avanzado que sea el*hypervisor*. *Vmware* permite instalar un sistema operativo sobre un virtualizador del mismo modo que sobre un PC, la ventana del

virtualizador se asemeja al monitor de un PC, por ello se puede entrar en la BIOS, reiniciar el computador y demás funciones.

Vmware permite tres modos de instalación: "Workstation", GSX, o ESX. Cada uno con diferentes capacidades, según las necesidades del usuario.

2.2.1.3.3 Ventajas

- El entorno virtualizado completo se respalda en un fichero, esto significa que el sistema operativo alojado, y el hardware virtual puede migrarse con facilidad y rapidez a una nueva máquina anfitrión

2.2.1.3.4 Desventajas

- *Vmware* consume gran cantidad de recursos en la ejecución, esto afecta directamente en rendimiento y desempeño de las máquinas virtuales.

A continuación se realiza un análisis comparativo de las herramientas de virtualización expuestas VNUML, XEN y VMWARE con el objetivo de seleccionar la mejor opción conforme a los requerimientos necesarios para implementar el prototipo del sistema de telefonía IP.

XML[10] Propuesto por el W3C en 1998 tras el auge del lenguaje HTML (*Hypertext Markup Language)* en la web, *XML* surge como un estándar abierto para el intercambio de información por Internet basado en etiquetas de texto.

Perl [11]Es un lenguaje de programación diseñado por Larry Wall en 1987. Perl toma características del lenguaje C, del lenguaje interpretado shell (sh), AWK, sed, Lisp y en un grado inferior, de muchos otros lenguajes de programación.

ACPI[12] Interfaz Avanzada de Configuración y Energía. Es un estándar resultado de la actualización de APM a nivel de hardware, que controla el funcionamiento del BIOS y proporciona mecanismos avanzados para la gestión y ahorro de la energía.

APM[13]Es un API desarrollado por Intel y Microsoft que permita que el BIOS administre la energíatal como reducir la velocidad de la CPU después de un período de inactividad para conservar corriente eléctrica, especialmente para las computadoras portátiles.

2.2.1.4 VNUML vs VMWARE y XEN

ITEMS	*XEN*	*VMWARE*	*VNUML*
Tipo de Virtualilzación	Virtualización completa Para-virtualización	Virtualización completa	Para-virtualización
Tipo de licencia	GNU/GPL	Licencia privativa	GNU/GPL
Sistemas Operativos a virtualizar	Windows Linux	Windows Linux	Linux
Monitoreo	Amplia variedadde herramientas de monitoreo	Una sola herramienta de monitoreo	Una sola herramienta de monitoreo por medio de ssh.
Funciones	Reconfiguración de máquinas en tiempo real. Utiliza vnc para que el sistema guest se veacomo una tarjeta VGA. Permite migrar en caliente un dominio de un servidor a otro.	Inflexibilidad al momento de reconfigurar máquinas. Emula una tarjeta gráfica SVGA completa. Presenta una interfaz gráfica de usuario. No Permite migrar en caliente.	La reconfiguración de máquinas sólo requiere la edición del archivo *.XML*. No emula una tarjeta gráfica su interacción con el usuario es por medio de línea de comandos. No Permite migrar en caliente.
Entornos de Red	Únicamente se puede crear redes simples con un limitado número de hosts dependiendo de la memoria RAM.	Únicamente se puede crear redes simples con un limitado número de hosts dependiendo de	Permite la fácil creación de redes virtuales complejas.

		la memoria RAM.	
Consumo de recursos físicos del computador	Excelente rendimiento, 48.6% mayor a *Vmware*. 8 VM en 2GB de RAM.	Consume altos ciclos de procesador y tiempo de ejecución para su virtualización. 5 VM en 2GB RAM.	Menor consumo de recursos en relación a *vmware*.
Rendimiento	Rendimiento comparable al código nativo. Permite ejecutar varias VMs a la vez.	La arquitectura *XEN*/x86 es bastante similar a la x86 nativa	Buen rendimiento.
Requisitos de Instalación	Pentium 4, 1 GB de RAM.	Pentium 4, 1 GB de RAM. 10 Gb de espacio en el disco duro. Sistema Operativo: Windows y Linux.	Pentium 4, 1 GB de RAM Sistema Operativo: Linux.
Costo	Accesible (Software Libre)	Precio elevado	Accesible (Software Libre)

Tabla 2.2 Comparación de Herramientas de virtualización

2.2.2 SELECCIÓN DEL SOFTWARE DE VIRTUALIZACIÓN

En función de la red que se desea simular la mejor opción para implementar el Prototipo de Telefonía IP es ***VNUML***. Las razones para su elección son las siguientes:

- *VNUML*brinda la gran ventaja de crear redes virtuales complejas, interconectando varios nodos de una red WAN sin consumir muchos recursos de la máquina física. Cabe resaltar que con esta herramienta de virtualización, no se tiene un ambiente gráfico del sistema operativo instalado, en cada Central de Telefonía IP. Este ambiente gráfico se podría visualizar al utilizar *XEN*o*VMWARE*, pero su costo es el consumo excesivo de recursos del Computador, a la vez de limitar al mínimo el número de nodos a simular, haciendo imposible la simulación del Sistema de Telefonía IP. Se cumple así el primer requisito para la implementación del sistema, interconectar una red integrada por un total ocho centrales de telefonía IP, optimizando los recursos de la máquina física

- En cuanto al segundo requisito para la configuración de Asterisk, *DUNDi* y *SIPP*, únicamente es necesario realizar la instalación de Asterisk y sus dependencias sobre el *filesystem*(sistema de archivos) que se va a utilizar para la simulación del Sistema.

- Finalmente la conexión a la red de los usuarios queda garantizada, debido a la optimización de recursos para simular el sistema.

2.3 DESCRIPCIÓN DEL FUNCIONAMIENTO DEL PROTOTIPO

El prototipo del Sistema de telefonía IP está formado por un total de ocho nodos, los nodos simulan estar ubicados en diferentes ciudades del Ecuador.

Cada nodo en la red cumple con dos funciones:

- Router mediante la implementación del protocolo de enrutamiento RIP.

- Central de Telefonía IP basada en la solución open source Asterisk.

Este escenario es implementado con el software de virtualización seleccionado *VNUML*, mediante la edición de un archivo *XML* llamado esquema_red_tesis_1.*XML*. Este archivo contiene la descripción de la red, es decir especifica el número de redes y nodos a crearse con sus respectivas

interfaces de red, además de indicar el *Filesystem* y el *Kernel* que empleará cada nodo para su simulación.

2.3.1 FUNCIONAMIENTO DE UNA CENTRAL DE TELEFONÍA IP

En cada Central de Telefonía IP se configura:

a) El Dial-Plan con la edición del archivo extensions.conf. Las funciones principales que cumple este archivo, en el esquema propuesto, se pueden resumir en la siguiente lista:

- Conexión básica entre extensiones en una misma central de telefonía IP.
- Búsqueda de extensiones enroladas dentro del dominio *DUNDI*.
- Registro y enrolamiento de extensiones personales.
- Activación del servidor de redirección.
- Des-registro y Des-enrolamiento de extensiones de manera manual o automática.

Configurados los servicios básicos de conectividad, enrutamiento IP y de llamada, este archivo coordina de manera funcional cada una de las labores del esquema de telefonía redundante.

b) El protocolo *DUNDi* con la edición del archivo dundi.conf, para el descubrimiento de ruta de llamada, de acuerdo a la solución propuesta al inicio del capítulo. En este archivo se configura las centrales de telefonía adyacentes. Por ejemplo de acuerdo al diagrama de Red expuesto, la Central Telefónica de Quito (UIO) deberá configurar en dundi.conf los nodos cercanos, en este caso Esmeraldas (ESM), Latacunga (LAT) y Guayaquil (GYE). Esta configuración permite el intercambio de información entre las centrales telefónicas, información que detalla la ubicación de un usuario enrolado en el sistema, haciendo posible llegar al destino de una llamada para establecer una comunicación.

c) El protocolo SIP, mediante la edición del archivo sip.conf, para definir las extensiones que se registran y enrolan, además para determinar el contexto mediante el que se atenderá la llamada en el dialplan.

d) El protocolo IAX editando el archivo iax.conf para la comunicación entre Centrales de Telefonía IP.

2.3.2 FUNCIONAMIENTO DE UN CLIENTE DE TELÉFONIA IP

- Los teléfonos IP serán simulados en una máquina virtual sobre la cual está corriendo una instancia SIP. Esta simulación será realizada mediante la herramienta *SIPP*. *SIPP*permite simular la generación de tráfico IP, es decir nos permite generar los mensajes que utiliza SIP para iniciar, establecery terminar una llamada. Un teléfono IP se identifica con un número de extensión físico, este número será utilizado por el administrador del sistema para registrar el teléfono IP a la central telefónica asignada.

2.3.3 FUNCIONAMIENTO DE *SIPP*

Las acciones que se implementarán utilizando *SIPP* son:

2.3.3.1 Registro de un Teléfono IP

- EL proceso de registro de un terminal de VoIP o como en este caso de una instancia *SIPP*, permite que luego de que se realice este proceso, la central telefónica sepa cómo alcanzar esa terminal o instancia, identificándola a través del número de la extensión físicao estática.

2.3.3.1.1 Procedimiento

El proceso de registro implica, el envío de un mensaje SIP del tipo *REGISTER* a la central de telefonía IP. La central telefónica almacena en una especie de mapa la siguiente información:

- ***Dirección IP*** desde donde se originó la petición de registro, la dirección IP del terminal de VoIP (o de la instancia *SIPP* en esta simulación).
- ***Puerto local*** desde donde se originó la petición de registro.

- ***Extensión estática*** que se está registrando en la central.

2.3.3.1.2 Ejemplo de uso

De este modo, por ejemplo cuando registramos la extensión estática 6000 en la central telefónica de Esmeraldas a través de una instancia *SIPP*, registraríamos estos datos:

```
6000 5200 3200 192.168.1.4
```

Si la central telefónica, mediante su plan de marcado, recibe una llamada hacia la extensión 6000, esta central sabe cómo direccionar la llamada gracias a la información registrada con anterioridad. Es así como una llamada hacia la extensión 6000 irá dirigida en realidad a la dirección IP 192.168.1.4 en el puerto 5200 (socket 192.168.1.4:5200), permitiendo el establecimiento de la comunicación. El puerto de media que se configura con el valor 3200, es el puerto que se utilizará para el tráfico RTP cuando se establece la comunicación entre usuarios del sistema.

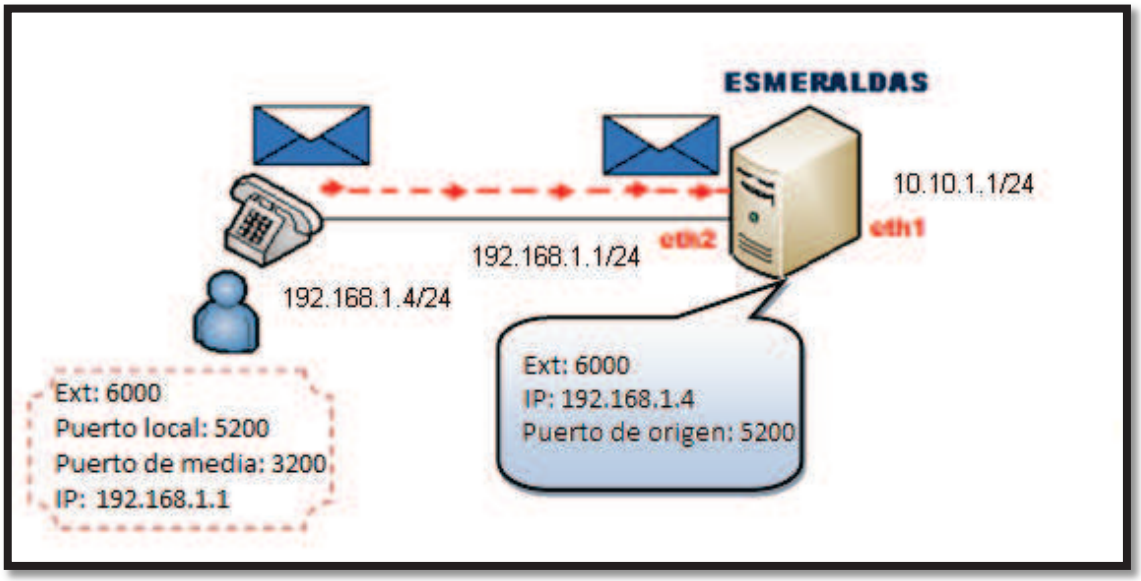


Figura 2.1Proceso de Registro de un teléfono IP

2.3.3.2 Enrolamiento de un usuario al Sistema

El enrolamiento es un proceso que le permite al usuario que lo lleva a cabo ser alcanzado vía telefónica desde cualquier otra extensión dentro del dominio de VoIP, mediante el uso de su número personal (dinámico).

El usuario se enrolaría empleando un teléfono IP, que está registrado previamente en la central telefónica (mediante su extensión estática), a través del marcado del código de enrolamiento más su número de extensión personal. Tal como está descrito en el literal 2.1.1., los usuarios en el sistema se identifican utilizando su número de extensión personal.

2.3.3.2.1 Procedimiento

El enrolamiento, involucra realizar una llamada desde un cliente de VoIP hacia la central telefónica donde el usuario desea enrolarse.El teléfono desde el que se marca la llamada está registrado previamente en la central telefónica con su número de extensión estática, de ese modo, y tomando en cuenta que las extensiones registradas SIP se procesan en el contexto local, en el archivo extensions.conf, la llamada es procesada en ese contexto. El usuario, al realizar la llamada, marcará el código 456más su extensión personal. Se envía entonces desde el cliente un mensaje SIP del tipo *INVITE* a la Central, esta responde con los códigos de respuesta *TRYING* y RINGING para indicar el progreso de la petición.El código 456 marcado por el usuario indica a la central telefónica que diferencie la llamada como una solicitud de enrolamiento por lo tanto, será atendida por el contexto enrolamiento el cual se incluye dentro del contexto local en el plan de marcado. Finalmente para indicar que la petición de iniciar sesión ha llegado a su fin con éxito la Central envía la respuesta *OK*, al igual que el cliente. Para terminar la sesión el cliente envía un mensaje SIP del tipo BYE a la central y espera recibir la respuesta *ACK* de mensaje recibido para finalizar.Básicamente se ejecuta un script mediante el cual se afilia desde la misma central telefónica, de manera dinámica, la extensión personal con la dirección IP de la misma central y con un puerto origen (puerto local) definido en el registro).

De acuerdo al contexto enrolamiento,el cual se encarga de procesar el enrolamiento, en este proceso se realizan dos pasos:

- Primero se verifica si la extensión personal, está ya enrolada en alguna central telefónica del dominio *DUNDi*, mediante el comando ***DUNDILOOKUP.*** En caso de estar enrolada, se procede a des-enrolar de la central telefónica en la que se encuentre (el proceso de des-enrolamiento será explicado en el siguiente literal) si la extensión, no se encuentra en ninguna central del dominio, se ejecuta en forma normal el enrolamiento. Se asigna entonces el puerto local y puerto de media para que la extensión pueda ser localizada en el sistema.

  Segundo enrolada la extensión personal en la central telefónica, se inicia en el puerto local asignado el Servidor de Redirección. Este servidor se encarga de asociar la extensión personal del usuario, con la extensión estática del terminal de VoIP, con el fin de re-direccionar la llamada desde la central hacia el terminal de VoIP. (El proceso del Servidor de Redirección se explicará más adelante).

La central telefónica almacena como resultado la siguiente información:

- ***Dirección IP*** de la central telefónica en la cual se enrola el cliente
- ***Puerto local*** asignado en el proceso de enrolamiento por el dial-plan. .
- ***Extensión Personal*** del usuario.

2.3.3.2.2 Ejemplo de uso:

Para afiliar la extensión personal 7000 en la central telefónica de Esmeraldas, se toma el terminal de VoIP (con extensión estática 6000) que ya ha sido registrado en la central anteriormente, como se describe en el literal *2.3.3.1.2.* Se procede entonces a realizar la llamada enviando los siguientes datos:

```
7000 4500 3500 10.10.1.1 6000
```

La central recibe la llamada y procede a enrolar la extensión personal 7000 asociada con la extensión estática 6000 de acuerdo a la configuración del plan de marcado, a la vez en el puerto 5600 se inicia el servidor de redirección, el cual se encarga de direccionar la llamada desde la central hasta el terminal de VoIP.Cabe resaltar que el puerto local utilizado por el cliente para establecer la sesión con la central no es el mismo puerto local que se asigna al enrolar la extensión personal. El puerto local asociado a la extensión en la central de telefonía corresponde al valor asignado en el dial-plan para este proceso.

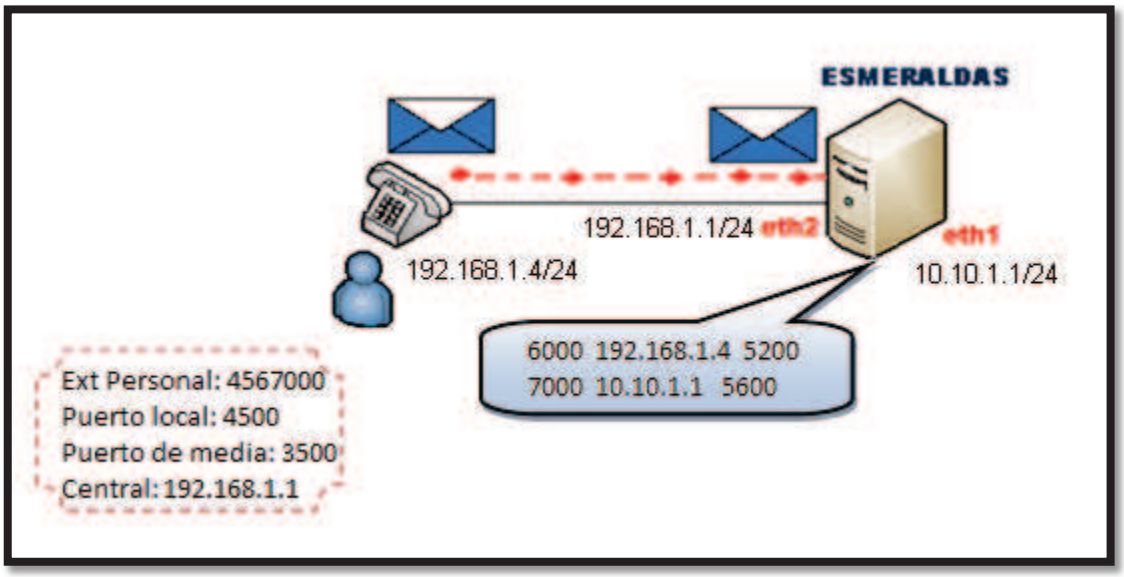


Figura 2.2 Proceso de Enrolamiento de un teléfono IP

2.3.3.3 Des-enrolamiento de un usuario del Sistema de Telefonía IP

El des-enrolamiento es un proceso que lo ejecuta el usuario que desea salir del sistema de telefonía IP, al des-enrolarse el usuario ya no será alcanzado por ninguna extensión en el dominio de VoIP, debido a que su extensión personal ya no estará enrolada en ninguna central de telefonía IP del sistema.

2.3.3.3.1 Procedimiento

El proceso de des-enrolamientoal igual que el proceso de enrolamiento involucra realizar una llamada. El usuario marcará el código de des-enrolamiento seguido de su número de extensión personal, esta petición será procesada por el contexto local del plan de marcado, el cual a su vez incluye el contexto des-enrolamiento el mismo que atenderá la llamada. El cliente

envía entonces un mensaje SIP del tipo *REGISTER* a la central telefónica de donde se desea des-enrolar y espera recibir el código de respuesta 200 (*OK*) que le confirme el des-enrolamiento de su extensión personal.

2.3.3.3.2 Ejemplo de uso

Se procede a des-enrolar el número de extensión personal 7000, el cual fue enrolado en la central telefónica de Esmeraldas de acuerdo al ejemplo de uso del literal 2.3.3.2.2. Se marca desde el terminal de VoIP el código de des-enrolamiento 123 más la extensión personal, puerto local, puerto de media, dirección IP de la central telefónica y número de extensión estática del teléfono IP.

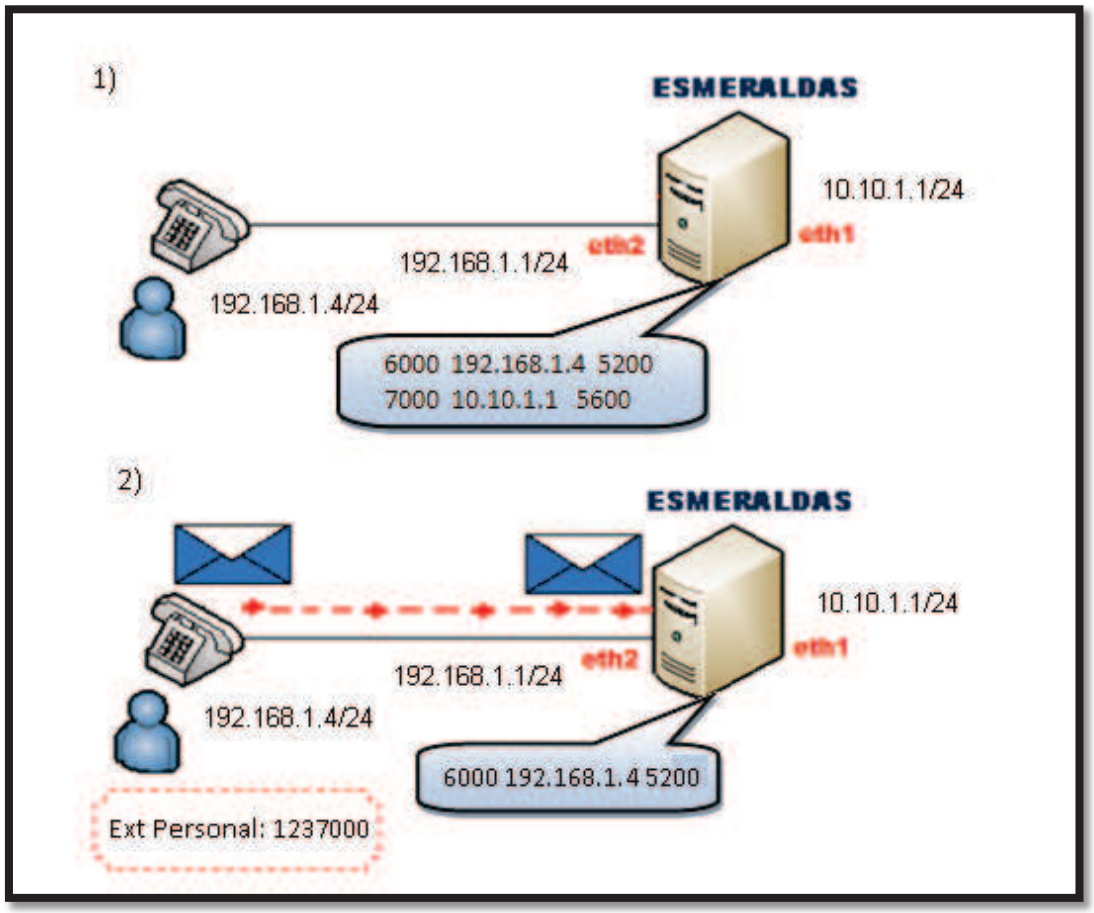


Figura 2.3 Proceso de Des-enrolamiento de un teléfono IP

2.3.3.4 Esperar llamadas

Cuando un usuario se enrola en el dominio de VoIP, puede realizar el proceso de esperar llamada. Este proceso es indispensable para que el usuario pueda recibir llamadas provenientes de otros usuarios en el sistema, a través de su terminal de VoIP, sino lo activa las llamadas no serán establecidas.

2.3.3.4.1 Procedimiento

El proceso de esperar llamada se activa en un terminal de VoIP registrado en una central telefónica. Al registrar el terminal la central telefónica conoce el socket mediante el cual el terminal establecerá la comunicación con otros terminales en el sistema. Para ejecutar este procedimiento es necesario entonces conocer el puerto local, puerto de media y dirección IP del terminal de VoIP. Al activar este procedimiento, el terminal de VoIP estará esperando la llegada de un mensaje SIP del tipo *INVITE*, que le indica la solicitud de llamada por parte de otro usuario, al recibir este mensaje el terminal responde con los códigos de respuesta TRY, RINGING y *OK* para indicar que se puede establecer la conexión, el solicitante entonces le envía un mensaje *ACK* de confirmación y la llamada es establecida.

2.3.3.4.2 Ejemplo de uso

Para activar el proceso de esperar llamada, se toma el terminal de VoIP con extensión estática 6000 registrado en la central telefónica de Esmeraldas. En este caso el socket de conexión es 192.168.1.4:5200, y además el puerto de media es 3200. Con estos datos se procede a ejecutar el proceso en el terminal de VoIP, permitiendo así la recepción de llamadas en este terminal.

5200 3200 192.168.1.4

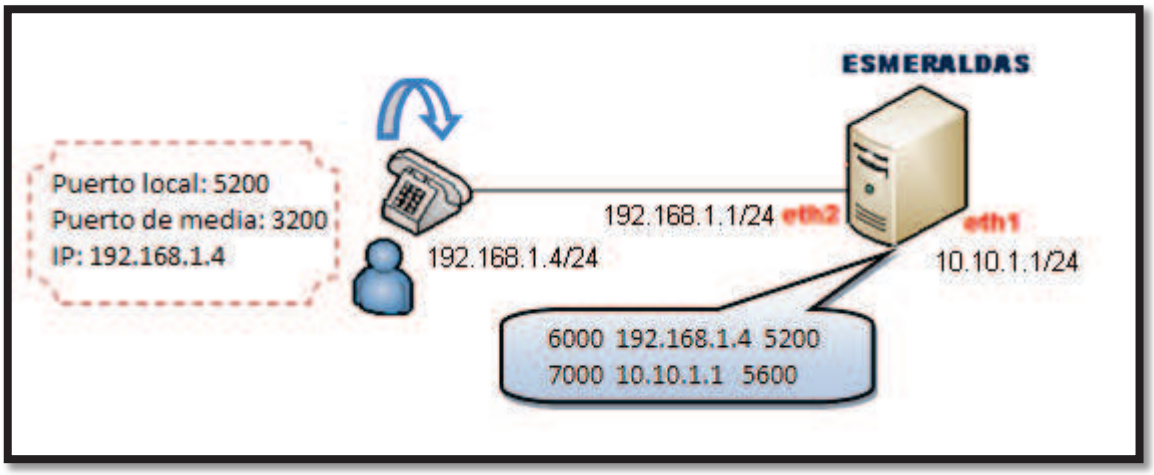


Figura 2.4 Proceso Esperar Llamada en un teléfono IP

2.3.3.5 Llamadas a usuarios dentro del Sistema de Telefonía IP

Las llamadas son realizadas por los usuarios que previamente se han enrolado a su central telefónica local. El usuario por lo tanto, debe conocer la extensión personal de a quién desea llamar, y digitarla desde su terminal de VoIP, para establecer la comunicación con el destino. A la vez para que la comunicación pueda darse,el usuario destino debe activar el proceso esperar llamada en su terminal de VoIP. (Este proceso será descrito en el siguiente literal) Al enrolarse los usuarios al sistema deciden si desean activar el proceso de esperar llamada, sino lo activan únicamente podrán realizar llamadas pero no recibirlas. Cuando la central recibe una solicitud de llamada hacia una extensión personal de usuario, esta busca la extensión empleando el comando*DUNIDLOOKUP*a nivel local, sino la localiza procede a ejecutar *DUNDILOOKUP* en el dominio *DUNDI* hasta encontrar la central telefónica en la cual se encuentra enrolada la extensión personal.

2.3.3.5.1 Procedimiento

El cliente que realiza la llamada, envía esta solicitud a su central telefónica local. La central recibe entonces, un mensaje SIP del tipo *INVITE*, y responde con el código de respuesta *TRYING*,para indicar al cliente que originó la llamada, el envío del mensaje*INVITE* hacia el cliente destino. Para localizar a la extensión personal destino de acuerdo a la configuración del plan de marcado, se ejecuta el macro *DUNDi-lookup,* el cual primero realiza la

búsqueda en la central telefónica local,y en caso de no ser favorable el resultado busca la extensión en el dominio *DUNDI*, hasta encontrar la central telefónica destino donde está enrolada dicha extensión. El cliente destino al recibir el mensaje INIVITE desde la central telefónica donde él esta enrolado, responde a la central con los códigos de respuesta *TRYING*, RINGING y finalmente *OK*para que la central telefónica confirme al cliente solicitante que acepto la petición de iniciar sesión. Al recibir el cliente que originó la llamada esta confirmación, esdecir el mensaje *ACK*se establece la comunicación y empieza el diálogo entre los dosusuarios. Finalizado el diálogo el cliente que solicitó la llamada, envía un mensaje BYE al cliente destino y espera recibir el mensaje de confirmación *ACK* para dar por terminada la sesión.

2.3.3.5.2 Ejemplo de uso

En el siguiente escenario se tiene a dos usuarios en el Sistema de Telefonía IP, un usuario A en la central telefónica de Esmeraldas y un usuario B en la central telefónica de Quito.

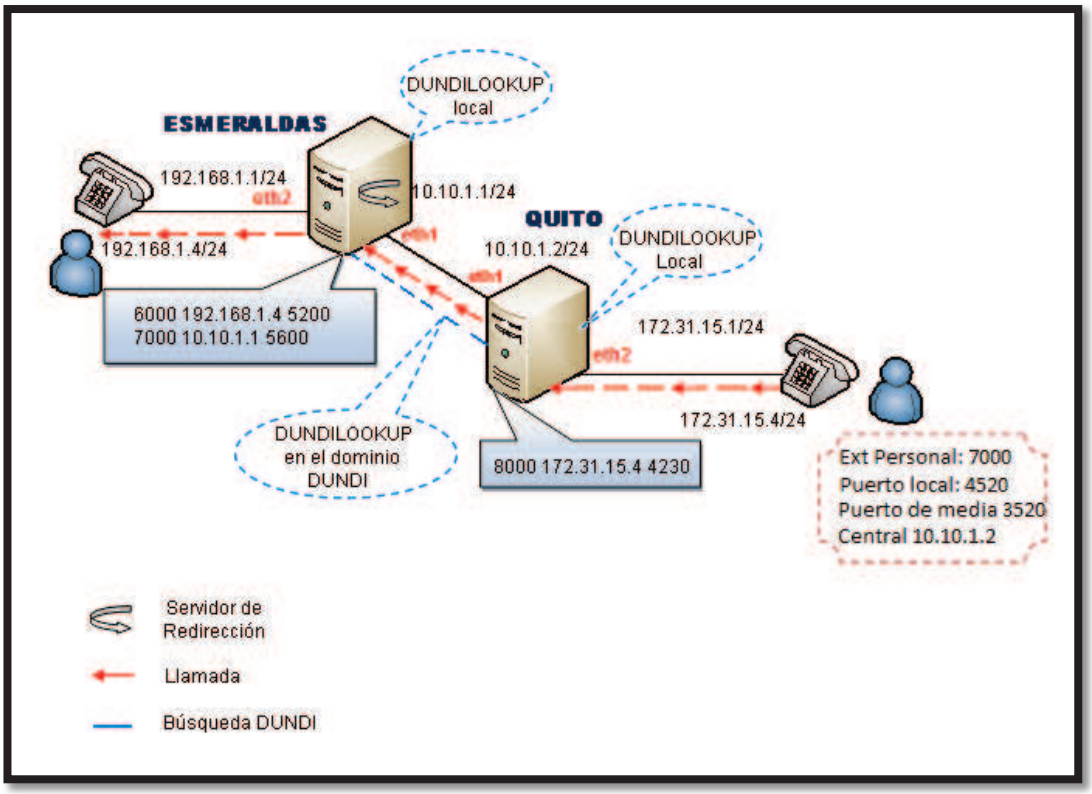


Figura 2.5 Proceso Llamadas a usuarios dentro del Sistema de Telefonía IP

El usuario A está enrolado en su central telefónica local con su número de extensión personal 7000 en el puerto local 5600 (socket 10.10.1.1:5200), a la vez el servidor de redirección escucha solicitudes de llamada en este puerto. El usuario A se ha enrolado a través de un terminal de VoIP, el cual ha sido registrado en la central telefónica de Esmeraldas con el número de extensión estática 6000 en el puerto local 5200 (socket 192.168.1.4:5200), en el cual se ha ejecutado el procedimiento de esperar llamada, con el fin de recibir llamadas de otros usuarios en el sistema. El usuario B realiza la petición de llamada al usuario A, empleando un terminal de VoIP previamente registrado en la central telefónica de Quito con un número de extensión estática 8000 (socket 172.31.15.4:4230). El usuario B entonces marca el número de extensión personal del usuario A (7000), enviando la petición de llamada a su central local. La central telefónica de Quito al recibir esta petición, empieza a localizar la ubicación de la extensión personal en el dominio *DUNDI*. Al obtener el resultado de la búsqueda se encamina la solicitud de llamada a la central telefónica de Esmeraldas, al llegar la solicitud es procesada por el servidor de redirección el cual informa a la central telefónica de Quito, que debe redirigir la llamada hacia la extensión estática 6000 en el puerto local 5200 IP 192.168.1.4 para llegar así al usuario A.

2.3.3.6 Des-registro de un teléfono IP

El proceso de des-registro, permite desconectar o deshabilitar al teléfono IP de la central telefónica en la cual se encuentra registrado. Una vez que se haya llevado a cabo, la central de telefonía IP no podrá determinar el socket del terminal de VoIP y por lo tanto no podrá alcanzarlo si una llamada entra con dirección hacia la extensión estática des-registrada.

2.3.3.6.1 Procedimiento

El proceso de des-registro consiste en enviar un mensaje SIP del tipo *REGISTER* a la central telefónica en la cual está registrado el teléfono IP. La central telefónica recibe el mensaje procesa la solicitud y envía como respuesta *OK* (código de respuesta 200) al terminal de VoIP, éste al recibir la respuesta da por terminado el proceso.

2.3.3.6.2 Ejemplo de uso

En el ejemplo de uso del literal 2.3.3.1.2 se realizó el registro del teléfono IP identificado con la extensión estática 6000. Tomando como referencia este ejemplo se procede a des-registrar el terminal de VoIP de la central telefónica de Esmeraldas. Desde el cliente se envía la solicitud a la central telefónica con el número de extensión estática.

6000

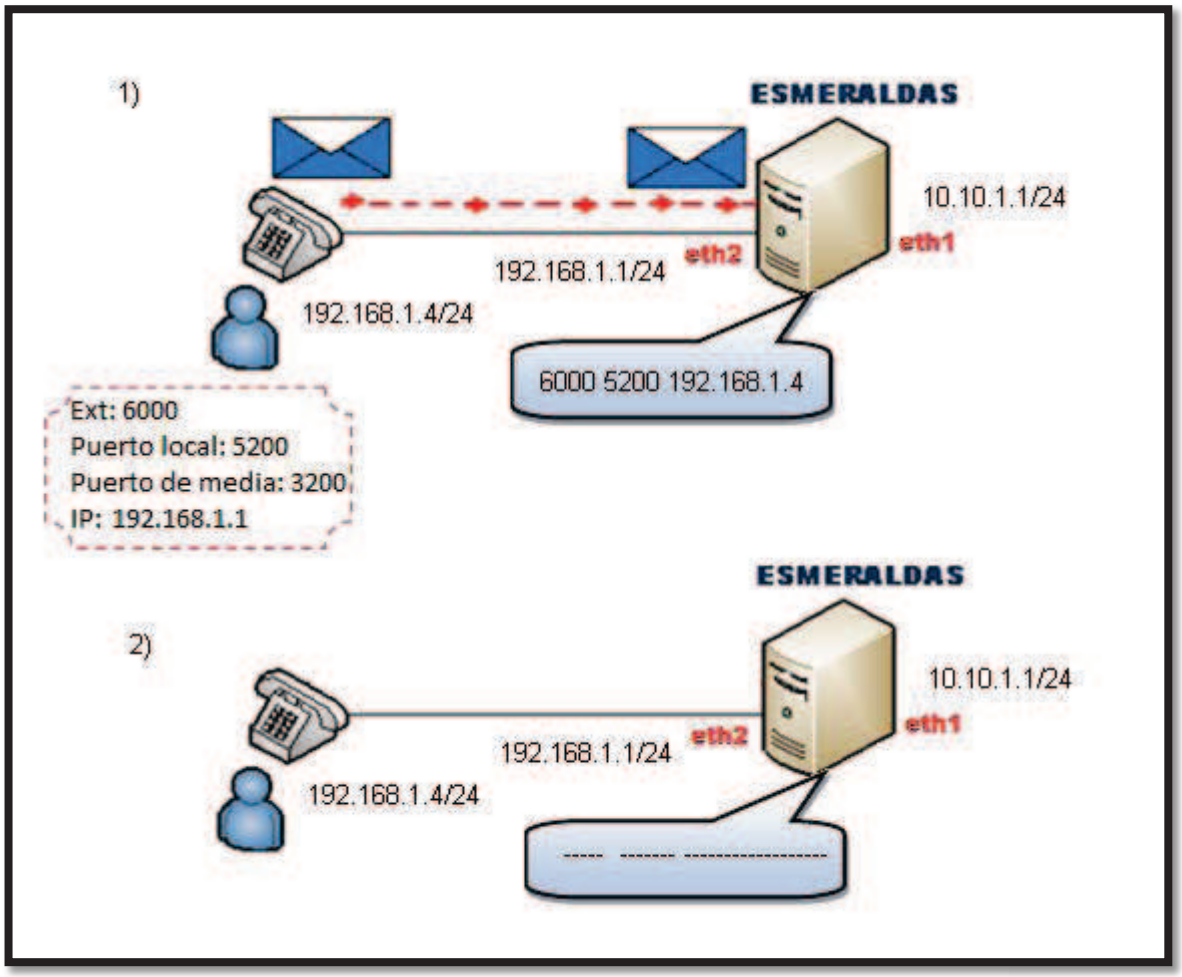


Figura 2.6 Proceso de des-registro de un Telefonía IP

2.3.3.7 Servidor de Redirección

El Servidor de redirección, cumple con la función de redirigir las solicitudes de llamadas de los usuarios desde la central telefónica hacia el terminal de VoIP, con el fin de establecer la comunicación entre dos usuarios. Cuando un

usuario realiza la petición de enrolamiento a la central telefónica, esta además de enrolar la extensión personal en su plan de marcado, inicializa el servidor de redirección en el puerto local que asigna el proceso de enrolamiento.El servidor de redirección por lo tanto, asocia la extensión estática del terminal de VoIP con la extensión personal del usuario.

2.3.3.7.1 Procedimiento

Un usuario al realizar la solicitud de enrolamiento será atendido por el contexto enrolamiento del plan de marcado, el cual después de afiliar a la extensión personal de usuario procede a ejecutar el script para levantar el servidor de redirección en el puerto local asignado por el proceso. Para este efecto se le pasa como argumentos el número de extensión estática del terminal de VoIP, el puerto local y puerto de media asociados al número de extensión personal del usuario. El script abre un socket en el cual espera la llegada de un mensaje *INVITE* dirigido al número de extensión personal. Cuando un usuario en el sistema intenta llamar a dicho número de extensión personal, inicia la búsqueda de la extensión en todas las centrales de telefonía del sistema de telefonía IP, localizada la extensión en determinada central de telefonía ésta envía un mensaje *INVITE* al socket asociado a la extensión, que sería en el cual está escuchando el servidor de redirección. Una vez que el servidor de redirección recibe el *INVITE*, éste responde con un mensaje 302 (movido temporalmente) a la central, indicando la dirección física (socket asociado a la extensión estática del terminal de VoIP) correspondiente al número lógico que está siendo marcado. Finalmente la central, con esta información, envía un *INVITE* al número de extensión estática que ya está registrado y establece la llamada con la instancia (teléfono) que originó la llamada.

En cada central de telefonía habrá varios procesos de redirección, dependiendo de la cantidad de usuarios que se hayan afiliado. Cada uno de estos procesos que están esperando un mensaje de *INVITE* serán identificados mediante el puerto local, desde el cual envían los mensajes.

2.3.3.7.2 Ejemplo de uso

Se toma como referencia el ejemplo de uso del literal 2.3.3.6.3, en el cual se enrola a la extensión personal 7000 ingresando los siguientes datos

7000 4500 3500 10.10.1.1 6000

Ingresada esta información la solicitud es procesada por el contexto enrolamiento del plan de marcado, en forma transparente para el usuario, obteniendo como resultado el enrolamiento de la extensión personal en la central telefónica y el inicio del servidor de redirección.

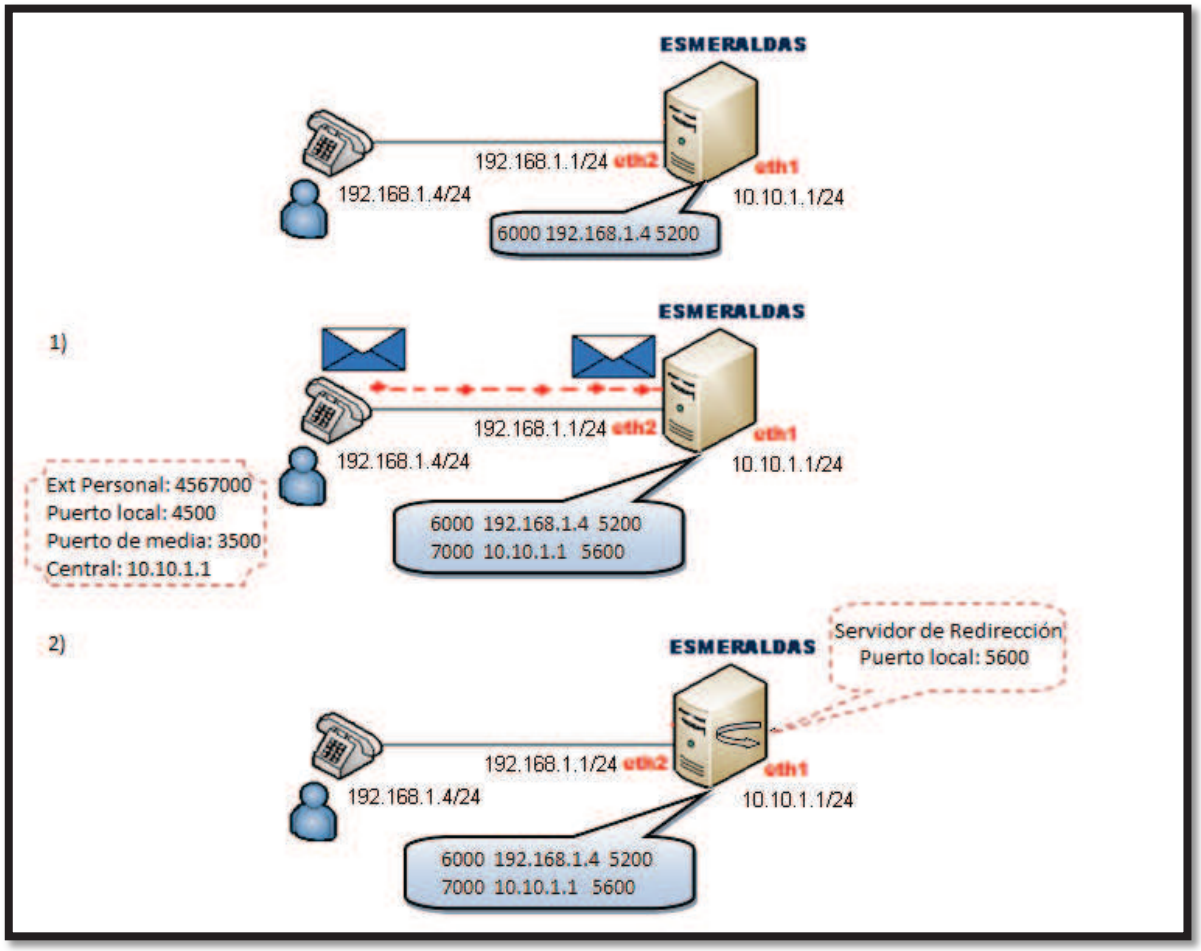


Figura 2.7 Proceso Servidor de Redirección

CAPÍTULO III

IMPLEMENTACIÓN, PRUEBAS Y ANÁLISIS DE LOS RESULTADOS EN EL PROTOTIPO DEL SISTEMA DE PBXs INTERCONECTADAS

3.1 INSTALACIÓN DE LINUX Y *VNUML* EN LA MÁQUINA FÍSICA

3.1.1 INSTALACIÓN DE UBUNTU 9.10

De acuerdo a la herramienta de virtualización seleccionada (*VNUML),*la distribución de Linux que presenta mayor facilidad para su instalación es UBUNTU, debido a que la totalidad de paquetes para instalar *VNUML* están disponibles para esta distribución en el repositorio creado por la *UPM* (Universidad Politécnica de Madrid. Bajo esta consideración se procede a instalar la versión Ubuntu 9.10. Esta distribución de Linux al igual que otras versiones que poseenlicencia GPL (*General Public License*), son fácilmente obtenidas desdeInternet.La fuente de descarga utilizada es:

http://www.ubuntu.com/getubuntu/download

El procedimiento de instalación es sencillo, únicamente se siguen las ayudas gráficas que presenta el programa creando las respectivas particiones que ocupará el sistema operativo. En este caso se crearon las particiones /boot, /swap y /root.

3.1.2 INSTALACIÓN DE *VNUML*

- Abrir un Shell, hacer login como usuario root utilizando el comando sudo su y el password configurado en la instalación.

```
servidor@servidor-desktop:~$ sudo su
```

Figura 3.1Login en Ubuntu

- En el archivo /etc/sources.list, des-comentar todas las líneas que hagan referencia a las fuentes de la comunidad Ubuntu (universe).

- Agregar al archivo de fuentes (/etc/sources.list) la línea que permite la conexión con el repositorio de la UPM para descargar la herramienta *VNUML*.

```
echo "deb http://jungla.dit.upm.es/~vnuml/debian binary/"
>> /etc/apt/sources.list
```

- Hacer un update y un upgrade del sistema. Reiniciar el sistema luego de este proceso.

```
root@servidor-desktop:/home/servidor#apt-get update
root@servidor-desktop:/home/servidor#apt-get upgrade
```

- Instalar el paquete *VNUML* con la herramienta apt-get.

```
root@servidor-desktop:/home/servidor# apt-get installvnuml
```

- Instalar el *kernel* de Linux.

```
root@servidor-desktop:/home/servidor# apt-get install
linux-um
```

- Adicionalmente, aunque no es mandatorio pero si muy recomendable, se puede instalar los siguientes paquetes:

```
root@servidor-desktop:/home/servidor#apt-get install vlan
xterm bridge-utils screen
```

- Para tener acceso remoto vía SSH a las máquinas virtuales habilitadas en ese sentido, es necesario crear una llave DSA (SSH v.2) en la máquina host y que no se solicite una contraseña cada vez que se haga login a las máquinas virtuales. Luego de ejecutar el comando ssh-keygen, aceptar las opciones por defecto (sin agregar passphrase). El path asignado a la llave en el host es: /root/.ssh/id_dsa.pub.

```
root@servidor-desktop:/home/servidor# ssh-keygen -t dsa
```

3.1.3 MANEJO DEL FILESYSTEM

3.1.3.1 Obtención del filesystem

Luego de instalar la herramienta *VNUML* y el *kernel*, se debe instalar o guardar el *filesystem* que se utilizará para las simulaciones. Una forma sencilla es descargarlo mediante el siguiente procedimiento:

```
root@servidor-desktop:/home/servidor# cd
/usr/share/vnuml/filesystems/

root@servidor-desktop:/usr/share/vnuml/filesystems# wget
http://www.dit.upm.es/vnuml/download/scripts/root-fs-installer
perl root-fs-installer
```

Es posible, así también, descargar un *filesystem* de la página del proyecto *UML* (*User Mode Linux*) del que parte el desarrollo de *VNUML*o deeste link http://uml.nagafix.co.uk/, para almacenarlo en el directorio/usr/share/vnuml/filesystems.

Finalmente es recomendable tomar en cuenta la ubicación del archivo (ejecutable) del *kernel* que se utiliza (generalmente llamado Linux en los ejemplos) y del *filesystem* (generalmente llamado root_fs_tutorial), en el momento en el que se haga referencia a ellos desde los archivos *XML* de diseño del entorno virtual.

3.1.3.2 Instalación de Asterisk y *DUNDi*

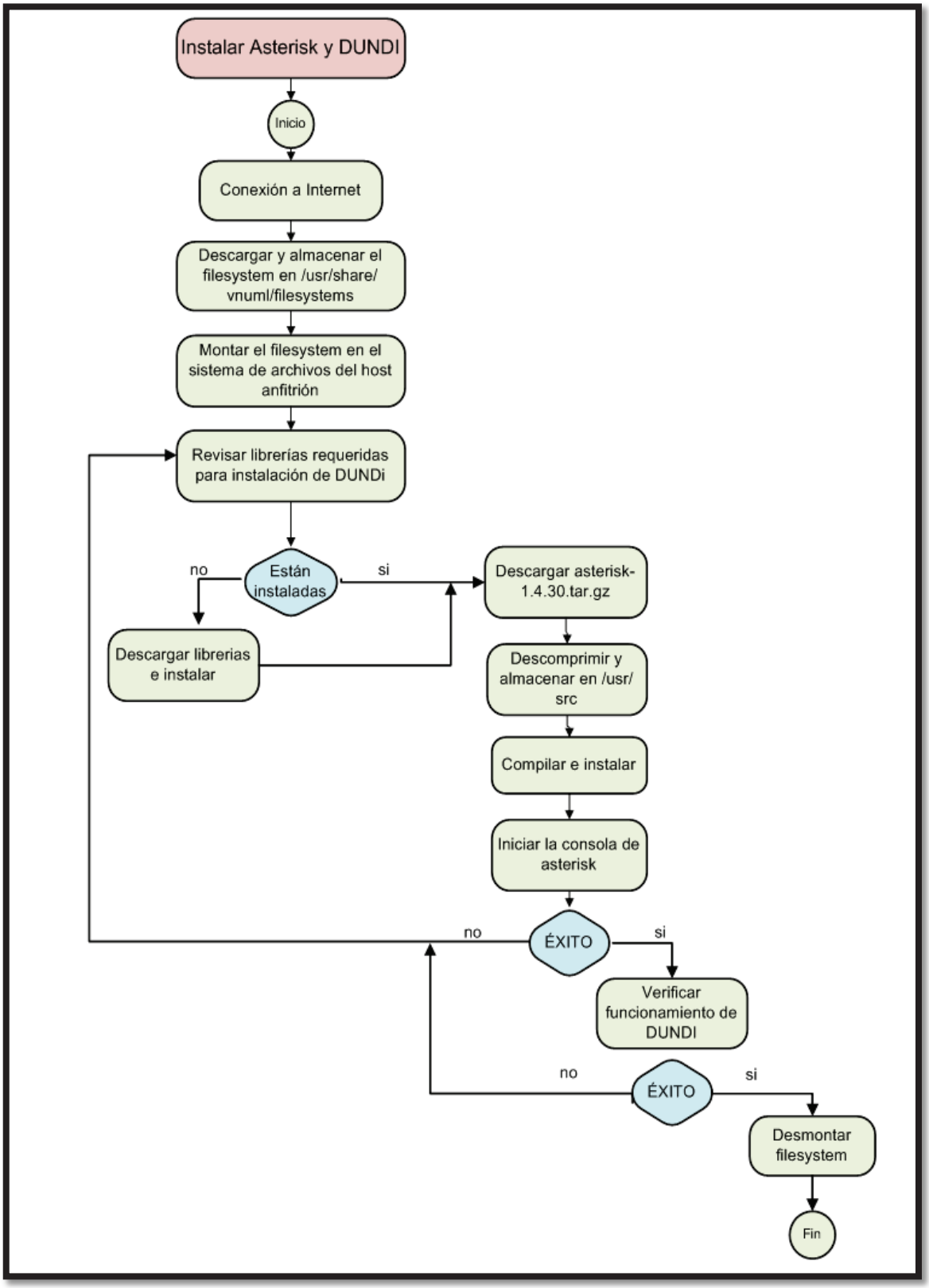


Figura 3.2 Diagrama de flujo de la instalación de Asterisk

- **Montar el *filesystem***

Cuando se requiera instalar software adicional en *VNUML,*se debe realizar el siguiente proceso:

1. Abrir un Shell, hacer login como usuario root.

```
servidor@servidor-desktop:~$ sudo su
```

2. Montar el *filesystem*que se va a utilizar para la simulación del Sistema de Telefonía IP en la máquina física. Los cambios que se realicen sobre este *filesystem* serán permanentes.

```
root@servidor-desktop:~# cd /usr/share/vnuml/filesystems

root@servidor-desktop:/usr/share/vnuml/filesystems# mount -o
loop esquema_red_tesis_1_fs /mnt
```

3. Una vez montado el *filesystem* se procede a instalar el software, en este caso Asterisk y *DUNDi*

Instalación de Asterisk y *DUNDi*

Normalmente, *DUNDi* es un módulo integrado con la herramienta de VoIP Asterisk, siempre y cuando determinados paquetes de los que depende *DUNDi* estén previamente instalados en el sistema operativo. Si no lo están, simplemente *DUNDi* no se instalará junto con Asterisk.

- Los paquetes de los cuales es dependiente la instalación de Asterisk, así como de *DUNDi* son:
    - libncurses5-dev_5.7+20081213-1_i386.deb
    - zlib-1.2.4.tar.gz
    - libssl-dev_0.9.8m-2_i386.deb

- Descargados los paquetes se procede a instalar

```
root@servidor-desktop:/home/servidor#   dpkg -i libncurses5-
dev_5.7+20081213-1_i386.deb

root@servidor-desktop:/home/servidor#  dpkg -i libssl-
dev_0.9.8m-2_i386.deb

root@servidor-desktop:/home/servidor# tar -zxvf zlib-
1.2.4.tar.gz

root@servidor-desktop:/home/servidor#  cd  zlib-1.2.4

root@servidor-desktop:/home/servidor#  make

root@servidor-desktop:/home/servidor#  make install
```

- Una vez confirmada la instalación de estos paquetes, se procede a la instalación de Asterisk. Se copia el paquete asterisk-1.4.30.tar.gz que se ha descargado de la página web de Digium, dentro de la carpeta /usr/src, y se descomprime para obtener el directorio de asterisk.

```
root@servidor-desktop:/home/servidor# cp  asterisk-
1.4.30.tar.gz /usr/src

root@servidor-desktop:/home/servidor# cd  /usr/src

root@servidor-desktop:/usr/src# tar -zxvf  asterisk-
1.4.30.tar.gz

root@servidor-desktop:/usr/src# cd asterisk-1.4.30
```

- Dentro del directorio de asterisk se ejecuta el comando configure que verifica las dependencias de algunos de los módulos a instalarse con Asterisk.

```
root@servidor-desktop:/usr/src/asterisk-1.4.30#./ configure
```

- Luego de ello se ejecuta el comando make menuselect para obtener una interfaz de selección de módulos, en la cual se podrá observar si *DUNDi* está entre los que se instalarán. En el menú se selecciona la opción *PBX* Modules, dentro de la cual estarán listados una serie de módulos (seleccionados con un *) que se instalarán con asterisk. Debe estar seleccionado el módulo *PBX_DUNDi*. Si en lugar del asterisco aparecen unas X, significa que las dependencias para ese módulo no están aún instaladas, impidiendo esto la instalación de *DUNDi*.

```
usr/src/asterisk-1.4.30#make menuselect
```

- Del mismo modo, se debe verificar dentro de la opción número 8 del menú (Resource Modules) que el módulo res_crypto esté seleccionada con un asterisco. Si no lo está y en su lugar aparecen XXX, las dependencias de ese recurso no están instaladas.

- Luego de verificar la instalación de estos módulos, se procede a compilar e instalar el software. En primera instancia se ejecuta el comando make, luego make install, make samples y finalmente make config; siempre dentro del directorio descomprimido.

- Finalmente se procede a verificar la instalación de Asterisk y sus elementos.

```
root@servidor-desktop:/home/servidor# /etc/init.d/asterisk
restart

root@servidor-desktop:/home/servidor# asterisk -rvvvvvvvv
```

Al ejecutar estos comandos se activará el CLI de Asterisk 1.4.30 con lo cual se comprueba la correcta instalación del paquete.

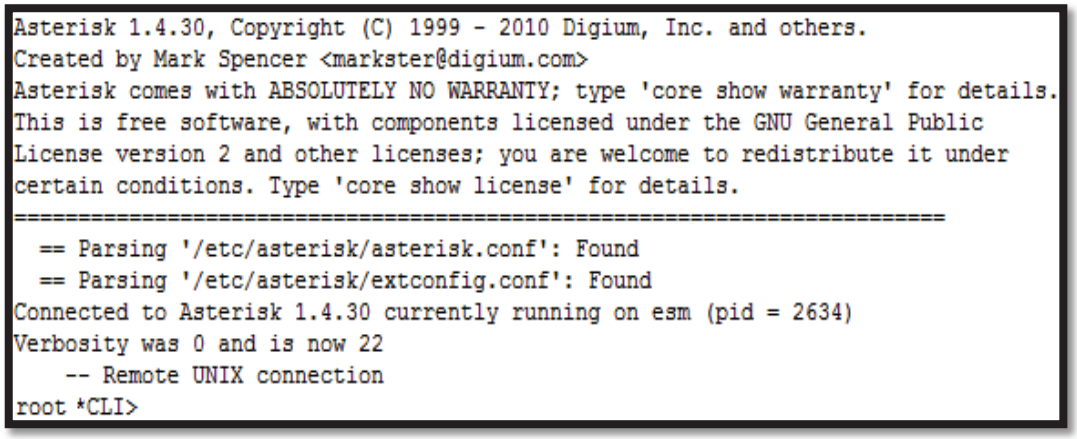


Figura 3.3 Consola de Asterisk 1.4

- Dentro de la nueva consola que se obtiene se ejecuta el comando *DUNDi* show peers. Si no se encuentra el comando hay un problema en la instalación de *DUNDi*. Si se obtiene alguna salida diferente a un error, *DUNDi* se ha instalado correctamente.

```
cli> dundi show peers
```

Desmontar el *Filesystem*

- Terminada la instalación, para cambiar la referencia de directorio raíz a /mnt, se ejecuta el siguiente comando:

```
#chroot /mnt
```

Deacuerdo al diagrama de flujo de la figura 3.1 y a los resultados obtenidos, se verifica la correcta instalación de Asterisk 1.4 y el protocolo *DUNDi* sobre el *filesystem* llamado esquema_red_tesis_1_fs. De este modo se ha preparado en forma adecuada el *filesystem* para ser utilizado en la simulación del esquema de red.

3.2 IMPLEMENTACIÓN Y PRUEBAS DE FUNCIONAMIENTO DEL PROTOTIPO

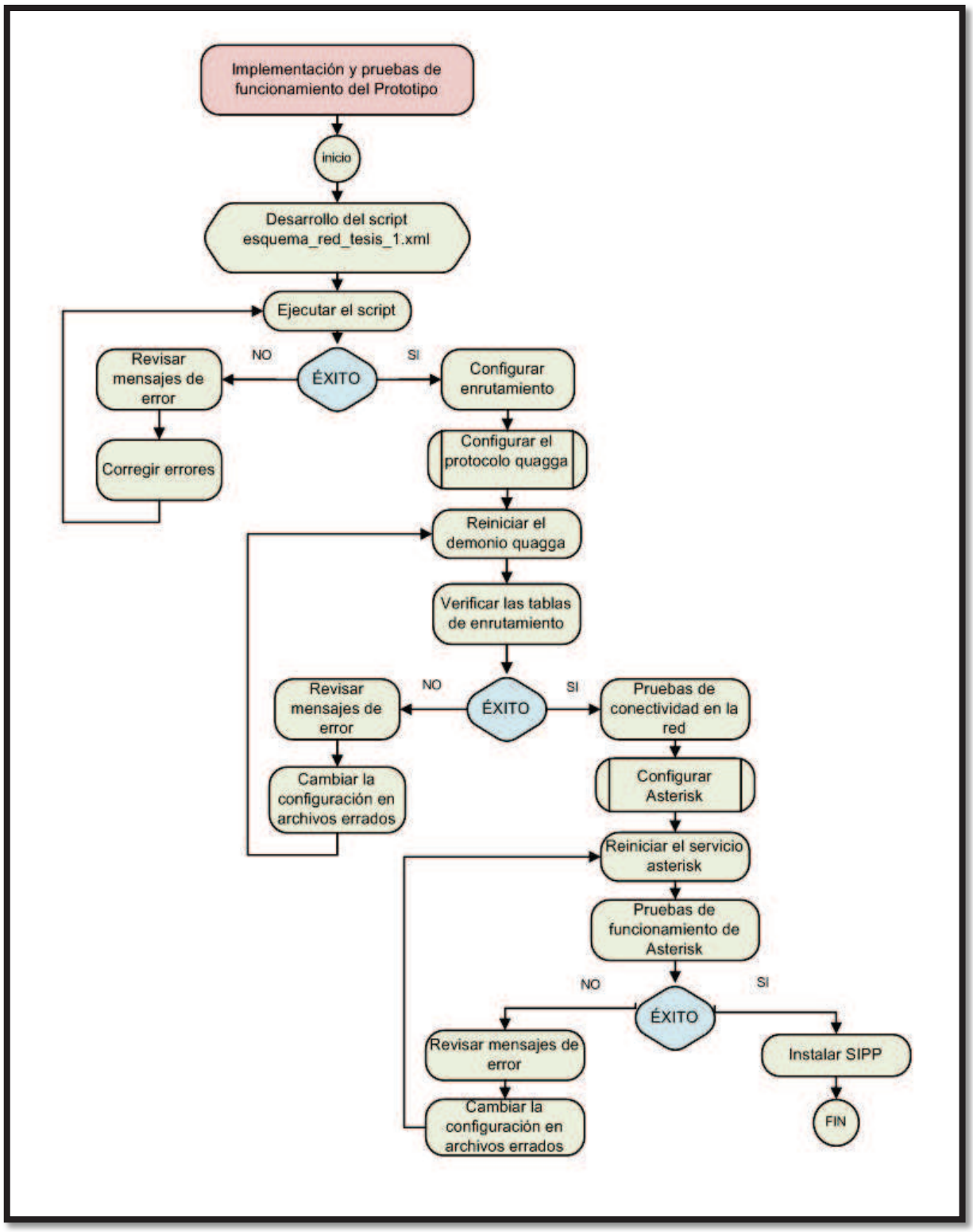


Figura 3.4 Diagrama de flujo de implementación y funcionamiento del prototipo

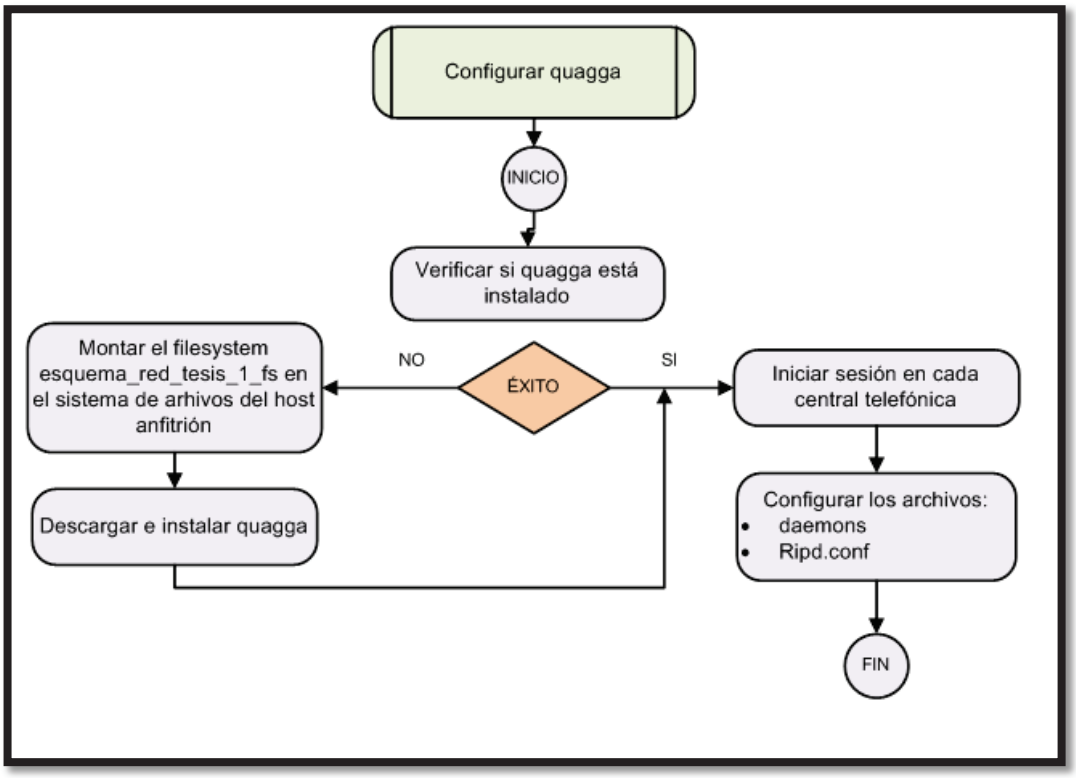


Figura 3.5 Diagrama de flujo de la configuración de quagga

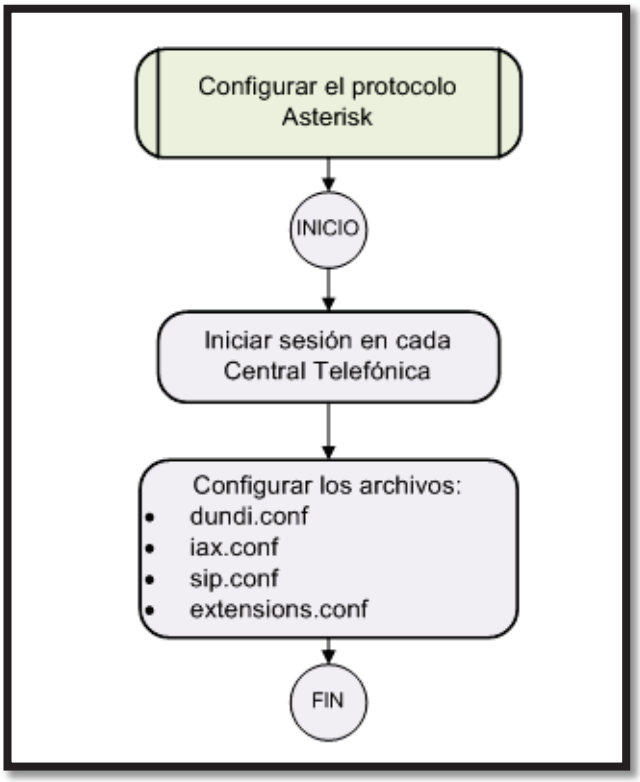


Figura 3.6 Diagrama de flujo de la configuración de asterisk

3.2.1 DESARROLLO DEL SCRIPT. xml esquema_red_tesis_1.xml

3.2.1.1 Análisis de programación

De acuerdo al diagrama de red, presentado en el Capítulo II, se realiza la programación del archivo esquema_red_tesis_1.xml, para que cumpla con los requisitos del diseño de red.

El archivo *XML* empieza con una cabecera en la cual se indica la versión de lenguaje *XML* que se va a utilizar, en este caso 1.0.

```
<?XML version="1.0" encoding="UTF-8"?>
```

La programación del archivo *XML* se basa en el uso de dos tipos de etiquetas:

- **Etiquetas estructurales.-** Se utilizan para definir la estructura del archivo. Estas permiten organizar el contenido del archivo en diferentes secciones, las cuales a la vez contienen otras etiquetas específicas.

  a) **<vnuml>.-**Delimita la descripción de todo el escenario *VNUML* que se va a simular. La sintaxis de esta etiqueta es la siguiente:

```
<vnuml>
        <-->
        .........
        .........
        </-->
        <-->
        .........
        .........
        </-->

</vnuml>
```

  b) **<global>.-**Esta etiqueta agrupa a etiquetas específicas que realizarán las definiciones básicas a partir de las cuales se iniciará el escenario de simulación. Por ejemplo: el sistema de archivo, el *kernel*,

la red de administración entre otros. La sintaxis de esta etiqueta es igual a la de `vnuml`.

A continuación se detallan las etiquetas que agrupa `global`:

- **<version>:** Especifica la versión de lenguaje *VNUML* que se utilizará.

```
<version>1.8 </version>
```

- **<simulation_name>:** Detalla el nombre que se le dará a la simulación. En caso de tener varias simulaciones, cada simulación debe tener un nombre único para evitar que los escenarios se sobrescriban en el sistema produciendo un incorrecto funcionamiento.

```
<simulation_name>esquema_tesis_3</simulation_name>
```

- **<ssh_version>:** Establece la versión de protocolo SSH [14]que se utilizará para conectarse remotamente a las máquinas virtuales, a través de la interfaz de Administración.

```
<ssh_version>2</ssh_version>
```

- **< ssh_key >:** El uso de esta etiqueta es opcional, pero es muy útil ya que elimina la necesidad de autenticarse cada vez que se ingresa remotamente a una máquina virtual. En el literal 3.1.2 con este propósito se creó una llave DSA (SSH v.2) en la máquina host anfitrión, la llave se almacenó en la ubicación: /root/.ssh/id_dsa.pub. Para utilizarla se agrega este path a la etiqueta.

```
<ssh_key>/root/.ssh/id_rsa.pub</ssh_key>
```

– **<automac offset>:** Permite generar en forma automática la dirección MAC para las máquinas virtuales. Al usar **automac offset,** se evita el uso de la etiqueta**<mac>**para definir en forma individual la dirección MAC de cada máquina virtual, permitiendo optimizar el código del escenario de simulación. El parámetro que se debe configurar para este fin es ***offset,*** el cual puede estar en el rango de 0 a 65535.

El formato de la dirección MAC que se genera es:

fe:fd:0:X:Y:0

Dónde:

X: Es el valor del parámetro ***offset*** que se configura en la etiqueta

Y: Es el número de máquina virtual en el orden de creación especificado en el archivo *XML*.

```
<automac offset="0"/>
```

– **<vm_mgmt >:** En esta etiqueta se configura la interfaz de administración para las máquinas virtuales. Se crea entonces la Red de Administración 192.168.0.0 con máscara de 24 bits, la cual permitirá acceder remotamente a las máquinas virtuales.

```
<vm_mgmt type="private" network="192.168.0.0"
mask="24" offset="100">
<host_mapping/>
</vm_mgmt>
```

SSH[14](*Secure Shell*) Es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite a los usuarios conectarse a un host remotamente.

Los parámetros configurados son:

- **type:** Define el tipo de interfaces de red que se crearán, en este caso privado (`private`). Con el valor private, se crea una interfaz para cada máquina virtual del escenario, obteniendo una conexión punto a punto desde el host anfitrión, hacia la máquina virtual. El acceso a las máquinas virtuales será entonces vía SSH con la dirección IP asignada a cada máquina, o con su nombre de host. Las direcciones IP serán configuradas con IPv4.
- **network:** Red de administración 192.168.0.0
- **mask:** Máscara de red de 24 bits, es decir 255.255.255.0
- **offset:** Este parámetro se utiliza con el fin de evitar colisiones de direcciones IP, al tener varias simulaciones simultáneas. Su valor por defecto es 0.

– **<vm_defaults>:** Se configuran los valores por defecto que tendrá todas las máquinas virtuales que se van a crear. Para especificar estos valores se emplean las siguientes etiquetas:

- **<filesystem>:** En esta etiqueta se especifica el sistema de archivos (*filesystem*) que van a utilizar las máquinas virtuales para la simulación. Además se indica que se va a utilizar el tipo de *filesystem COW*, explicado en el Capítulo I.

```
<filesystemtype="cow">/usr/share/vnuml/filesystems/es
quema_tesis_3_fs</filesystem>
```

- **<kernel>:** Se detalla la ubicación (*path*) donde se encuentra el *kernel* que van a usar las máquinas virtuales.

```
    <kernel>/usr/share/vnuml/kernels/linux</kernel>
```

- **<shell>:** Especifica que se utilizará sh para ejecutar los scripts en las máquinas virtuales.

```
<shell>/bin/sh</shell>
```

- **Etiquetas de topología.-** Son utilizadas para configurar los aspectos relacionados a la topología de red.

a) <net>.- Configura las redes virtuales para la simulación del escenario. De acuerdo al esquema de red en el capítulo II se crearán 13 redes virtuales, de las cuales 11 son redes WAN que interconectarán Centrales de Telefonía IP y las dos restantes serán redes LAN ubicadas en diferentes nodos de la red. Los parámetros que se configuran son:

- **name:** Nombre de la red.
- **mode:** Modo de operación. Se define el modo uml_switch para crear una red conmutada.

```
<net name="Net1" mode="uml_switch"/>
```

b) <vm>.- Define la creación de una máquina virtual UML, especificando su nombre (parámetro **name**) y las interfaces de red mediante la etiqueta **if.**

- **<if>** Permite definir la interfaz de red. Los parámetros a configurar son:
  - **id** Identificador de red.
  - **net** Red a la cual pertenece la interfaz de red.
  - **ipv4** dirección IP de la interfaz.

- **mask** máscara de red.

```
<vm name="esm">
<if id="1" net="Net21">
<ipv4 mask="255.255.255.0">10.10.1.1</ipv4>
</if>
<if id="2" net="Net31">
<ipv4 mask="255.255.255.0">192.168.1.1</ipv4>
</if>
    </vm>
```

3.2.1.2 Archivo esquema_red_tesis_1.xml

```
<?XML version="1.0" encoding="UTF-8"?>
<!DOCTYPEVNUML SYSTEM "/usr/share/xml/vnuml/vnuml.dtd" [
<!ENTITYVNUMLDIR "/usr/share/vnuml/">
<!ENTITY REDIR  "&lt;/dev/null &gt;/dev/null 2&gt;&amp;1 &amp;">
]>
<vnuml>
<global>
<version>1.8</version>
<simulation_name>esquema_red_tesis_1</simulation_name>
<ssh_version>2</ssh_version>
<ssh_key>/root/.ssh/id_rsa.pub</ssh_key>
<automac offset="0"/>
<vm_mgmt type="private" network="192.168.0.0" mask="24" offset="100">
<host_mapping/>
</vm_mgmt>
<vm_defaults>
<filesystem
type="cow">/usr/share/vnuml/filesystems/esquema_red_tesis_1_fs</filesystem>
<kernel>/usr/share/vnuml/kernels/linux</kernel>
<shell>/bin/sh</shell>
</vm_defaults>
</global>


<net name="Net1" mode="uml_switch"/>
<net name="Net2" mode="uml_switch"/>
<net name="Net3" mode="uml_switch"/>
<net name="Net4" mode="uml_switch"/>
<net name="Net5" mode="uml_switch"/>
<net name="Net6" mode="uml_switch"/>
<net name="Net7" mode="uml_switch"/>
<net name="Net8" mode="uml_switch"/>
<net name="Net9" mode="uml_switch"/>
<net name="Net10" mode="uml_switch"/>
<net name="Net11" mode="uml_switch"/>
<net name="Net12" mode="uml_switch"/>
<net name="Net13" mode="uml_switch"/>


<vm name="esm">
 <if id="1" net="Net1">
 <ipv4 mask="255.255.255.0">10.10.1.1</ipv4>
 </if>
 <if id="2" net="Net3">
 <ipv4 mask="255.255.255.0">10.10.3.1</ipv4>
 </if>
 <if id="3" net="Net12">
 <ipv4 mask="255.255.255.0">192.168.1.1</ipv4>
 </if>
</vm>
```

```
<vm name="uio">
  <if id="1" net="Net1">
  <ipv4 mask="255.255.255.0">10.10.1.2</ipv4>
  </if>
  <if id="2" net="Net2">
  <ipv4 mask="255.255.255.0">10.10.2.1</ipv4>
  </if>
  <if id="3" net="Net11">
  <ipv4 mask="255.255.255.0">10.10.11.2</ipv4>
  </if>
</vm>

<vm name="lat">
  <if id="1" net="Net2">
  <ipv4 mask="255.255.255.0">10.10.2.2</ipv4>
  </if>
  <if id="2" net="Net4">
  <ipv4 mask="255.255.255.0">10.10.4.2</ipv4>
  </if>
  <if id="3" net="Net5">
  <ipv4 mask="255.255.255.0">10.10.5.1</ipv4>
  </if>
</vm>

<vm name="gye">
  <if id="1" net="Net3">
  <ipv4 mask="255.255.255.0">10.10.3.2</ipv4>
  </if>
  <if id="2" net="Net4">
  <ipv4 mask="255.255.255.0">10.10.4.1</ipv4>
  </if>
  <if id="3" net="Net6">
  <ipv4 mask="255.255.255.0">10.10.6.1</ipv4>
  </if>
  <if id="4" net="Net11">
  <ipv4 mask="255.255.255.0">10.10.11.1</ipv4>
  </if>
</vm>

<vm name="cue">
  <if id="1" net="Net6">
  <ipv4 mask="255.255.255.0">10.10.6.2</ipv4>
  </if>
  <if id="2" net="Net7">
  <ipv4 mask="255.255.255.0">10.10.7.2</ipv4>
  </if>
  <if id="3" net="Net9">
  <ipv4 mask="255.255.255.0">10.10.9.1</ipv4>
  </if>
</vm>

<vm name="amb">
  <if id="1" net="Net5">
  <ipv4 mask="255.255.255.0">10.10.5.2</ipv4>
  </if>
  <if id="2" net="Net7">
  <ipv4 mask="255.255.255.0">10.10.7.1</ipv4>
  </if>
  <if id="3" net="Net8">
  <ipv4 mask="255.255.255.0">10.10.8.1</ipv4>
  </if>
</vm>
```

```
<vm name="mac">
 <if id="1" net="Net8">
 <ipv4 mask="255.255.255.0">10.10.8.2</ipv4>
 </if>
 <if id="2" net="Net10">
 <ipv4 mask="255.255.255.0">10.10.10.1</ipv4>
 </if>
</vm>

<vm name="loj">
 <if id="1" net="Net9">
 <ipv4 mask="255.255.255.0">10.10.9.2</ipv4>
 </if>
 <if id="2" net="Net10">
 <ipv4 mask="255.255.255.0">10.10.10.2</ipv4>
 </if>
 <if id="3" net="Net13">
 <ipv4 mask="255.255.255.0">172.31.15.1</ipv4>
 </if>
</vm>

<vm name="tesm">
 <if id="1" net="Net12">
 <ipv4 mask="255.255.255.0">192.168.1.4</ipv4>
 </if>
 </vm>
 <vm name="tloj">
 <if id="1" net="Net13">
 <ipv4 mask="255.255.255.0">172.31.15.4</ipv4>
 </if>
</vm>

</vnuml>
```

3.2.2 PRUEBAS DE FUNCIONAMIENTO DEL SCRIPT esquema_red_tesis_1.xml

Para realizar las pruebas de funcionamiento del script primero se debe ubicar en el directorio en el cual está almacenado el archivo *XML*.

```
root@servidor-desktop:~# cd /usr/share/vnuml/examples
```

El comando para ejecutar el archivo esquema_red_tesis_1.xml, para probar el funcionamiento del entorno virtual es **vnumlparser.pl:**

```
root@servidor-desktop:/usr/share/vnuml/examples~#
vnumlparser.pl -t esquema_red_tesis_1.xml -v -u root -w 0
```

Donde los parámetros:

-t: Construir la topología.

-v: Modo *Verbose*

-w =0: Sin *timeout*

-u root: Ejecución como usuario root

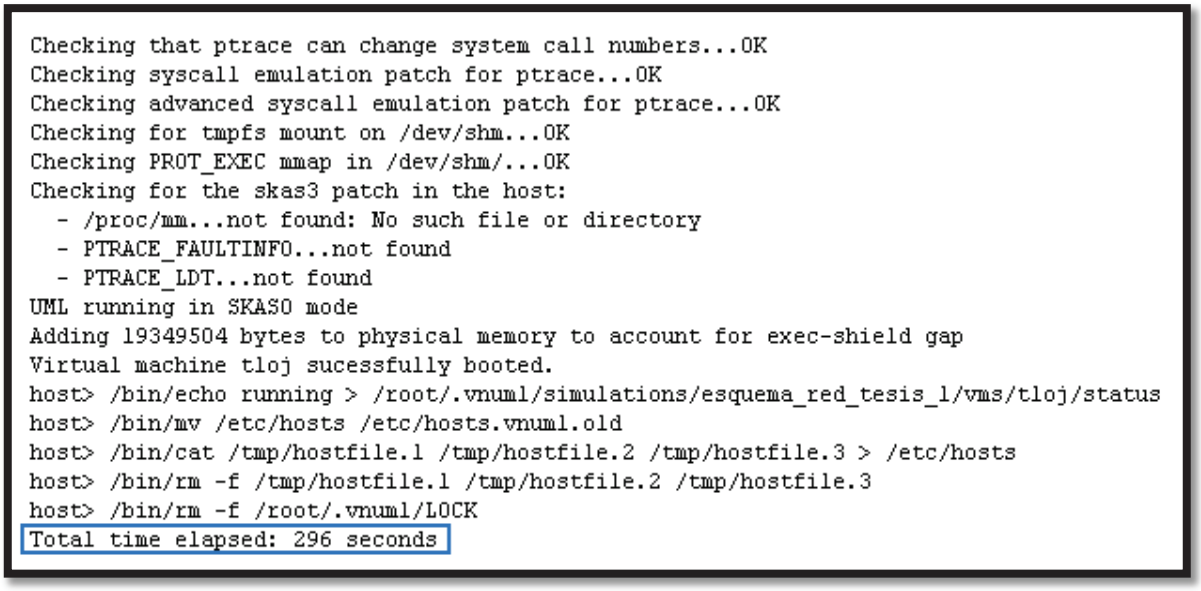


Figura 3.7Ejecución del escenario de simulación

3.2.3 ENRUTAMIENTO DEL PROTOTIPO DEL SISTEMA DE TELEFONÍA IP

Para realizar el enrutamiento en el Sistema de Telefonía IP, se utilizará el software quagga

El enrutamiento para interconectar las Centrales de Telefonía IP, será realizado con el protocolo RIP. Al instalar quagga existe la posibilidad de configurar el enrutamiento con OSPF o RIP. Con el fin de disminuir el consumo innecesario de recursos del CPU, se selecciona el protocolo RIP, el cual ayuda a cumplir este propósito.

CARACTERISTICAS	RIP	OSPF
Tipo	Vector- Distancia	Estado de enlace
Tiempo de convergencia	Lento	Rápido
Consumo de recursos	Bajo	Alto Más memoria y potencia de procesamiento.
Licencia	Libre uso	Libre uso

Tabla 3.1 Comparación RIP vs OSPF

3.2.3.1 Instalación de quagga

Quagga es un software libre de enrutamiento avanzado, el cual permite usar la familia de sistemas operativos Unix como enrutadores. Está diseñado especialmente para Solaris y Linux. Da soporte para IPv4 e IPv6. Zebra es el corazónpara el funcionamiento de Quagga. Zebra se encarga de administrarel ruteo IP, proporcionando las actualizaciones a las tablas de ruteo del *kernel*, lookups en la interface y redistribución de rutas.

Quagga permite configurar todos protocolos de enrutamiento basados en TCP/IP:

- RIP v1/v2
- OSPF v2/v3
- BGP v4

Para instalar quagga se siguen los siguientes pasos:

1. Abrir un Shell, hacer login como usuario root.

```
servidor@servidor-desktop:~$ sudo su
```

2. Obtener el software desde el internet e instalar mediante el comando apt-get install.

```
root@servidor-desktop:~# apt-get install quagga
```

3. Copiar los archivos zebra.conf.sample y ripd.conf.sample instalados en el directorio /etc/quagga y cambiar de nombre a zebra.conf y ripd.conf respectivamente, para su posterior configuración.

```
~# cd/usr/share/doc/quagga/examples/

~# cp  zebra.conf.sample /etc/quagga/

~#cp  ripd.conf.sample /etc/quagga/

~#cd/etc/quagga/

~# mv zebra.conf.sample zebra.conf

~# mv ripd.conf.sample ripd.conf
```

3.2.3.2 Edición del archivo daemons

Una vez instalado *quagga*, para configurar el enrutamiento se deben editar los archivos en función del protocolo de enrutamiento seleccionado, en este caso RIP. En el archivo *daemons,* se habilitan con yes las opciones *ripd y zebra,*con el fin de arrancar los demonios*zebra* y *ripd*al iniciar quagga. La configuración de este archivo es la misma para todos los ocho nodos que integran la red.

```
# This file tells the quagga package which daemons to
start.
#
# Entries are in the format: <daemon>=(yes|no|priority)
#   0, "no"  = disabled
#   1, "yes" = highest priority
#   2 ..10  = lower priorities
# Read /usr/share/doc/quagga/README.Debian for details.
#
# Sample configurations for these daemons can be found in
# /usr/share/doc/quagga/examples/.
#
# ATTENTION:
#
# When activation a daemon at the first time, a config
file, even if it is
# empty, has to be present *and* be owned by the user and
group "quagga", else
# the daemon will not be started by /etc/init.d/quagga.
The permissions should
# be u=rw,g=r,o=.
# When using "vtysh" such a config file is also needed. It
should be owned by
```

```
# group "quaggavty" and set to ug=rw,o= though. Check
/etc/pam.d/quagga, too.
#
zebra=yes
bgpd=no
ospfd=no
ospf6d=no
ripd=yes
ripngd=no
isisd=no
```

3.2.3.3 Edición del archivo ripd.conf

El archivo ripd.conf establece la configuración del demonio ripd para cada nodo de la red. En este archivo se especifican las redes que se comunicarán con el router (nodo) a través de sus interfaces de red. La configuración de estos archivos por lo tanto es distinta para cada nodo de la red. Como ejemplo se detalla la configuración del nodo Esmeraldas, las configuraciones del archivo ripd.conf para los otros siete nodos de la red se detalla en el ANEXO I.

- ***Nodo Esmeraldas (ESM)***

  En este archivo se resalta la configuración de la red adyacente a la central telefónica de Esmeraldas, en este caso network: 10.10.1.0/24. Esta zona es la única que cambia en la configuración ripd.conf de cada central telefónica.

```
! -*- rip -*-
!
! RIPd sample configuration file
!
! $Id: ripd.conf.sample,v 1.1 2002/12/13 20:15:30 paul Exp
$
!
hostname ripd
password zebra
!
! debug rip events
! debug rip packet
!
```

```
router rip

network 10.10.1.0/24

network 10.10.3.0/24

! network eth0

! route 10.0.0.0/8

! distribute-list private-only in eth0

!access-list private-only permit 10.0.0.0/8

!access-list private-only deny any

log file /var/log/quagga/ripd.log

log stdout
```

3.2.3.4 Inicialización del protocolo

Para inicializar el enrutamiento en el sistema de telefonía IP inicializamos *quagga* en cada nodo de la red.

```
nodo:/etc/quagga# etc/init.d/quagga restart
```

Para facilitar la inicialización de *quagga*en cada nodo de la red, se crea un scriptllamado script_quagga.sh, que inicializará el servicio en cada nodo desde la máquina anfitrión. La localización de este archivo es: /usr/share/vnuml/examples/script_quagga.sh.

Script:

```
ssh esm /etc/init.d/quagga restart
ssh uio /etc/init.d/quagga restart
ssh gye /etc/init.d/quagga restart
ssh amb /etc/init.d/quagga restart
ssh lat /etc/init.d/quagga restart
ssh cue /etc/init.d/quagga restart
ssh mac /etc/init.d/quagga restart
ssh loj /etc/init.d/quagga restart
```

Para ejecutar el script se utiliza el comando sh:

```
root@servidor-desktop:/usr/share/vnuml/examples~#sh
script_quagga.sh
```

```
root@servidor-desktop:/usr/share/vnuml/examples# vim script_quagga.sh
root@servidor-desktop:/usr/share/vnuml/examples# sh script_quagga.sh
Stopping Quagga daemons (prio:0): (ripd) (zebra) (bgpd) (ripngd) (ospfd) (ospf6d) (isisd).
Removing all routes made by zebra.
Starting Quagga daemons (prio:10): zebra ripd.
Stopping Quagga daemons (prio:0): (ripd) (zebra) (bgpd) (ripngd) (ospfd) (ospf6d) (isisd).
Removing all routes made by zebra.
Starting Quagga daemons (prio:10): zebra ripd.
Stopping Quagga daemons (prio:0): (ripd) (zebra) (bgpd) (ripngd) (ospfd) (ospf6d) (isisd).
Removing all routes made by zebra.
Starting Quagga daemons (prio:10): zebra ripd.
Stopping Quagga daemons (prio:0): (ripd) (zebra) (bgpd) (ripngd) (ospfd) (ospf6d) (isisd).
Removing all routes made by zebra.
Starting Quagga daemons (prio:10): zebra ripd.
Stopping Quagga daemons (prio:0): (ripd) (zebra) (bgpd) (ripngd) (ospfd) (ospf6d) (isisd).
Removing all routes made by zebra.
Starting Quagga daemons (prio:10): zebra ripd.
Stopping Quagga daemons (prio:0): (ripd) (zebra) (bgpd) (ripngd) (ospfd) (ospf6d) (isisd).
Removing all routes made by zebra.
Starting Quagga daemons (prio:10): zebra ripd.
Stopping Quagga daemons (prio:0): (ripd) (zebra) (bgpd) (ripngd) (ospfd) (ospf6d) (isisd).
Removing all routes made by zebra.
Starting Quagga daemons (prio:10): zebra ripd.
Stopping Quagga daemons (prio:0): (ripd) (zebra) (bgpd) (ripngd) (ospfd) (ospf6d) (isisd).
Removing all routes made by zebra.
Starting Quagga daemons (prio:10): zebra ripd.
root@servidor-desktop:/usr/share/vnuml/examples#
```

Figura 3.8 Inicialización de quagga en el escenario de simulación

3.2.4 PRUEBAS DE ENRUTAMIENTO Y CONECTIVIDAD ENTRE CENTRALES DE TELEFONÍA IP

3.2.4.1 Verificación de tabla de rutas en las centrales de telefonía IP

Para verificar la tabla de rutas en cada central de telefonía IP se emplea el comando route con la opción -n, para desplegar todas las rutas reconocidas.

```
~#route -n
```

- **Esmeraldas**

```
esm:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.100   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       10.10.3.2       255.255.255.0   UG    2      0        0 eth2
10.10.7.0       10.10.3.2       255.255.255.0   UG    3      0        0 eth2
10.10.4.0       10.10.3.2       255.255.255.0   UG    2      0        0 eth2
10.10.5.0       10.10.3.2       255.255.255.0   UG    3      0        0 eth2
10.10.2.0       10.10.1.2       255.255.255.0   UG    2      0        0 eth1
10.10.3.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.1.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.10.0      10.10.3.2       255.255.255.0   UG    4      0        0 eth2
10.10.11.0      10.10.1.2       255.255.255.0   UG    2      0        0 eth1
10.10.8.0       10.10.3.2       255.255.255.0   UG    4      0        0 eth2
10.10.9.0       10.10.3.2       255.255.255.0   UG    3      0        0 eth2
```

Figura 3.9 Tabla de rutas de la central telefónica Esmeraldas

- **Quito**

```
uio:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.104   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       10.10.11.1      255.255.255.0   UG    2      0        0 eth3
10.10.7.0       10.10.2.2       255.255.255.0   UG    3      0        0 eth2
10.10.4.0       10.10.11.1      255.255.255.0   UG    2      0        0 eth3
10.10.5.0       10.10.2.2       255.255.255.0   UG    2      0        0 eth2
10.10.2.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.3.0       10.10.1.1       255.255.255.0   UG    2      0        0 eth1
10.10.1.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.10.0      10.10.11.1      255.255.255.0   UG    4      0        0 eth3
10.10.11.0      0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.8.0       10.10.2.2       255.255.255.0   UG    3      0        0 eth2
10.10.9.0       10.10.11.1      255.255.255.0   UG    3      0        0 eth3
```

Figura 3.10 Tabla de rutas de la central telefónica Quito

- **Latacunga**

```
lat:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.108   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       10.10.4.1       255.255.255.0   UG    2      0        0 eth2
10.10.7.0       10.10.5.2       255.255.255.0   UG    2      0        0 eth3
10.10.4.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.5.0       0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.2.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.3.0       10.10.4.1       255.255.255.0   UG    2      0        0 eth2
10.10.1.0       10.10.2.1       255.255.255.0   UG    2      0        0 eth1
10.10.10.0      10.10.5.2       255.255.255.0   UG    3      0        0 eth3
10.10.11.0      10.10.4.1       255.255.255.0   UG    2      0        0 eth2
10.10.8.0       10.10.5.2       255.255.255.0   UG    2      0        0 eth3
10.10.9.0       10.10.4.1       255.255.255.0   UG    3      0        0 eth2
```

Figura 3.11 Tabla de rutas de la central telefónica Latacunga

- **Guayaquil**

```
gye:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.112   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.7.0       10.10.6.2       255.255.255.0   UG    2      0        0 eth3
10.10.4.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.5.0       10.10.4.2       255.255.255.0   UG    2      0        0 eth2
10.10.2.0       10.10.11.2      255.255.255.0   UG    2      0        0 eth4
10.10.3.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.1.0       10.10.11.2      255.255.255.0   UG    2      0        0 eth4
10.10.10.0      10.10.6.2       255.255.255.0   UG    3      0        0 eth3
10.10.11.0      0.0.0.0         255.255.255.0   U     0      0        0 eth4
10.10.8.0       10.10.4.2       255.255.255.0   UG    3      0        0 eth2
10.10.9.0       10.10.6.2       255.255.255.0   UG    2      0        0 eth3
```

Figura 3.12 Tabla de rutas de la central telefónica Guayaquil

- **Ambato**

```
amb:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.120   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       10.10.7.2       255.255.255.0   UG    2      0        0 eth2
10.10.7.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.4.0       10.10.5.1       255.255.255.0   UG    2      0        0 eth1
10.10.5.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.2.0       10.10.5.1       255.255.255.0   UG    2      0        0 eth1
10.10.3.0       10.10.5.1       255.255.255.0   UG    3      0        0 eth1
10.10.1.0       10.10.5.1       255.255.255.0   UG    3      0        0 eth1
10.10.10.0      10.10.8.2       255.255.255.0   UG    2      0        0 eth3
10.10.11.0      10.10.5.1       255.255.255.0   UG    3      0        0 eth1
10.10.8.0       0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.9.0       10.10.7.2       255.255.255.0   UG    2      0        0 eth2
```

Figura 3.13 Tabla de rutas de la central telefónica Ambato

- **Cuenca**

```
cue:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.116   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.7.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.4.0       10.10.6.1       255.255.255.0   UG    2      0        0 eth1
10.10.5.0       10.10.7.1       255.255.255.0   UG    2      0        0 eth2
10.10.2.0       10.10.6.1       255.255.255.0   UG    3      0        0 eth1
10.10.3.0       10.10.6.1       255.255.255.0   UG    2      0        0 eth1
10.10.1.0       10.10.6.1       255.255.255.0   UG    3      0        0 eth1
10.10.10.0      10.10.9.2       255.255.255.0   UG    2      0        0 eth3
10.10.11.0      10.10.6.1       255.255.255.0   UG    2      0        0 eth1
10.10.8.0       10.10.7.1       255.255.255.0   UG    2      0        0 eth2
10.10.9.0       0.0.0.0         255.255.255.0   U     0      0        0 eth3
```

Figura 3.14 Tabla de rutas de la central telefónica Cuenca

- **Macas**

```
mac:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.124   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       10.10.8.1       255.255.255.0   UG    3      0        0 eth1
10.10.7.0       10.10.8.1       255.255.255.0   UG    2      0        0 eth1
10.10.4.0       10.10.8.1       255.255.255.0   UG    3      0        0 eth1
10.10.5.0       10.10.8.1       255.255.255.0   UG    2      0        0 eth1
10.10.2.0       10.10.8.1       255.255.255.0   UG    3      0        0 eth1
10.10.3.0       10.10.8.1       255.255.255.0   UG    4      0        0 eth1
10.10.1.0       10.10.8.1       255.255.255.0   UG    4      0        0 eth1
10.10.10.0      0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.11.0      10.10.8.1       255.255.255.0   UG    4      0        0 eth1
10.10.8.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.9.0       10.10.10.2      255.255.255.0   UG    2      0        0 eth2
```

Figura 3.15 Tabla de rutas de la central telefónica Macas

- **Loja**

```
loj:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.128   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       10.10.9.1       255.255.255.0   UG    2      0        0 eth1
10.10.7.0       10.10.9.1       255.255.255.0   UG    2      0        0 eth1
10.10.4.0       10.10.9.1       255.255.255.0   UG    3      0        0 eth1
172.31.15.0     0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.5.0       10.10.9.1       255.255.255.0   UG    3      0        0 eth1
10.10.2.0       10.10.9.1       255.255.255.0   UG    4      0        0 eth1
10.10.3.0       10.10.9.1       255.255.255.0   UG    3      0        0 eth1
10.10.1.0       10.10.9.1       255.255.255.0   UG    4      0        0 eth1
10.10.10.0      0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.11.0      10.10.9.1       255.255.255.0   UG    3      0        0 eth1
10.10.8.0       10.10.10.1      255.255.255.0   UG    2      0        0 eth2
10.10.9.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
```

Figura 3.16 Tabla de rutas de la central telefónica Loja

3.2.5 VERIFICACIÓN DE LA CONECTIVIDAD ENTRE LAS CENTRALES DE TELEFONÍA IP

Comprobadas la correcta configuración de las rutas en cada central, se procede a verificar la conectividad usando el comando *ping* entre las interfaces de red.

A continuación se verifica la conectividad de las centrales telefónicas de Esmeraldas y Loja. En el anexo II se muestra la conectividad de las restantes seis centrales telefónicas.

```
~#ping direccion IP (0.0.0.0)
```

3.2.5.1 Esmeraldas

- Interfaces de red de la central telefónica

Para verificar la correcta creación de las interfaces de red de acuerdo al esquema de red planteado, se ejecuta el comando *ifconfig*.

```
~#ifconfig
```

```
esm:~# ifconfig
eth0      Link encap:Ethernet  HWaddr fe:fd:00:00:01:00
          inet addr:192.168.0.102  Bcast:192.168.0.103  Mask:255.255.255.252
          inet6 addr: fe80::fcfd:ff:fe00:100/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:378 errors:0 dropped:0 overruns:0 frame:0
          TX packets:349 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:45028 (43.9 KiB)  TX bytes:53486 (52.2 KiB)
          Interrupt:5

eth1      Link encap:Ethernet  HWaddr fe:fd:00:00:01:01
          inet addr:10.10.1.1  Bcast:10.10.1.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:ff:fe00:101/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:64 errors:0 dropped:0 overruns:0 frame:0
          TX packets:69 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:10228 (9.9 KiB)  TX bytes:10782 (10.5 KiB)
          Interrupt:5

eth2      Link encap:Ethernet  HWaddr fe:fd:00:00:01:02
          inet addr:10.10.3.1  Bcast:10.10.3.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:ff:fe00:102/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:64 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:11388 (11.1 KiB)  TX bytes:6098 (5.9 KiB)
          Interrupt:5

eth3      Link encap:Ethernet  HWaddr fe:fd:00:00:01:03
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:ff:fe00:103/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:384 (384.0 B)  TX bytes:468 (468.0 B)
          Interrupt:5

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:20 errors:0 dropped:0 overruns:0 frame:0
          TX packets:20 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1324 (1.2 KiB)  TX bytes:1324 (1.2 KiB)
```

Figura 3.17 Interfaces de red central telefónica Esmeraldas

- Conectividad
  - Cliente TESM

```
esm:~# ping 192.168.1.4
PING 192.168.1.4 (192.168.1.4) 56(84) bytes of data.
64 bytes from 192.168.1.4: icmp_seq=1 ttl=64 time=10.5 ms
64 bytes from 192.168.1.4: icmp_seq=2 ttl=64 time=0.148 ms
64 bytes from 192.168.1.4: icmp_seq=3 ttl=64 time=0.174 ms
```

Figura 3.18Conectividad ESM-TESM

  - CentralUIO

```
esm:~# ping 10.10.1.2
PING 10.10.1.2 (10.10.1.2) 56(84) bytes of data.
64 bytes from 10.10.1.2: icmp_seq=1 ttl=64 time=10.6 ms
64 bytes from 10.10.1.2: icmp_seq=2 ttl=64 time=0.182 ms
64 bytes from 10.10.1.2: icmp_seq=3 ttl=64 time=0.181 ms
```

Figura 3.19Conectividad ESM-UIO

  - CentralLAT

```
esm:~# ping 10.10.2.2
PING 10.10.2.2 (10.10.2.2) 56(84) bytes of data.
64 bytes from 10.10.2.2: icmp_seq=1 ttl=63 time=10.5 ms
64 bytes from 10.10.2.2: icmp_seq=2 ttl=63 time=22.8 ms
64 bytes from 10.10.2.2: icmp_seq=3 ttl=63 time=0.295 ms
```

Figura 3.20Conectividad ESM-LAT

  - Central GYE

```
esm:~# ping 10.10.3.2
PING 10.10.3.2 (10.10.3.2) 56(84) bytes of data.
64 bytes from 10.10.3.2: icmp_seq=1 ttl=63 time=17.1 ms
64 bytes from 10.10.3.2: icmp_seq=2 ttl=63 time=10.3 ms
64 bytes from 10.10.3.2: icmp_seq=3 ttl=63 time=4.21 ms
```

Figura 3.21Conectividad ESM-GYE

– Central AMB

```
esm:~# ping 10.10.5.2
PING 10.10.5.2 (10.10.5.2) 56(84) bytes of data.
64 bytes from 10.10.5.2: icmp_seq=1 ttl=62 time=22.0 ms
64 bytes from 10.10.5.2: icmp_seq=2 ttl=62 time=4.11 ms
64 bytes from 10.10.5.2: icmp_seq=3 ttl=62 time=10.4 ms
```

Figura 3.22Conectividad ESM-AMB

– Central GYE

```
esm:~# ping 10.10.3.2
PING 10.10.3.2 (10.10.3.2) 56(84) bytes of data.
64 bytes from 10.10.3.2: icmp_seq=1 ttl=63 time=17.1 ms
64 bytes from 10.10.3.2: icmp_seq=2 ttl=63 time=10.3 ms
64 bytes from 10.10.3.2: icmp_seq=3 ttl=63 time=4.21 ms
```

Figura 3.23Conectividad ESM-GYE

– Central CUE

```
esm:~# ping 10.10.6.2
PING 10.10.6.2 (10.10.6.2) 56(84) bytes of data.
64 bytes from 10.10.6.2: icmp_seq=1 ttl=62 time=10.8 ms
64 bytes from 10.10.6.2: icmp_seq=2 ttl=62 time=0.410 ms
64 bytes from 10.10.6.2: icmp_seq=3 ttl=62 time=4.12 ms
```

Figura 3.24Conectividad ESM-CUE

– Central MAC

```
esm:~# ping 10.10.8.2
PING 10.10.8.2 (10.10.8.2) 56(84) bytes of data.
64 bytes from 10.10.8.2: icmp_seq=1 ttl=61 time=10.8 ms
64 bytes from 10.10.8.2: icmp_seq=2 ttl=61 time=4.38 ms
64 bytes from 10.10.8.2: icmp_seq=3 ttl=61 time=20.3 ms
```

Figura 3.25Conectividad ESM-MAC

– Central loj

```
esm:~# ping 10.10.9.2
PING 10.10.9.2 (10.10.9.2) 56(84) bytes of data.
64 bytes from 10.10.9.2: icmp_seq=1 ttl=61 time=73.6 ms
64 bytes from 10.10.9.2: icmp_seq=2 ttl=61 time=0.526 ms
64 bytes from 10.10.9.2: icmp_seq=3 ttl=61 time=4.29 ms
```

Figura 3.26Conectividad ESM-LOJ

3.2.5.2 Loja

- Interfaces de red de la central telefónica

```
~#ifconfig
```

```
loj:~# ifconfig
eth0      Link encap:Ethernet  HWaddr fe:fd:00:00:08:00
          inet addr:192.168.0.130  Bcast:192.168.0.131  Mask:255.255.255.252
          inet6 addr: fe80::fcfd:ff:fe00:800/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:198 errors:0 dropped:0 overruns:0 frame:0
          TX packets:160 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:23068 (22.5 KiB)  TX bytes:26524 (25.9 KiB)
          Interrupt:5

eth1      Link encap:Ethernet  HWaddr fe:fd:00:00:08:01
          inet addr:10.10.9.2  Bcast:10.10.9.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:ff:fe00:801/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:222 errors:0 dropped:0 overruns:0 frame:0
          TX packets:220 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:29568 (28.8 KiB)  TX bytes:18828 (18.3 KiB)
          Interrupt:5

eth2      Link encap:Ethernet  HWaddr fe:fd:00:00:08:02
          inet addr:10.10.10.2  Bcast:10.10.10.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:ff:fe00:802/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:187 errors:0 dropped:0 overruns:0 frame:0
          collisions:0 txqueuelen:1000
          RX bytes:26396 (25.7 KiB)  TX bytes:28292 (27.6 KiB)
          Interrupt:5

eth3      Link encap:Ethernet  HWaddr fe:fd:00:00:08:03
          inet addr:172.31.15.1  Bcast:172.31.15.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:ff:fe00:803/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:384 (384.0 B)  TX bytes:468 (468.0 B)
          Interrupt:5
```

```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:14 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:940 (940.0 B)  TX bytes:940 (940.0 B)
```

Figura 3.27Interfaces de red central telefónica Loja

- Conectividad
  - Central ESM

```
loj:~# ping 10.10.1.1
PING 10.10.1.1 (10.10.1.1) 56(84) bytes of data.
64 bytes from 10.10.1.1: icmp_seq=1 ttl=61 time=10.8 ms
64 bytes from 10.10.1.1: icmp_seq=2 ttl=61 time=0.545 ms
64 bytes from 10.10.1.1: icmp_seq=3 ttl=61 time=0.588 ms
```

Figura 3.28ConectividadLOJ-ESM

  - Central UIO

```
loj:~# ping 10.10.2.1
PING 10.10.2.1 (10.10.2.1) 56(84) bytes of data.
64 bytes from 10.10.2.1: icmp_seq=1 ttl=62 time=3.84 ms
64 bytes from 10.10.2.1: icmp_seq=2 ttl=62 time=2.53 ms
```

Figura 3.29 ConectividadLOJ-UIO

  - Central LAT

```
loj:~# ping 10.10.4.2
PING 10.10.4.2 (10.10.4.2) 56(84) bytes of data.
64 bytes from 10.10.4.2: icmp_seq=1 ttl=62 time=26.4 ms
64 bytes from 10.10.4.2: icmp_seq=2 ttl=62 time=3.24 ms
```

Figura 3.30 ConectividadLOJ-LAT

– Central GYE

```
loj:~# ping 10.10.6.1
PING 10.10.6.1 (10.10.6.1) 56(84) bytes of data.
64 bytes from 10.10.6.1: icmp_seq=1 ttl=63 time=1.86 ms
64 bytes from 10.10.6.1: icmp_seq=2 ttl=63 time=0.292 ms
```

Figura 3.31 Conectividad LOJ-GYE

– Central AMB

```
loj:~# ping 10.10.8.1
PING 10.10.8.1 (10.10.8.1) 56(84) bytes of data.
64 bytes from 10.10.8.1: icmp_seq=1 ttl=63 time=22.6 ms
64 bytes from 10.10.8.1: icmp_seq=2 ttl=63 time=0.323 ms
```

Figura 3.32 Conectividad LOJ-AMB

– Central CUE

```
loj:~# ping 10.10.9.1
PING 10.10.9.1 (10.10.9.1) 56(84) bytes of data.
64 bytes from 10.10.9.1: icmp_seq=1 ttl=64 time=1.32 ms
64 bytes from 10.10.9.1: icmp_seq=2 ttl=64 time=0.215 ms
```

Figura 3.33 Conectividad LOJ-CUE

– Central MAC

```
loj:~# ping 10.10.10.1
PING 10.10.10.1 (10.10.10.1) 56(84) bytes of data.
64 bytes from 10.10.10.1: icmp_seq=1 ttl=64 time=0.301 ms
64 bytes from 10.10.10.1: icmp_seq=2 ttl=64 time=0.226 ms
```

Figura 3.34 Conectividad LOJ-MAC

– ClienteTLOJ

```
loj:~# ping 172.31.15.4
PING 172.31.15.4 (172.31.15.4) 56(84) bytes of data.
64 bytes from 172.31.15.4: icmp_seq=1 ttl=64 time=10.6 ms
64 bytes from 172.31.15.4: icmp_seq=2 ttl=64 time=0.259 ms
```

Figura 3.35 Conectividad LOJ-TLOJ

3.2.6 CONFIGURACIÓN DE ASTERISK

La configuración de Asterisk, involucra editar cuatro archivos: dundi.conf, iax.conf, sip.conf y extensions.conf, de acuerdo a los servicios que se va a proveer al Sistema de Telefonía IP.

3.2.6.1 Configuración del protocolo *DUNDI*

Comprobada la conectividad entre todos los nodos de la red, se tiene por consiguiente conectividad entre las Centrales de Telefonía IP. Realizado este procedimiento,se procede a configurar el protocolo *DUNDi* en las Centrales de Telefonía IP. Al configurar del archivo *DUNDi*.conf se definen los parámetros de conexión entre los servidores Asterisk y los contextos que se van a compartir. En el mismo archivo se define el tipo de protocolo que los servidores usarán para efectuar y recibir llamadas una vez que se haya encontrado la ruta (puede ser IAX2, SIP o H323). Por último se modifica el Dialplan (plan de llamadas) para definir en los contextos configurados en dundi.conf, cuáles son las rutas o contextos que se quiere compartir.

A continuación se detalla una tabla con información útil para la configuración del protocolo DUNDi en las centrales de telefonía IP.

SERVIDOR	DIRECCIÓN MAC	INTERFAZ DE RED	SERVIDOR ADYACENTE	INTERFAZ DE RED
ESM	fe:fd:00:00:01:00	Eth1: 10.10.1.1/24	UIO	Eth1: 10.10.1.2/24
		Eth2: 10.10.3.1/24	GYE	Eth1: 10.10.3.2/24
UIO	fe:fd:00:00:02:00	Eth1: 10.10.1.2/24	ESM	Eth1: 10.10.1.1/24
		Eth2: 10.10.2.1/24	LAT	Eth1: 10.10.2.2/24
		Eth3: 10.10.11.2/24	GYE	Eth4: 10.10.11.1/24
LAT	fe:fd:00:00:03:00	Eth1: 10.10.2.2/24	UIO	Eth2: 10.10.2.1/24
		Eth2: 10.10.4.2/24	GYE	Eth2: 10.10.4.1/24

		Eth3: 10.10.5.1/24	AMB	Eth1: 10.10.5.2/24
GYE	fe:fd:00:00:04:00	Eth1: 10.10.3.2/24	ESM	Eth 2: 10.10.3.1/24
		Eth2: 10.10.4.1/24	LAT	Eth2: 10.10.4.2/24
		Eth3: 10.10.6.1/24	CUE	Eth1: 10.10.6.2/24
		Eth4: 10.10.11.1	UIO	Eth3: 10.10.11.2/24
AMB	fe:fd:00:00:06:00	Eth1: 10.10.5.2/24	LAT	Eth3: 10.10.5.1/24
		Eth2: 10.10.7.1/24	CUE	Eth2: 10.10.7.2/24
		Eth3: 10.10.8.1/24	MAC	Eth1: 10.10.8.2/24
CUE	fe:fd:00:00:05:00	Eth1: 10.10.6.2/24	GYE	Eth3: 10.10.6.1/24
		Eth2: 10.10.7.2/24	AMB	Eth2: 10.10.7.1/24
		Eth3: 10.10.9.1/24	LOJ	Eth1: 10.10.9.2/24
MAC	fe:fd:00:00:07:00	Eth1: 10.10.8.2/24	AMB	Eth3: 10.10.8.1/24
		Eth2: 10.10.10.1/24	LOJ	Eth2: 10.10.10.2/24
LOJ	fe:fd:00:00:08:00	Eth1: 10.10.9.2/24	CUE	Eth3: 10.10.9.1/24
		Eth2: 10.10.2/24	MAC	Eth2: 10.10.10.1/24

Tabla 3.2 Direccionamiento IP y Direcciones MAC

3.2.6.1.1 Edición del archivo dundi.conf

Central de Telefonía IP Esmeraldas

```
esm~#vim /etc/asterisk/dundi.conf
```

- **Archivo:**

```
[general]
department=IT
organization=esm
locality=Esmeraldas
stateprov=Esmeraldas
country=Ecuador
email=admin@esm
phone=+59322330220
bindaddr=0.0.0.0
port=4520
entityid=fe:fd:00:00:01:00
cachetime=2
ttl=32
autOKill=yes
secretpath=dundi
storehistory=yes

[mappings]

ecuador =>
registrosip,0,IAX2,dundi:${SECRET}@10.10.1.1/${NUMBER}

;esm->uio
[fe:fd:00:00:02:00]
model = symmetric
host = 10.10.1.2
inkey = uio
outkey = esm
include = all
permit = all
qualify = yes
order=primary

;esm->gye
[fe:fd:00:00:04:00]
model = symmetric
host = 10.10.3.2
inkey = gye
outkey = esm
include = all
permit = all
qualify = yes
order=primary
```

- **Explicación:**

[general]

En esta sección se configuran los parámetros informativos que identifican a cada Central de Telefonía IP.

- `bindaddr=0.0.0.0` ->En esta Interfaz es donde escucha peticiones *DUNDi*, si es 0.0.0.0 escucha en todas.
- `port=4520` ->Puerto predefinido usado por el protocolo *DUNDi*. Hay que abrirlo en el cortafuegos (4520 UDP).
- `entityid=fe:fd:00:00:01:01` ->Identificador de entidad *DUNDi*. Se usa la dirección MAC del servidor Asterisk.
- `cachetime=3600` ->Cuando se envía un solicitud para conocer la disponibilidad de una ruta y se encuentra una disponible, el resultado se guardará por el tiempo definido en este parámetro (en segundos) en la base de datos interna de Asterisk.
- `ttl=32` ->Time To Live es un número que representa cuanto lejos hay que ir para recibir una respuesta a la solicitud que se hizo para buscar una ruta.
- `autOKill=yes` ->Si no se recibe una respuesta dentro de 2000 ms de un servidor conectado a la red *DUNDi* se anulará la solicitud. Esto permite evitar quelas solicitudes queden colgadas en algún punto de la red.
- `secretpath=dundi`->El módulo*PBX_DUNDi* crea una clave que rotará de manera automática y que se guarda en la base de datos interna de Asterisk.
- `storehistory=yes` ->Mantiene un registro de las ultimas solicitudes efectuadas con los tiempos de respuesta por cada una. De esta forma es posible averiguar cuáles son los nodos lentos dentro de la red *DUNDi*.

[mappings]

Este contexto se comparte a otros peers, es decir a las Centrales de Telefonía IP adyacentes.

- `esm`: el contexto *DUNDi* que se va a compartir.
- `esm-local`: el contexto definido en el plan de llamadas que contendrá todas las rutas que se van a compartir.
- `IAX2`: la tecnología usada (protocolo) para las llamadas.
- `esm`: usuario que se configurará en el archivo iax.conf.
- `${SECRET}`: es la variable que contiene la clave que el módulo*DUNDi* crea en automático y que se usará para autenticar al usuario.
- `10.10.1.1`:Dirección IP del Servidor esm.
- `${NUMBER}`: la variable que contendrá el número que ha sido solicitado.

Una vez definido el mapping se procede a configurar los peers con los cuales se compartirá información.

- `fe:fd:00:00:02:00`:-> MAC address de la tarjeta de red de Servidor UIO con el que se crea la conexión.
- `model`:->Puede ser:

MODELO	FUNCIÓN
inbound	Recibe solamente solicitudes.
outbound	Solo efectúa solicitudes pero no las recibe.
symmetric	Recibe y efectúa solicitudes.

Tabla3.3 Tipos de modelos de configuración *DUNDi*

- `host`: -> Direccion IP o nombre de dominio del servidor Asterisk UIO.

- `inkey:->`Clave RSA usada para autenticarse con el servidor Asterisk UIO.
- `outkey:->` Clave RSA usada por el Servidor Asterisk UIO para autenticarse con el Servidor Asterisk ESM.
- `include:->` Incluye esta conexión para todas las solicitudes efectuadas en el Servidor Asterisk ESM.
- `permit:->`Se definen los contextos *DUNDi* a los que tendrá acceso este nodo.
- `qualify:->`Se controlará periódicamente que la conexión con el nodo esté activa.
- `order:->`Configura el orden de búsqueda que utiliza *DUNDi*, en este caso primario.

3.2.6.1.1 Creación de claves RSA para autenticación de centrales telefónicas

Terminada la edición, se guardan los cambios y se crea la clave RSA para autenticar el Servidor Asterisk UIO. Para crear la clave se usará una utilidad que viene con la instalación de Asterisk.

Para crear la clave se siguen los siguientes pasos:

1. Primero se ingresa en la carpeta donde se guardan las claves.

```
esm~#cd /var/lib/asterisk/keys
```

2. Se genera la clave mediante el comando *astgenkey.*

```
esm:/var/lib/asterisk/keys~#astgenkey -n esm
```

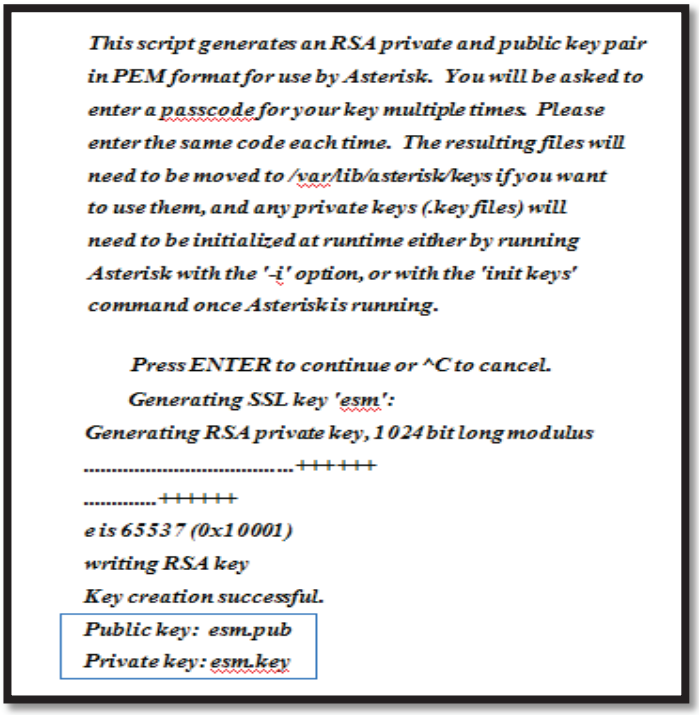


Figura 3.36 Creación de la llave de autenticación *DUNDi*

3. Se crearán dos claves, una pública y una privada. La clave pública se copia en el Servidor Asterisk UIO.

```
~#scp esm.pub root@10.10.1.2:/var/lib/asterisk/keys
```

La lógica de configuración en las demás centrales de telefonía IP sigue el mismo formato del explicado para la central telefónica de Esmeraldas. La diferencia radica en los nodos adyacentes a cada central telefónica. Las configuraciones se detallan en el anexo III.

De igual forma la creación de la clave RSA para autenticar cada Servidor Asterisk sigue los mismos pasos explicados anteriormente.

3.2.6.2 CONFIGURACIÓN DEL PROTOCOLO IAX

3.2.6.2.1 Edición del archivo iax.conf

En el archivo iax.conf se definirá las conexiones entre centrales de telefonía IP.En el archivo se especifican las extensiones: `[general]` para configurar los

parámetros de funcionamiento de IAX y la extensión **[dundi]** para configurar el funcionamiento del dominio DUNDi en el protocolo IAX.

Central de Telefonía IP Esmeraldas

```
vim /etc/asterisk/iax.conf
```

```
[general]
port=4569
bindaddr=10.10.1.1
bindaddr=10.10.3.1
[dundi]
type=friend
dbsecret=dundi/secret
context=registrosip
qualify=yes
disallow=all
allow=ulaw
allow=alaw
```

[general]

- **Port:** Puerto UDP en el que opera el protocolo IAX, le corresponde 4569.
- **bindaddr:** Dirección IP en la cual se escucharán los pedidos de conexiones. A Esmeraldas le corresponde su interface eth1: 10.10.1.1.

[dundi]

- **type:** Se define el tipo de comunicación*friend*, el cual funciona como peer y user a la vez es decir recibe y realiza llamadas.
- **dbsecret:** Especifica el password que usará para logar enviar información a otro servidor dentro del dominio DUNDi.
- **context:** Se configura el contexto del archivo extensions.confque utilizará IAX para su funcionamiento **[registrosip]**.
- **qualify:** Verifica la conectividad del usuario con el servidor Asterisk (ESM) con un ping.

- **disallow:** Se desactivan CODECS de audio y video que se utilizan por defecto, con el fin de especificar los CODECS precisos para la comunicación.
- **allow:** Define la utilización de los CODECS de audio que se utilizarán para las comunicaciones, en este caso ley U y ley A.

La configuración de iax.conf es similar en todas las centrales de telefonía IP, únicamente cambia la extensión **[general]** en el parámetro **bindaddr**debido a las conexiones entre centrales.La configuración del archivo iax.conf se describe en el anexo IV para las demás centrales telefónicas.

3.2.6.3 Configuración Del Protocolo SIP

3.2.6.3.1 Edición del archivo sip.conf

El archivo sip.conf es el mismo para todas las centrales de telefonía IP. En este archivo únicamente se define una sección **[general]**para configurar la funcionalidad del protocolo SIP, en esta sección se especifica la configuración de los usuarios y servidores del Sistema de telefonía IP.

```
vim /etc/asterisk/sip.conf
```

```
[general]
context=local
allow=all
regcontext=registrosip
bindport=5060
autocreatepeer=yes
insecure=very
```

[general]

- **context:** Se define el contexto de extensions.conf que atenderán las llamadas de los usuarios en el sistema .
- **allow:** Permite habilitar el uso de CODECS de audio.
- **regcontext:** Especifica el contexto donde se registrarán las extensiones dinámicas enroladas por los usuarios.

- **bindport:** Especifica el Puerto en el cual va a trabajar el protocolo SIP, en este caso 5060.
- **autocreatepeer:** Activada la opción yes, permite que los usuarios SIP se registren con su PBX local como peer.
- **insecure:** Configura el nivel de seguridad de acceso entre peers, es decir si se deben autenticar o no entre ellos.

3.2.6.4 Configuración del Dial-Plan

3.2.6.4.1 Edición del archivo extensions.conf

A continuación, el análisis del archivo que gobierna el plan de marcado (dialplan) de los sistemas de telefonía IP Asterisk en cada uno de los nodos del esquema mallado. Una vez configurados los servicios básicos de conectividad, enrutamiento IP y de llamada, este archivo coordina de manera funcional cada una de las labores del esquema de telefonía redundante.

Central de Telefonía IP Esmeraldas

```
vim /etc/asterisk/extensions.conf
```

```
[general]
static=yes
writeprotect=no
autofallthrough=yes
clearglobalvars=no

[globals]
LOCALPORT=5600
MEDIAPORT=6011
[local]
include => enrolamiento
include => desenrolamiento
exten => _XXXX,1,Dial(SIP/${EXTEN})
```

```
exten => _XXXX,2,System(echo $(date) La extension ${EXTEN} no se
encuentra localmente registrada >>
/etc/asterisk/fuentesip/redireccion.log)
exten => _XXXX,3,System(echo $(date) Iniciando BUSQUEDA en el
dominio DUNDI >> /etc/asterisk/fuentesip/redireccion.log)
exten => _XXXX,4,Macro(buscar-dundi,${EXTEN})

[local-privado]
include => registrosip

[enrolamiento]
exten =>
_456XXXX,1,Set(DESTINO=${DUNDILOOKUP(${EXTEN:3},ecuador,b)})
exten => _456XXXX,2,GotoIf($["${DESTINO}" = ""]?7:3)
exten => _456XXXX,3,System(sh
/etc/asterisk/fuentesip/desregistrar_remoto.sh ${EXTEN:3}
${DESTINO})
exten => _456XXXX,4,Wait,4
exten => _456XXXX,5,System(echo $(date) Extension ${EXTEN:3}
DESENROLADA de ${DESTINO} >>
/etc/asterisk/fuentesip/redireccion.log)
exten => _456XXXX,6,Goto(1)
exten => _456XXXX,7,System(sh /etc/asterisk/fuentesip/registrar.sh
${EXTEN:3} ${LOCALPORT} ${MEDIAPORT} 10.10.1.1)
exten => _456XXXX,8,Wait,5
exten => _456XXXX,12,System(echo $(date) Servidor de REDIRECCION
INICIADO mapeo extension logica ${EXTEN:3} y extension fisica
${CALLERID(num)} >> /etc/asterisk/fuentesip/redireccion.log)
exten => _456XXXX,13,System(echo $(date) Extension ${EXTEN:3}
ENROLADA } >> /etc/asterisk/fuentesip/redireccion.log)
exten => _456XXXX,14,Set(GLOBAL(LOCALPORT)=$[${LOCALPORT}+1])exten
=> _456XXXX,15,Set(GLOBAL(MEDIAPORT)=$[${MEDIAPORT}+10])
exten => _456XXXX,16,Answer
exten => _456XXXX,17,Wait,5
exten => _456XXXX,18,Hangup
;exten => _456XXXX,1,System(sh

/etc/asterisk/fuentesip/registrar.sh ${EXTEN:3} 5200 6011)
```

```
;exten => _456XXXX,2,Wait,2
;exten => _456XXXX,3,System(sh /etc/asterisk/fuentesip/redireccion.sh
${CALLERID(num)} 5200 6011)
;exten => _456XXXX,4,Wait,2
;exten => _456XXXX,5,Answer
;exten => _456XXXX,6,Wait,5
;exten => _456XXXX,7,Hangup

[desenrolamiento]
exten => _123XXXX,1,System(sh
/etc/asterisk/fuentesip/desregistrar_local.sh ${EXTEN:3})exten =>
_123XXXX,2,Wait,4
exten => _123XXXX,3,Set(DESTINO=${DUNDILOOKUP(${EXTEN:3},ecuador,b)})
exten => _123XXXX,4,GotoIf($["${DESTINO}" = ""]?8:5)
exten => _123XXXX,5,System(sh
/etc/asterisk/fuentesip/desregistrar_remoto.sh ${EXTEN:3} ${DESTINO})
exten => _123XXXX,6,Wait,4
exten => _123XXXX,7,Goto(3)
exten => _123XXXX,8,Answer
exten => _123XXXX,9,Wait,4
exten => _123XXXX,10,Hangup
[default]
include => local
[registrosip]
exten => _XXXX,2,Dial(SIP/${EXTEN},10)
exten => _XXXX,3,Hangup
exten => _XXXX,103,Hangup

[dundi-entrante]
include => registrosip

[buscar-dundi]
switch => DUNDI/ecuador

;MACRO BUSCAR_DUNDI

[macro-buscar-dundi]
exten => s,1,Goto(${ARG1},1)
include => local-privado
include => buscar-dundi
```

En este archivo se definen los siguientes contextos:

- **[general]:** En este contexto de definen algunas variables y comportamientos generales en el proceso de conexión y establecimiento de llamada. Estas variables son:
  - static: Tiene dos valores válidos yes y no. Yes permite que los comandos ingresados desde la consola de asterisk, tenga efecto sobre la configuración del dial-plan.
  - writeprotect: Al igual que static tiene dos posibles valores yes y no. Esta opción trabaja en conjunto con static, para permitir que un comando sea ejecutado desde la consola, con este fin el valor configurado es no.
  - autofallthrough: Esta opción es útil para evitar congestiones en el sistema, por ejemplo, cuando una extensión ya no tiene más procesos que ejecutar se termina la llamada con BUSY, CONGESTION o HANGUP. Esta opción se activa con yes.
  - clearglobalvars: Para liberar los valores que han sido asignado a las variables globales, cuando se recargan las extensiones o se reinicia Asterisk.
- **[globals]:** En este contexto se definen las variables globales que podrán ser utilizadas por todos los contextos en extensions.conf.
  - LOCALPORT->Se establece el valor inicial que tomará el puerto local en una comunicación. El valor configurado es 5600
  - MEDIAPORT->Se establece el valor inicial que tomará el puerto de media en una comunicación.El valor configurado es 6011.
- **[local]:** Este contexto es de especial importancia puesto que es el que permitirá, desde el principio, atender las llamadas que intenten hacerse a través de la central de telefonía IP. Fundamentalmente se presentarán 3 escenarios:
  - Cuando el usuario intente enrolar una extensión personal al sistema.
  - Cuando el usuario intente des-enrolar una extensión personal del sistema.

– Cuando el usuario intente llamar a una extensión, ya sea ésta local al nodo o remota, dentro del dominio *DUNDI*.

De esta manera, y mediante la directiva include, se incluyen los contextos **[enrolamiento]** y **[desenrolamiento]**, que cumplen las funciones que sus nombres describen. Asimismo, en caso de que el usuario haya digitado un código correspondiente a estas funciones, el plan de marcado intenta contactar directamente la extensión marcada de manera local (exten => _XXXX,1,Dial(SIP/${EXTEN})). Si la extensión marcada no está registrada localmente, entonces el dialplan va a la prioridad 2 y 3 del contexto donde se cargan mensajes de *log* en el archivo /asterisk/fuentesip/redireccion.log mediante el uso de la función System() que permite ejecutar comandos en la consola del sistema operativo.

Finalmente, y en la prioridad 4 del contexto, se llama al Macro buscar-dundi, definido posteriormente. Este macro permitirá, hacer una búsqueda de la extensión marcada en todo el dominio DUNDI.

- **[priv-local]:** Este contexto incluye solamente al contexto [registro-sip] que permitirá la comunicación de las extensiones registradas dinámicamente en cada nodo.

- **[registrosip]:** En este contexto se define la posibilidad de registrar en las centrales, de manera dinámica, las extensiones personales de los usuarios sin la necesidad de que estas extensiones hayan sido previamente creadas en el archivo sip.conf. Las extensiones registradas dinámicamente en el servidor de telefonía IP podrán entonces comunicarse entre sí, mediante la función Dial() de la prioridad 2, ya que la prioridad 1 se utiliza para cargar la extensión de manera dinámica. La prioridad 2 en este contexto permite marcar/llamar (Dial) a la extensión que haya digitado el usuario (${EXTEN}).
- **[enrolamiento]:** Este contexto es, probablemente, el más importante dentro del plan de marcado, pues maneja los procedimientos necesarios para dar la funcionalidad de movilidad a los usuarios del

sistema, mediante el enrolamiento de las extensiones personales, así como el des-enrolamiento automático (remoto). Del mismo modo, se levanta el servidor de redirección como parte del proceso de enrolamiento explicado previamente. Este contexto llevará a cabo cada una de las tareas que se indican en sus líneas cuando un usuario haya llamado a un número que inicia con los dígitos **456** y que contiene además cuatro dígitos más (**XXXX**). Cabe recordar que **456** es entonces el código que permitirá a los usuarios enrolar el número de 4 cifras que digiten luego. En la prioridad 1 del contexto se carga la variable DESTINO con la salida de la búsqueda *DUNDI* realizada por la función DUNDILOOKUP. Esta función del dialplan hace una búsqueda de la extensión que se le pasa como argumento (${EXTEN}:3) dentro del dominio DUNDI que también se pasa como argumento (ecuador). El resultado de esta búsqueda es una cadena de conexión que contendrá la dirección IP del nodo remoto donde está registrada la extensión que intenta enrolar el usuario.En la prioridad 2 se utiliza la función GotoIf() para que, en caso de que la búsqueda haya encontrado la extensión ya registrada/enrolada remotamente, el proceso de la llamada pase a la prioridad 3. En caso contrario el proceso pasará a la prioridad 7.En la prioridad 3, como es de esperarse, mediante la función System() se ejecuta el script de des-registro remoto (desregistrar_remoto.sh) pasándole como argumento la extensión marcada por el usuario, así como la cadena de conexión obtenida en el proceso de búsqueda DUNDI. Este script se encarga de des-registrar y des-enrolar la extensión indicada en el nodo pertinente.En la prioridad 4 se espera (Wait()) 4 segundos para dar tiempo para que el script anterior pueda cumplir su objetivo.En la prioridad 5, una vez que se ha des-registrado correctamente la extensión que desde un inicio está intentando enrolarse, se genera un mensaje de log dentro del archivo redirección.log para nuestra bitácora.En la prioridad 6 se regresa nuevamente a la prioridad 1 para realizar una nueva búsqueda DUNDI en el dominio ecuador, en caso de que existiesen enrolamientos previos adicionales en otros nodos.En la prioridad 7 se ejecuta el script

registrar.sh, para registrar localmente la extensión (${EXTEN}:3) que se solicita enrolar mediante la llamada inicial.En la prioridad 8 se espera un tiempo adecuado para que el script anterior pueda ejecutarse.En la prioridad 9 se carga un nuevo mensaje de log en el archivo redirección.log respecto del registro previo.En la prioridad 10 se ejecuta el script del servidor de redirección pasándole como argumentos los siguientes: el número desde donde se está realizando la llamada (número de extensión estática), el puerto local en el que escuchará el servidor y la dirección IP local del nodo, completando el socket. Es importante destacar que el puerto local donde se registró la extensión personal es el mismo donde se levanta el servidor de redirección. Esto tiene mucho sentido pues, cuando desde el dominio ecuador el nodo reciba una llamada hacia la extensión enrolada, esta llamada será dirigida al puerto mencionado, donde estará escuchando el servidor de redirección. El servidor de redirección reenviará entonces la llamada a la extensión estática correspondiente, como se explicó en el capítulo II.En la prioridad 11, se espera unos segundos.En las prioridades 12 y 13 se cargan otros mensajes de log que permitan identificar en la bitácora el evento anterior.En las prioridades 15 y 16 se modifican los valores de las variables globales que determinan los puertos que se utilizarán en los registros y al levantar el servidor de redirección. Esto es muy importante pues, por cada cliente que se enrole en el sistema, deberá existir un servicio de redirección funcionando en la central de telefonía que permita re-direccionar la llamada al dispositivo físico correspondiente.En la prioridad 16 se contesta la llamada, en la 17 se esperan unos segundos y finalmente en la 18 se cuelga la llamada.

Para una mejor comprensión a continuación se muestran los diagramas de flujo de la programación de los contextos enrolamiento y des-enrolamientorespectivamente.

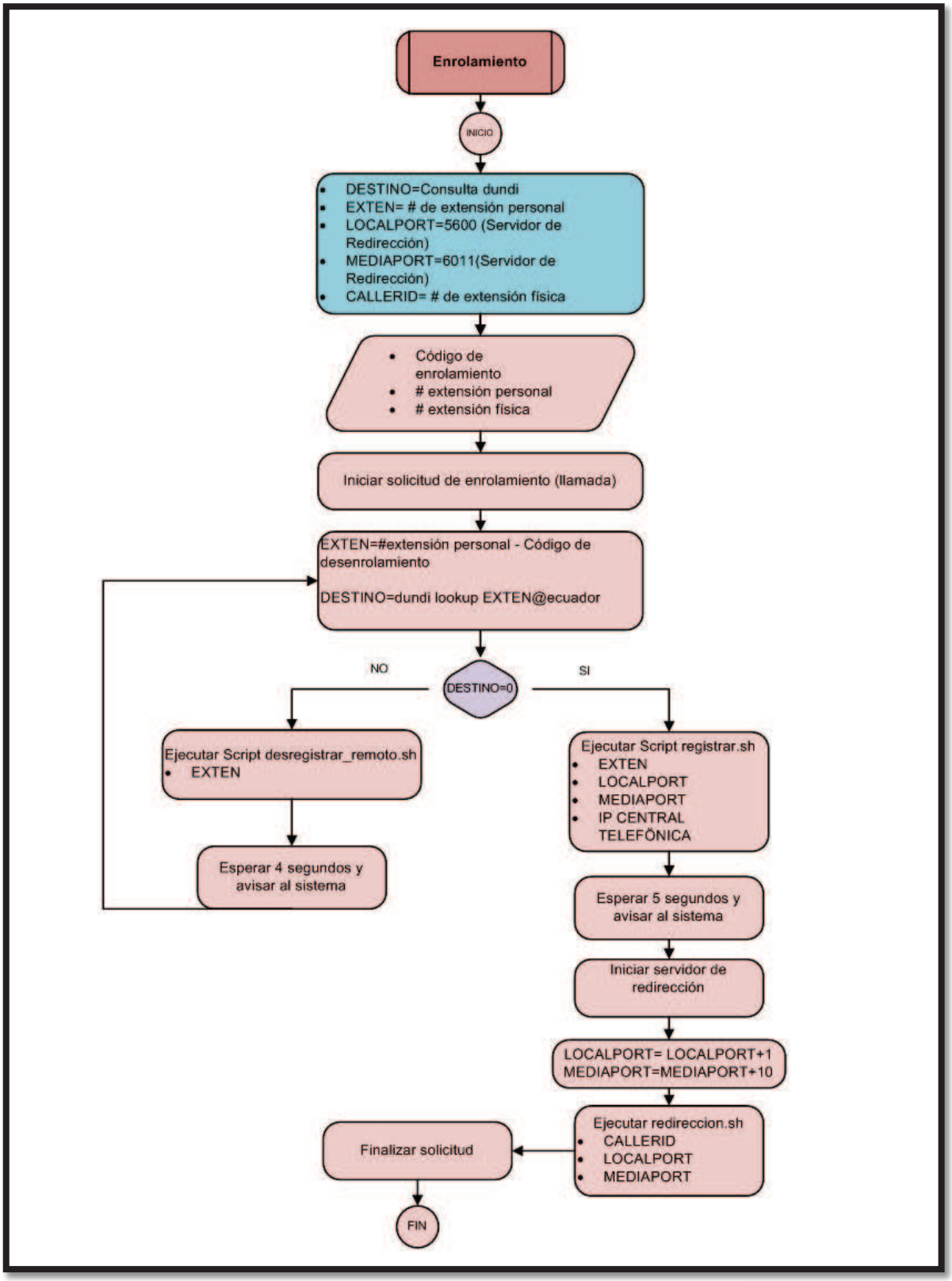


Figura 3.37 Diagrama de flujo del contexto enrolamiento

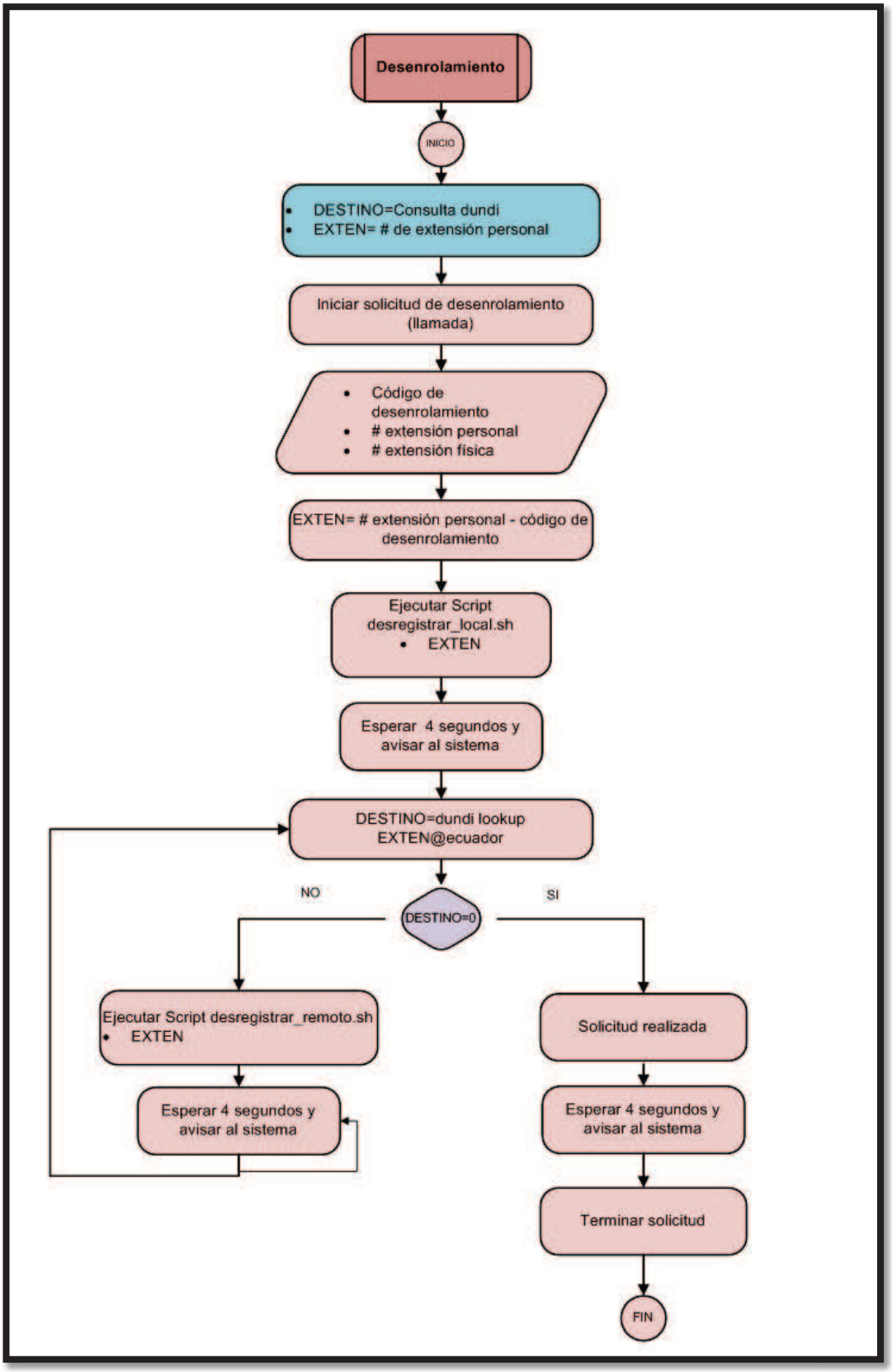


Figura 3.38 Diagrama de flujo del contexto desenrolamiento

- **[desenrolamiento]:** Este contexto (incluido en [local]) se encarga de atender las solicitudes directas de los usuarios para des-enrolar una extensión al hacer una llamada al número [código de des-enrolamiento(**123**) + número de extensión a des-enrolarse]. En la prioridad 1 de este contexto se ejecuta el script desregistrar_local.sh para des-registrar la extensión en caso de que estuviese registrada en el nodo local. En la prioridad 2 se esperan unos segundos. En la prioridad 3 se realiza una búsqueda de la extensión que se ha solicitado des-enrolar mediante la función DUNDILOOKUP. La cadena resultado de esta búsqueda se almacena dentro de la variable DESTINO. En la prioridad 4 se utiliza la Función GotoIf para saltar a la prioridad 8 en caso de que la variable DESTINO esté vacía (debido a que no se encontró la extensión registrada remotamente). En caso de que la variable DESTINO no esté vacía, salta a la prioridad 5. En la prioridad 5 se ejecuta el script desregistrar_remoto.sh que se encarga de des-enrolar la extensión (${EXTEN}:3) del nodo identificado en la búsqueda previa (${DESTINO}). En la prioridad 6 se esperan unos segundos. En la prioridad 7 se regresa a la prioridad 3 para verificar nuevamente si la extensión se encuentra enrolada en otro nodo. En la prioridad 8 se contesta la llamada. En la extensión 9 se esperan unos segundos y en la 10 finalmente se cuelga la llamada.

- **[default]:** En el contexto de funcionamiento por defecto se incluye el contexto [local] de manera que cualquier llamada que no se haya redirigido a un contexto en específico, sea atendida por el contexto [local].
- **[dundi- entrante]:** Este contexto es el que procesará las llamadas entrantes por el canal IAX de la central de telefonía IP y originadas por el protocolo *DUNDI*.

  Se puede observar también que dentro del archivo de configuración de IAX, iax.conf, se encuentra determinado, al igual que en SIP, el contexto que procesará las llamadas de cada protocolo utilizado, mediante la directiva context=dundi-entrante.

- **[buscar-dundi]:** Este contexto, mediante la sentencia ***switch***, permite utilizar el plan de marcado remoto que los otros nodos han compartido dentro del dominio, mediante el protocolo *DUNDI*.
- **[macro-buscar-dundi]:** Este es más que un contexto, un macro en Asterisk, que resulta como la definición de una función en programación, que en este caso permite fundamentalmente saltar al contexto buscar-dundipara realizar la búsqueda de determinada extensión dentro del dominio al cual pertenecen las centrales de telefonía IP.

La programación de este archivo es similar para todas las centrales telefónicas, únicamente cambia la especificación del servidor Asterisk en el cual se enrolarán las extensiones personales de los usuarios. En el caso de Esmeraldas la IP del servidor es **10.10.1.1**, de igual forma en cada archivo se configura la respectiva IP de acuerdo al direccionamiento establecido.

```
exten => _456XXXX,7,System(sh
/etc/asterisk/fuentesip/registrar.sh ${EXTEN:3} ${LOCALPORT}
${MEDIAPORT} 10.10.1.1)
```

3.2.7 PRUEBAS DE FUNCIONAMIENTO DE ASTERISK

Una vez configurado los archivos de Asterisk en las Centrales de Telefonía IP, se procede a verificar la comunicación entre ellas.Para iniciar Asterisk en cada central se ejecuta el comando en cada central de telefonía IP:

```
~# asterisk -rvvvvvvvvvvvvvvvvvvvvvvvvv
```

Con este comando se activará el CLI (Command Line Interface) de asterisk, en el cual se verifica la existencia de las adyacencias punto a punto del dominio **ecuador**, configuradas mediante DUNDI, utilizando el comando dundi show peers en cada nodo servidor de telefonía IP.

```
cli> dundi show peers
```

- **Esmeraldas**

```
esm*CLI> dundi show peers
EID                 Host                   Model      AvgTime  Status
fe:fd:00:00:04:00   10.10.3.2        (S)  Symmetric  Unavail  OK (64 ms)
fe:fd:00:00:02:00   10.10.1.2        (S)  Symmetric  Unavail  OK (82 ms)
2 dundi peers [2 online, 0 offline, 0 unmonitored]
```

Figura 3.39 Peers adyacentes a la central Esmeraldas

- **Quito**

```
uio*CLI> dundi show peers
EID                 Host                   Model      AvgTime  Status
fe:fd:00:00:04:00   10.10.11.1       (S)  Symmetric  Unavail  OK (130 ms)
fe:fd:00:00:03:00   10.10.2.2        (S)  Symmetric  Unavail  OK (63 ms)
fe:fd:00:00:01:00   10.10.1.1        (S)  Symmetric  Unavail  OK (40 ms)
3 dundi peers [3 online, 0 offline, 0 unmonitored]
```

Figura 3.40 Peers adyacentes a la central Quito

- **Latacunga**

```
lat*CLI> dundi show peers
EID                 Host                   Model      AvgTime  Status
fe:fd:00:00:02:00   10.10.2.1        (S)  Symmetric  Unavail  OK (110 ms)
fe:fd:00:00:04:00   10.10.4.1        (S)  Symmetric  Unavail  OK (2 ms)
fe:fd:00:00:06:00   10.10.5.2        (S)  Symmetric  Unavail  OK (48 ms)
3 dundi peers [3 online, 0 offline, 0 unmonitored]
lat*CLI>
```

Figura 3.41 Peers adyacentes a la central Latacunga

- **Ambato**

```
amb*CLI> dundi show peers
EID                 Host                   Model      AvgTime  Status
fe:fd:00:00:07:00   10.10.8.2        (S)  Symmetric  Unavail  OK (134 ms)
fe:fd:00:00:05:00   10.10.7.2        (S)  Symmetric  Unavail  OK (99 ms)
fe:fd:00:00:03:00   10.10.5.1        (S)  Symmetric  Unavail  OK (80 ms)
3 dundi peers [3 online, 0 offline, 0 unmonitored]
amb*CLI>
```

Figura 3.42 Peers adyacentes a la central Ambato

- **Guayaquil**

```
gye*CLI> dundi show peers
EID                 Host                   Model      AvgTime  Status
fe:fd:00:00:02:00   10.10.11.2       (S) Symmetric  Unavail  OK (56 ms)
fe:fd:00:00:01:00   10.10.3.1        (S) Symmetric  Unavail  OK (56 ms)
fe:fd:00:00:05:00   10.10.6.2        (S) Symmetric  Unavail  OK (268 ms)
fe:fd:00:00:03:00   10.10.4.2        (S) Symmetric  Unavail  OK (353 ms)
4 dundi peers [4 online, 0 offline, 0 unmonitored]
```

Figura 3.43 Peers adyacentes a la central Guayaquil

- **Cuenca**

```
cue*CLI> dundi show peers
EID                 Host                   Model      AvgTime  Status
fe:fd:00:00:08:00   10.10.9.2        (S) Symmetric  Unavail  OK (60 ms)
fe:fd:00:00:04:00   10.10.6.1        (S) Symmetric  Unavail  OK (59 ms)
fe:fd:00:00:06:00   10.10.7.1        (S) Symmetric  Unavail  OK (60 ms)
3 dundi peers [3 online, 0 offline, 0 unmonitored]
cue*CLI>
```

Figura 3.44 Peers adyacentes a la central Cuenca

- **Macas**

```
mac*CLI> dundi show peers
EID                 Host                   Model      AvgTime  Status
fe:fd:00:00:08:00   10.10.10.2       (S) Symmetric  Unavail  OK (59 ms)
fe:fd:00:00:06:00   10.10.8.1        (S) Symmetric  Unavail  OK (137 ms)
2 dundi peers [2 online, 0 offline, 0 unmonitored]
mac*CLI>
```

Figura 3.45 Peers adyacentes a la central Macas

- **Loja**

```
loj*CLI> dundi show peers
EID                 Host                   Model      AvgTime  Status
fe:fd:00:00:05:00   10.10.9.1        (S) Symmetric  Unavail  OK (81 ms)
fe:fd:00:00:07:00   10.10.10.1       (S) Symmetric  Unavail  OK (81 ms)
2 dundi peers [2 online, 0 offline, 0 unmonitored]
loj*CLI>
```

Figura 3.46 Peers adyacentes a la central Loja

3.2.8 INSTALACIÓN DE *SIPP*

Comprobada las adyacencias en las centrales de telefonía se proceden hacer las pruebas de funcionamiento de todo el sistema empleando la herramienta *SIPP* para simular la generación de tráfico SIP.

Para la instalación de *SIPP* se siguen los siguientes pasos:

1. Instalación de librerías requeridas.

```
$ sudo apt-get install ncurses-dev
$ sudo apt-get install build-essential
```

2. Obtención del software *SIPP*en formato comprimido .tar.gz.

```
wget -m -nd
http://downloads.sourceforge.net/project/sipp/sipp/3.1/sipp.3.1.src.
tar.gz
```

4. Extracción del paquete.

```
tar -xzf sipp.3.1.src.tar.gz
```

5. Para la última fase de instalación se ingresa al archivo /sipp.svn/scenario.hpp, con el fin de aumentar en la cabecera dos librerías para realizar la compilacióndel programa.

```
$ cdsipp.svn
$ vim scenario.hpp
```

```
#include <limits.h>
#include <sys/socket.h>
```

6. Terminada la edición del archivo scenario.hpp, se guarda los cambios y se compila empleando el comando make.

```
$make
```

7. Terminada la instalación de *SIPP,* se procede a realizar los scripts para simular los diferentes escenarios expuestos en el capítulo II en el funcionamiento del prototipo.

3.3 DESARROLLO DE SCRIPTS EN SIPP PARA SIMULAR LA GENERACIÓN DE TRÁFICO IP

3.1.1 REGISTRO

SCRIPTS	DATOS DE INGRESO	RESULTADO
registrar.xml	• # extensión estática • Puerto local de registro • Puerto de media de registro • Dirección IP de la central telefónica de registro	Registro de una instancia SIP (que simula ser un teléfono IP) en la central telefónica.

Tabla 3.4 script registrar.xml

3.3.1.1 Edición del script registrar.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<scenario name="registrar">
<send retrans="500">
<![CDATA[
                        REGISTER sip:10.0.0.0 SIP/2.0
                        Via: SIP/2.0/[transport]
[local_ip]:[local_port];branch=[branch]
                        From: ua1
<sip:[service]@10.0.0.1:[local_port]>;tag=[call_number]
                        To: ua1 <sip:[service]@10.0.0.2:[local_port]>
                        Call-ID: [call_id]
                        CSeq: 1 REGISTER
                        Contact: sip:[service]@[local_ip]:[local_port]
                        Content-Length: 0
                        Expires: 4800
                ]]>
</send>

<recv response="200">
</recv>

</scenario>
```

3.3.1.2 Explicación del script registrar.xml

Este escenario inicia enviando (send) un mensaje *REGISTER* a la central telefónica (variable [service] del campo To: del mensaje y este parámetro se pasa con la opción –s en línea de comandos). Luego la central telefónica confirma el registro con un mensaje OK (200), y por esa razón en el escenario anterior se espera recibir (recv) este mensaje.El parámetro *Expires* del archivo registrar.xml determina la cantidad de tiempo que el registro estará presente en la central telefónica. Luego de este tiempo (en segundos) el registro desaparecerá

Para ejecutar el escenario anteriormente descrito, se utilizará el comandosippcon las opciones:

OPCIÓN	FUNCIÓN
-s	Servicio dentro del archivo xml, en este caso corresponde la extensión estática del teléfono IP.
-p	Puerto local para la comunicación.
-mp	Puerto de media para la comunicación
-r	Funciona en conjunto con la opción -rp.
-rp	Establece el tiempo en milisegundos para le ejecución del script xml.
-sf	Path de localización del archivo registrar.xml.
0.0.0.0	Dirección IP de la central telefónica donde se va a registrar el teléfono IP.
-m	Parámetro para indicar las veces que se ejecutará el script.
-bg	Ejecución en background.
-trace_err	Generar un archivo de log

Tabla 3.5 Opciones del comando sipp

Por ejemplo:

```
#sipp -s 6000 -p 5200 -mp 6200 -r 1 -rp 2000 -sf
/etc/asterisk/fuentesip/registrar.xml 192.168.1.1 -trace_err -m 1 -bg >>
/etc/asterisk/fuentesip/logs
```

Esto implica ejecutar el escenario registrar.xml dentro de la carpeta /etc/asterisk/fuentesip, pasándole como argumentos la extensión 6000, utilizando como puerto de origen de registro al 5200 y como puerto de media el 6200. La opción –m que en este caso pasa el valor de 1, sirve para indicar que luego del primer registro, el escenario deja de ejecutarse. La dirección 192.168.1.1 identifica la dirección IP de la central telefónica a la que se registra la extensión, además los archivos de logs se localizarán en /etc/asterisk/fuentesip/logs.

Suponiendo que se ejecute este comando desde un host con dirección IP 192.168.1.4, permitiría el registro de la extensión 6000 en la central telefónica con dirección IP 192.168.1.1.

3.3.1.3 Script de ejecución registrar.sh

Para facilitar y simplificar la ejecución del script registrar.xml, se crea un script siguiendo el formato de ejecución con el comando sipp. Como resultado únicamente es necesario el ingreso de las variables: extensión estática, puerto local, puerto de media y dirección IP de la central telefónica.

```
/usr/sbin/sipp -s $1 -p $2 -mp $3 -r 1 -rp 2000 -sf
/etc/asterisk/fuentesip/registrar.xml $4 -trace_err -m 1 -bg>>
/etc/asterisk/fuentesip/logs
```

Retomando el ejemplo anterior, con el registro de la extensión 6000, el comando sería el siguiente:

```
# sh registrar.sh 6000 5200 6011 10.0.0.2
```

3.3.2 LLAMAR

SCRIPTS	DATOS DE INGRESO	RESULTADO
llamar.xml	• # extensión estática llamada • Puerto local de comunicación • Puerto de media de comunicación • Dirección IP de la central telefónica local	Se establece la solicitud de conexión desde una extensión específica hacia la extensión llamada.
	• # extensión personal llamada • Puerto local de comunicación • Puerto de media de comunicación • Dirección IP de la central telefónica local • # extensión estática asociada al # de extensión personal	

Tabla 3.6 script llamar.xml

3.3.2.1 Edición del script de llamar.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<scenario name="llamar">
<send retrans="500">
<![CDATA[
                        INVITE sip:[service]@10.0.0.2 SIP/2.0
                        Via: SIP/2.0/[transport]
[local_ip]:[local_port];branch=[branch]
                        From: jose[call_number]
<sip:[orgext]@10.0.0.1:[local_port]>;tag=[call_number]
                        To: juan<sip:5031@10.0.0.2:[remote_port]>
                        Call-ID: [call_id]
                        CSeq: 1 INVITE
                        Contact: sip:6000@[local_ip]:[local_port]
                        Max-Forwards: 70
                        Subject: Performance Test
                        Content-Type: application/sdp
                        Content-Length: [len]
```

```
                        v=0
                         o=user1 53655765 2353687637 IN IP[local_ip_type]
[local_ip]
                         s=-
                         c=IN IP[media_ip_type] [media_ip]
t=0 0
                         m=audio [media_port] RTP/AVP 0
a=rtpmap:0 PCMU/8000
                 ]]>
</send>
<recv response="100" optional="true">
</recv>

<recv response="180" optional="true">
</recv>

<recv response="200" crlf="true">
</recv>


<send>
<![CDATA[

                         ACK sip:[service]@[remote_ip]:[remote_port]
SIP/2.0
                         Via: SIP/2.0/[transport] [local_ip]:[local_port]
                         From:
jose<sip:6000@[local_ip]:[local_port]>;tag=[call_number]
                         To:
juan<sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
                         Call-ID: [call_id]
                         CSeq: 1 ACK
                         Contact: sip:6000@[local_ip]:[local_port]
Max-Forwards: 70
                         Subject: Performance Test
Content-Length: 0

]]>
</send>


<send retrans="500">
<![CDATA[
                         BYE sip:[service]@10.0.0.2 SIP/2.0
Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
                         From:
jose<sip:6000@10.0.0.1:[local_port]>;tag=[call_number]
                         To:
juan<sip:[service]@10.0.0.2:[remote_port]>[peer_tag_param]
                         Call-ID: [call_id]
                         CSeq: 2 BYE
                         Contact: sip:6000@[local_ip]:[local_port]
                         Max-Forwards: 70
                         Subject: Performance Test
                         Content-Length: 0
                 ]]>
</send>
```

```
<recv response="200" crlf="true">
</recv>

<ResponseTimeRepartition value="10, 20, 30, 40, 50, 100, 150, 200"/>

<CallLengthRepartition value="10, 50, 100, 500, 1000, 5000, 10000"/>

</scenario>
```

3.3.2.2 Explicación del script llamar.xml

Para realizar este script se toma como referencia los mensajes SIP que se generan en una llamada, los cuales están descritos en el Capítulo I. Al igual que el anterior script .xml para su ejecución se emplea el comando sipp, con las opciones descritas en la tabla 3.5

Por ejemplo:

Para llamar a la extensión 6001 desde el puerto 5200, puerto de media 6200 en la central con dirección IP 192.168.1.1, se debería ejecutar el escenario mediante el siguiente comando:

```
# sipp -s 6001 -p 5200 -mp 6200 -r 1 -rp 2000 -sf llamar.xml
192.168.1.1 -trace_err -m 1
```

3.3.2.3 Script de ejecución llamar.sh

Se crea un script para facilitar la ejecución del escenario.

```
/usr/sbin/sipp -s $1 -p $2 -mp $3 -r 1 -rp 2000 -sf
/etc/asterisk/fuentesip/registrar.xml $4 -trace_err -m 1 -bg>>
/etc/asterisk/fuentesip/logs
```

Retomando el ejemplo anterior, para llamar a la extensión 6001, el comando sería el siguiente:

```
# sh llamar.sh 6001 5200 6200192.168.1.1
```

3.3.3 ESPERAR LLAMADA

SCRIPTS	DATOS DE INGRESO	RESULTADO
esperar.xml	• Puerto local de registro • Puerto de media de registro • Dirección IP del teléfono IP	Se inicia el servicio de esperar llamada en el teléfono IP, para ser alcanzado por otro usuario en el sistema.

Tabla 3.7 script esperar.xml

3.3.3.1 Edición del script esperar.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<scenario name="Asterisk Wait INVITE">

<recv request="INVITE" crlf="true">
<action>
<exec command="echo [last_From] is the from header received >>
from_list.log"/>
</action>
</recv>
<send>
<![CDATA[

      SIP/2.0 100 Trying
      [last_Via:]
      [last_From:]
      [last_To:];tag=[call_number]
      [last_Call-ID:]
      [last_CSeq:]
      Contact: <sip:[local_ip]:[local_port];transport=[transport]>
      Content-Length: 0

    ]]>

</send>
```

```
<send>
<![CDATA[

      SIP/2.0 180 Ringing
      [last_Via:]
      [last_From:]
      [last_To:];tag=[call_number]
      [last_Call-ID:]
      [last_CSeq:]
      Contact: <sip:[local_ip]:[local_port];transport=[transport]>
      Content-Length: 0
    ]]>
</send>

<send>
<![CDATA[

                SIP/2.0 200 OK
                [last_Via:]
                [last_From:]
                [last_To:]
                [last_Call-ID:]
                [last_CSeq:]
                Contact:
<sip:[local_ip]:[local_port];transport=[transport]>
                Content-Length: 0
        ]]>
</send>

   <recv request="ACK" crlf="true">
   </recv>

   <recv request="BYE" crlf="true">
   </recv>
<send>
<![CDATA[

                        ACK sip:[service]@[remote_ip]:[remote_port]
SIP/2.0
                        Via: SIP/2.0/[transport]
[local_ip]:[local_port]
                        From:
jose<sip:6000@[local_ip]:[local_port]>;tag=[call_number]
                        To:
juan<sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
                        Call-ID: [call_id]
                        Cseq: 1 ACK
                        Contact: sip:6000@[local_ip]:[local_port]
                        Max-Forwards: 70
                        Subject: Performance Test
                        Content-Length: 0

                ]]>
</send>


</scenario>
```

3.3.3.2 Ejecución del script esperar.xml

El comando usado para la ejecución del escenario es *SIPP* con las opciones descritas en la tabla 3.5.

Por ejemplo:

Para esperar una llamada en el puerto 5201 en una instancia en la dirección IP 192.168.1.4, se debería ejecutar el escenario mediante el siguiente comando:

```
# sipp -p 5201 -mp 6012 -r 1 -rp 2000 -sf esperar.xml
192.168.1.4 -m 1 -trace_err
```

De este modo el terminal VoIP con dirección IP 192.168.1.4 habilita la opción de recibir llamadas de otros usuarios en el Sistema de Telefonía IP.

3.3.3.3 Script de ejecuciónesperar.sh

```
sipp -p $1 -mp $2 -r 1 -rp 2000 -sf esperar.xml $3 -m 2 -
trace_err
```

Para realizar el ejemplo anterior de espera de llamada solo basta con ejecutar:

```
#sh esperar.sh 5201 6012 192.168.1.4
```

3.3.4 SERVIDOR DE REDIRECCIÓN

SCRIPTS	DATOS DE INGRESO	RESULTADO
redirección.xml	• # de extensión estática • Puerto local del # de extensión enrolada • Puerto de media del # de extensión enrolada	La central telefónica conoce a qué número de extensión estática debe redirigir la solicitud de llamada al levantar el servicio de redirección en el puerto asignado

Tabla 3.8 script redireccion.xml

3.3.4.1 Edición del script redireccion.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">

<scenario name="Asterisk Wait INVITE">

<recv request="INVITE" crlf="true">
</recv>

<send>
<![CDATA[
                SIP/2.0 100 sip:[local_ip] SIP/2.0
                Via: SIP/2.0/UDP [local_ip]:[local_port]
                Max-Forwards:5
                [last_To:]
                [last_From:]
                Call-ID: [call_id]
                [last_CSeq:]
          ]]>
</send>

<send>
<![CDATA[
                SIP/2.0 302 sip:[local_ip] SIP/2.0
                Via: SIP/2.0/UDP [local_ip]:[local_port]
                Max-Forwards:5
                [last_To:]
                [last_From:]
                Call-ID: [call_id]
                [last_CSeq:]
                Contact: <sip:[nfisico]@[local_ip]>
                Expires: 10
                Content-Length:0
          ]]>
</send>

<recv response="200" optional="true">
</recv>

<recv request="ACK">
</recv>

</scenario>
```

3.3.4.2 Ejecución del script redireccion.xml

De acuerdo a la funcionalidad del servidor de redirección descrita en el CapítuloII, se realiza el script .xml el cual será ejecutado directamente desde el archivo extensions.conf cuando un usuario proceda a enrolar su número de extensión personal. El comando de ejecución del escenario es sipp, a continuación se muestra el script redirección.sh utilizado por extensions.conf

```
/usr/sbin/sipp -key nfisico $1 -p $2 -mp $3 -r 1 -rp 2000 -sf
/etc/asterisk/fuentesip/redireccion.xml $4 -trace_err -m 5 -bg
>>   /etc/asterisk/fuentesip/log   -watchdog_minor_maxtriggers
600000
```

Los argumentos que recibe este script son:El número desde donde se está realizando la llamada (extensión estática), el puerto local en el que escuchará el servidor, el puerto de media y la dirección IP de la central telefónica.

3.3.5 DES-REGISTRO DE UN TELÉFONO IP Y DESENROLAMIENTO DE UN USUARIO

El proceso de des-registro y des-enrolamiento es muy similar al proceso de registro, es necesario solamente enviar un mensaje de tipo REGISTER a la central de donde se desea des-registrar o des-enrolar determinada extensión.

SCRIPTS	DATOS DE INGRESO	RESULTADO
desregistrar.xml	• # de estática • Dirección IP del servidor en el cual está registrada la extensión estática.	La instancia SIP que simula ser un teléfono IP se des-registra de la central telefónica asociada.
	• # extensión personal	Un usuario des-enrola su # de extensión personal, la cual está asociada a un teléfono IP registrado en la central.

Tabla 3.9 script desregistrar.xml

3.3.5.1 Edición del script desregistrar.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">
```

```
<scenario name="desregistrar">
<send retrans="500">
<![CDATA[
                        REGISTER sip:10.0.0.2 SIP/2.0
                        Via: SIP/2.0/[transport]
[local_ip]:[local_port];branch=[branch]
                        From: ua1
<sip:[service]@10.0.0.1:[local_port]>;tag=[call_number]
                        To: ua1 <sip:[service]@10.0.0.2:[local_port]>
                        Call-ID: [call_id]
                        CSeq: 1 REGISTER
                        Contact: sip:[service]@[local_ip]:[local_port]
                        Content-Length: 0
                        Expires: 0
                ]]>
</send>

<recv response="200">
</recv>

</scenario>
```

Se puede observar que la única diferencia con el mensaje de REGISTER es el tiempo en el que expira el cual es 0. Se trata de un mensaje de registro de la extensión que caduca inmediatamente, permitiendo su des-registro.

3.3.5.2 Ejecución del script desregistrar.xml

3.3.5.3 Script de ejecución desregistrar_local_sh

Si se vuelve sobre la descripción del funcionamiento del archivo extensions.conf, es posible observar que el script desregistrar_local.sh es llamado cuando elusuario marca 123+[número de extensión a des-enrolarse]. Este evento obedece a un intento deliberado del usuario por des-enrolar su extensión personal.

El script desregistrar_local.sh, como su nombre lo indica, permite des-registrar una extensión si se encontrase registrada en la central telefónica que recibe inicialmente la petición de des-registro. Esto podría parecer extraño pero tiene sentido cuando se observa que la función DUNDILOOKUP que se hace en el dominio ecuador, en busca de extensiones registradas, busca efectivamente en todos los nodos del dominio, excepto en el que origina la búsqueda.

El contenido de este script será el siguiente:

```
/usr/sbin/sipp -s $1 -r 1 -rp 2000 -sf
/etc/asterisk/fuentesip/desregistrar.xml 10.10.1.2 -trace_err
-m 1 -bg >> /etc/asterisk/fuentesip/logs
```

Se observa que es el mismo comando analizado previamente en el ejemplo anterior con la diferencia de que los parámetros han sido reemplazados por variables, que podrían cargarse en la ejecución del script de la siguiente manera:

```
# sh desregistrar_local.sh 6000
```

En este caso el script des-registrará la extensión estática 6000 de la central 10.10.1.2, en caso de que estuviese registrada.

3.3.5.4 Script de ejecución desregistrar_remoto_sh

Este script contiene, en su parte fundamental la misma programación que se indicó en el script desregistrar_local.sh. Sin embargo, existen tres líneas adicionales que lo único que hacen es cortar la cadena que el script recibe como parámetro 2.

Para comprender esto, es necesario regresar al script extensions.conf, en el contexto **[enrolamiento]**. En la prioridad 3 se ejecuta el script desregistrar_remoto.sh pasándole como argumentos la extensión que se desea des-registrar y la cadena obtenida de la función DUNDILOOKUP que identifica la central telefónica donde dicha extensión está registrada. Pero cabe recordar que esa cadena incluye más información que la dirección IP del nodo remoto. Esta dirección IP está delimitada por un carácter @ a la izquierda y un carácter / a la derecha.Como se ha analizado, el script de des-registro necesita solamente la dirección IP. Por esta razón es necesario extraer de la cadena la dirección IP y ese dato pasarle al comando sipp. Ese paso adicional está incluido en el script, como se puede observar a continuación.

```
DEST=$2
VAR=${DEST#*@}
DIRIP=${VAR%/*}
/usr/sbin/sipp -s $1 -r 1 -rp 2000 -sf
/etc/asterisk/fuentesip/desregistrar.xml $DIRIP -trace_err -m
1 -bg>> /etc/asterisk/fuentesip/logs
```

En la primera línea se asigna el valor del segundo parámetro pasado al script a la variable DEST. Recordar que este parámetro corresponde a la cadena de conexión indicada previamente.

En la segunda línea, la variable DEST es cortada desde el inicio hasta el carácter @ (éste incluido) y el resultado es almacenado en la variable VAR.

En la segunda línea, la variable VAR es cortada desde el carácter / (éste incluido) hasta el final, obteniendo la dirección IP que se buscaba, y almacenándola en la variable DIRIP. Esta última, como se puede observar, es pasada como argumento al comando sipp.

```
# sh desregistrar_remoto.sh 6000DIRIP
```

En el caso de realizar el des-enrolamiento se debe entonces realizar una llamada con el código 123 más extensión personal, puerto local, puerto de media, dirección IP de la central telefónica local que atenderá la llamada y número de extensión estática del teléfono IP, con lo cual se activa internamente el funcionamiento de los scripts des-registrar dependiendo de la localización dela extensión personal.

```
#sh llamar.sh 1236000 6012 5014 192.168.1.1
```

Para realizar el des-registro de un teléfono IP se emplea el script desregistrar_local.sh, desde el terminal VoIP.

3.4 VERIFICACIÓN DE FUNCIONAMIENTO DE PROCESOS BÁSICOS DE LLAMADA MEDIANTE *SIPP* Y DUNDI

3.4.1 PROCESO DE REGISTRO(EXTENSIONES ESTÁTICAS)

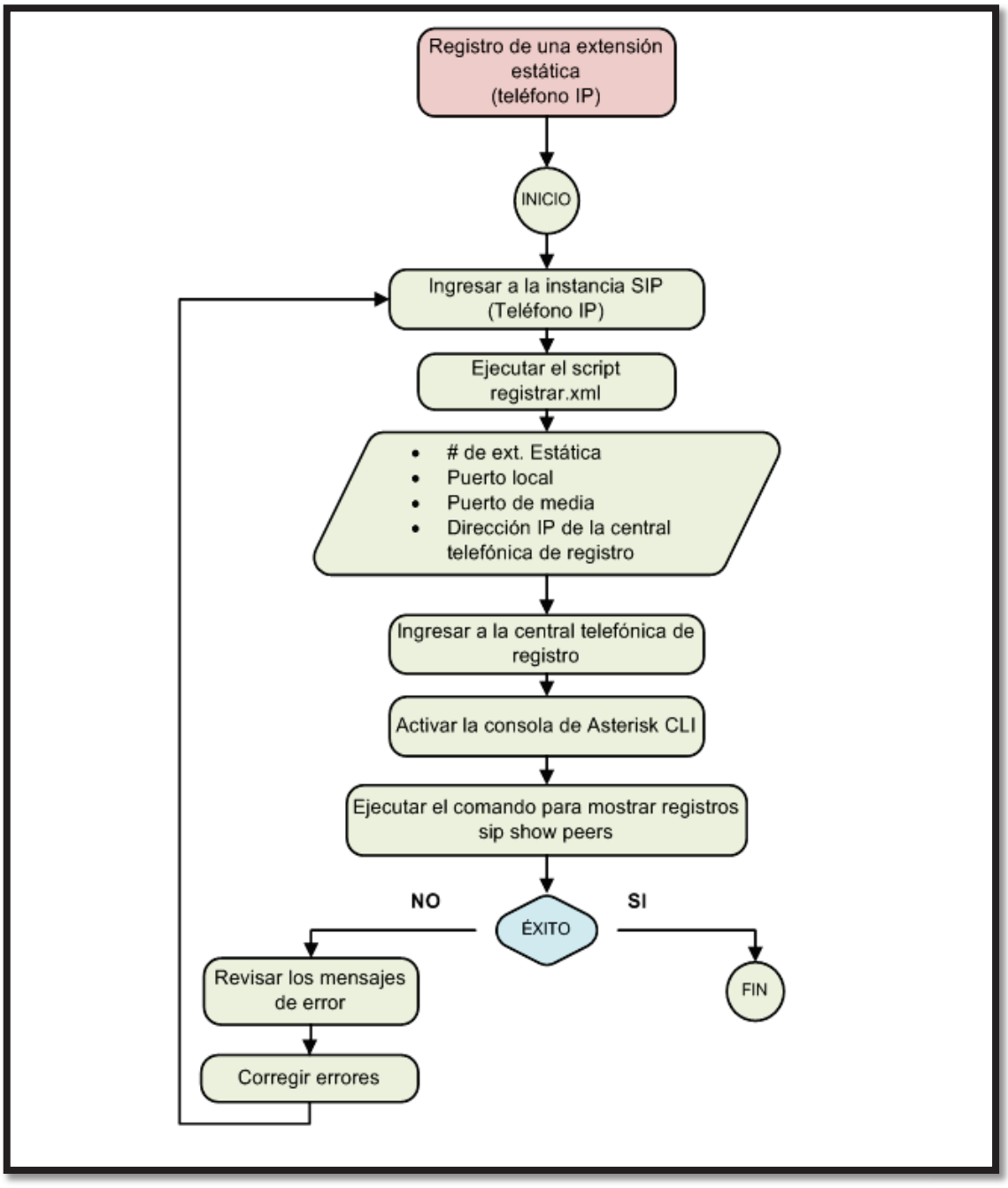


Figura 3.47 Diagrama de flujo del proceso de registro

Comprobada la conectividad, enrutamiento y funcionamiento de *DUNDi,*se toma como referencia el diagrama de flujo de la figura 3.47 yse procede a registrar el terminal de VoIP con número de extensión estática 1050 en la central telefónica de Esmeraldas. Para este propósitose ejecuta el script de registrodenominadoregistrar.sh desde el cliente TESM hacia el nodo en Esmeraldas (ESM). De igual forma se procede a registrar el terminal de VoIP con número de extensión estática 9050 en la central telefónica de Loja, es decir se ejecuta el script registrar.sh desde el cliente TLOJ hacia el nodo en Loja (LOJ).

- **Host cliente tesm**

  - Ingresar a la máquina virtual.

```
root@servidor-desktop:/usr/share/vnuml/examples# ssh tesm
```

```
root@servidor-desktop:/usr/share/vnuml/examples# ssh tesm
Linux vm 2.6.31.5 #4 Fri Oct 23 10:26:16 BST 2009 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jul 30 01:23:31 2010 from 192.168.0.133
tesm:~#
tesm:~#
```

Figura 3.48 Inicio de la máquina virtual TESM

  - Ingresar al directorio que contiene los scripts de ejecución .sh y ejecutarlo mediante el comando sh.

```
tesm:~# cd /etc/asterisk/fuentesip/

tesm:/etc/asterisk/fuentesip# sh registrar.sh 1050 4001
3002    192.168.1.1
```

- **Central telefónica Esmeraldas**

Se observa en la línea de comandos del servidor Asterisk en Esmeraldas, los *logs* correspondientes a este proceso.

```
root@servidor-desktop:/usr/share/vnuml/examples# ssh
esmesm:~#asterisk-rvvvvvvvvvvvv
```

```
Asterisk 1.4.30, Copyright (C) 1999 - 2010 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
  == Parsing '/etc/asterisk/asterisk.conf': Found
  == Parsing '/etc/asterisk/extconfig.conf': Found
Connected to Asterisk 1.4.30 currently running on esm (pid = 2574)
Verbosity was 45 and is now 47
    -- Remote UNIX connection
esm*CLI>
esm*CLI>
   -- Added extension '1050' priority 1 to registrosip
```

Figura 3.49 Extensión añadida a la central telefónica de Esmeraldas

- **Host cliente tloj**

  - Ingresar a la máquina virtual.

```
root@servidor-desktop:/usr/share/vnuml/examples# sshtloj
```

```
root@servidor-desktop:/usr/share/vnuml/examples# ssh tesm
Linux vm 2.6.31.5 #4 Fri Oct 23 10:26:16 BST 2009 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jul 30 01:23:31 2010 from 192.168.0.133
tesm:~#
```

Figura 3.50 Inicio de la máquina virtual TLOJ

  - Ingresar al directorio que contiene los scripts de ejecución .sh y ejecutarlo.

```
tloj:~# cd /etc/asterisk/fuentesip/

tloj:/etc/asterisk/fuentesip# sh registrar.sh 9050 4002
3001 172.31.15.1
```

- **Central telefónica Loja**

Se observa en la línea de comandos del servidor Asterisk en Loja, los logs correspondientes a este proceso.

```
root@servidor-desktop:/usr/share/vnuml/examples# ssh loj
```

```
root@servidor-desktop:/home/servidor# ssh loj
Linux vm 2.6.31.5 #4 Fri Oct 23 10:26:16 BST 2009 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jul 30 18:16:47 2010 from 192.168.0.129
loj:~#
```

Figura 3.51 Inicio de la central telefónica de Loja

```
loj:~#asterisk-rvvvvvvvvvvvvv
```

```
Asterisk 1.4.30, Copyright (C) 1999 - 2010 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=========================================================================
  == Parsing '/etc/asterisk/asterisk.conf': Found
  == Parsing '/etc/asterisk/extconfig.conf': Found
Connected to Asterisk 1.4.30 currently running on loj (pid = 2573)
Verbosity is at least 23
    -- Remote UNIX connection
    -- Added extension '9050' priority 1 to registrosip
    -- Registered SIP '9050' at 172.31.15.4 port 4002
loj*CLI>
```

Figura 3.52Extensión añadida a la central telefónica de Loja

Los resultados de las figuras 3.49 y 3.52 muestran el éxito del proceso de registro de las extensiones 1050 y 9050 en las centrales telefónicas de Esmeraldas y Loja respectivamente.

3.4.2 BÚSQUEDA DE EXTENSIONES ESTÁTICAS EN EL DOMINIO DUNDi ecuador

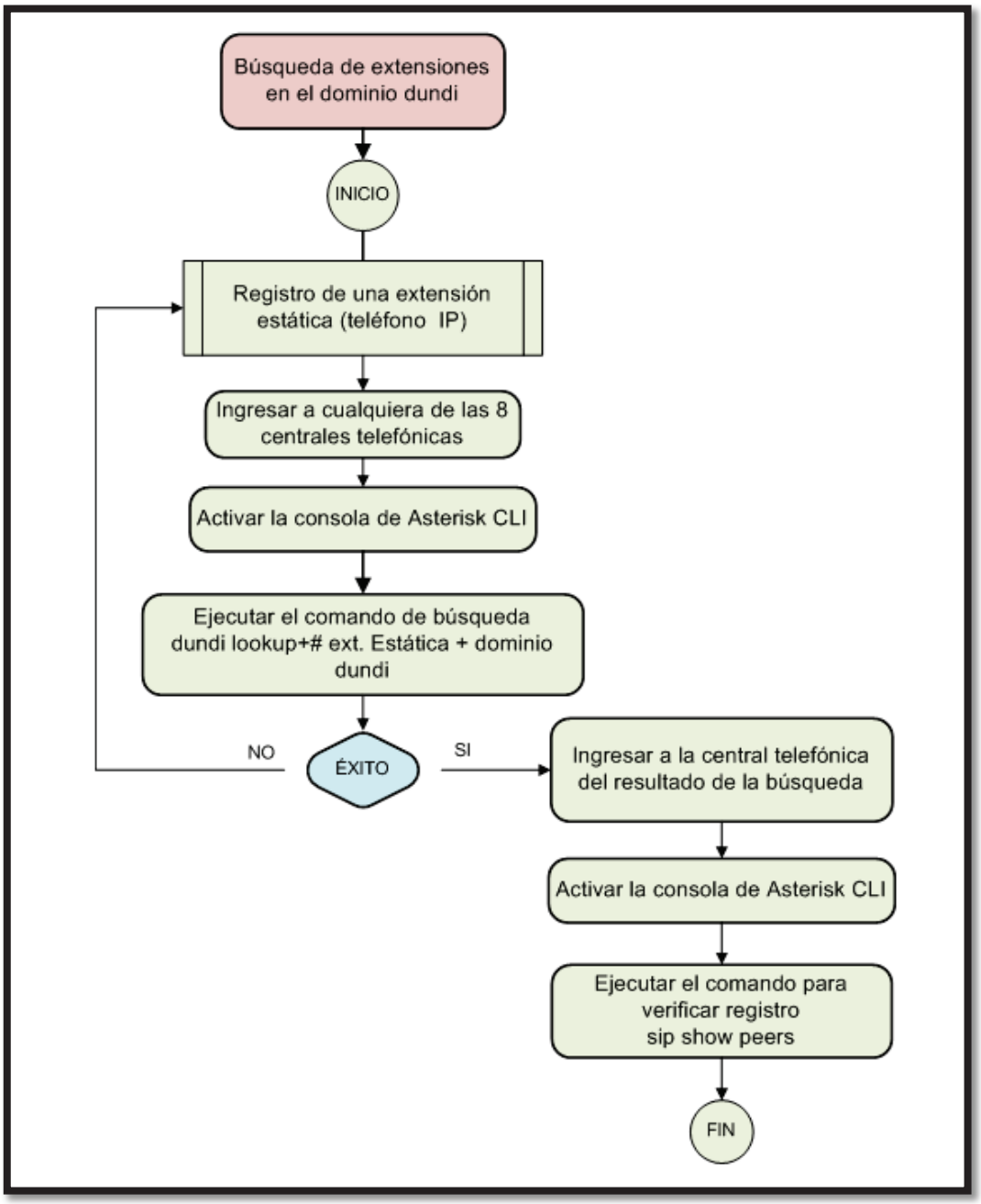


Figura 3.53 Diagrama de flujo del proceso de búsqueda de extensiones estáticas en el dominio *DUNDI*

De acuerdo al diagrama de flujo de la figura 3.53, registrada la extensión 1050 en el nodo de Esmeraldas desde el cliente respectivo (teléfono TESM), se verifica el funcionamiento de las consultas punto a punto del dominio *DUND*i, desde la central telefónica en Loja (LOJ). Para este efecto se utiliza el comando dundi lookup. El comando dundi lookup desde la central en Loja debería permitirnos determinar exactamente la ubicación de la extensión 1050 en el dominio *DUNDi*ecuador.

- **Central telefónica Loja**

```
loj*CLI>dundi lookup 1050@ecuador
```

```
loj*CLI> dundi lookup 1050@ecuador
  1.     0 IAX2/dundi:8eUkSSnzI1QHO60cGRqgKg==@10.10.1.1/1050 (EXISTS|CANMATCH)
     from fe:fd:00:00:01:00, expires in 2 s
DUNDi lookup completed in 3725 ms
loj*CLI>
```

Figura 3.54.1 Resultados del comando dundi lookup

Una vez registrada la extensión 9050 en el nodo de Loja desde el cliente respectivo (teléfono TLOJ), se verifica el funcionamiento de las consultas punto a punto del dominio DUNDI, desde la central telefónica en Esmeraldas (ESM), Guayaquil (GYE) y Ambato (AMB). El comando dundi lookup desde la central en Esmeraldas, Guayaquil y Ambato nos permitirá determinar la ubicación de la extensión 9050 en el dominio *DUNDi denominado* ecuador.

- **Central telefónica Esmeraldas**

```
esm*CLI>dundi lookup 9050@ecuador
```

```
esm*CLI> dundi lookup 9050@ecuador
  1.     0 IAX2/dundi:ro4yY8x3gg9FR+EJvCAuPw==@10.10.9.2/9050 (EXISTS|CANMATCH)
     from fe:fd:00:00:08:00, expires in 2 s
DUNDi lookup completed in 2125 ms
esm*CLI>
```

Figura 3.54.2 Resultados del comando dundi lookup en ESM

- **Central telefónica Guayaquil**

```
gye*CLI> dundi lookup 9050@ecuador
```

```
gye*CLI> dundi lookup 9050@ecuador
  1.     0 IAX2/dundi:a0fcPPEwJTNBbdJbLTy8JA==@10.10.9.2/9050 (EXISTS|CANMATCH)
     from fe:fd:00:00:08:00, expires in 2 s
DUNDi lookup completed in 1974 ms
gye*CLI>
```

Figura 3.54.3 Resultados del comando dundi lookup en GYE

- **Central telefónica Ambato**

```
amb*CLI> dundi lookup 9050@ecuador
```

```
amb*CLI> dundi lookup 9050@ecuador
  1.     0 IAX2/dundi:a0fcPPEwJTNBbdJbLTy8JA==@10.10.9.2/9050 (EXISTS|CANMATCH)
     from fe:fd:00:00:08:00, expires in 2 s
DUNDi lookup completed in 1897 ms
amb*CLI>
```

Figura 3.54.4 Resultados del comando dundi lookup

Como se puede comprobar en ambos casos, las extensiones 1050 y 9050 registradas previamente en las centrales telefónicas de Esmeraldas y Loja son localizadas sin ningún inconveniente desde otras centrales telefónicas en el dominio DUNDiecuador, con lo cual queda demostrado el proceso de búsqueda.

3.4.3 PROCESO DE LLAMADA Y ESPERAR LLAMADA ENTRE EXTENSIONES REGISTRADAS

Los procesos de llamar y esperar llamada son ejecutados a la par en las instancias SIP (teléfonos IP) involucrados en el establecimiento de llamada. Es decir un usuario del sistema de telefonía IP después de enrolar su número de extensión personal debe activar el servicio de esperar llamada con el fin de ser alcanzado por los otros usuarios en el sistema.

En el siguiente diagrama de flujo se detallan el funcionamiento de ambos procesos y posteriormente se realiza un ejemplo práctico.

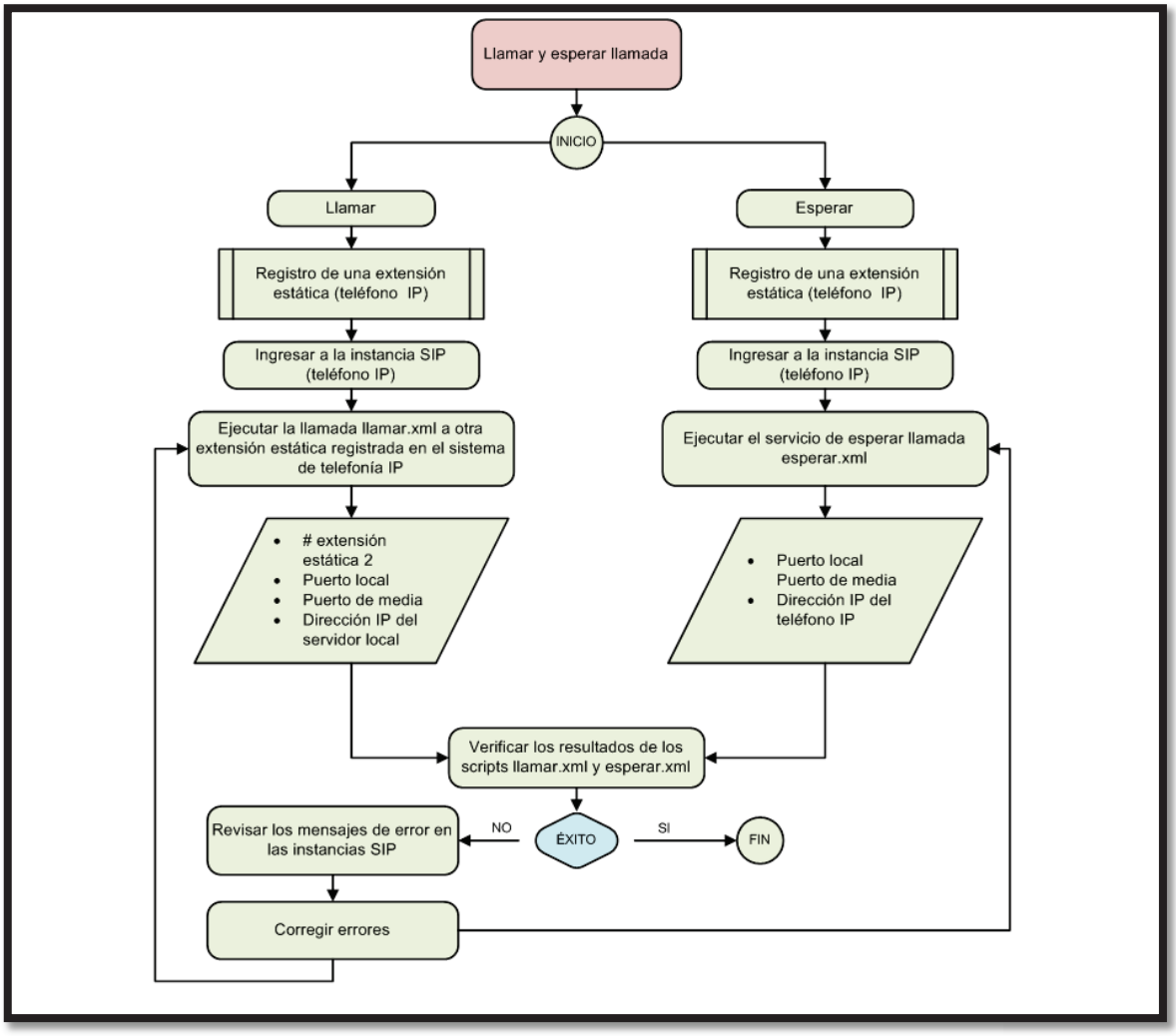


Figura 3.55 Diagrama de flujo del proceso de llamar y esperar llamada entre extensiones estáticas

Registradas las extensiones 1050 y 9050 en el sistema de telefonía IP, se procede a realizar la prueba de llamadas en el sistema. La llamada será realizada desde el cliente TESM hacia el cliente TLOJ.

Dado este escenario se inicia el script esperar.sh en el nodo que simula el teléfono TLOJ y se verifica que el puerto en el que escucha el servicio iniciado corresponda con el indicado en el proceso de registro. Para esto se empleará el comando netstat -a.

- **Host cliente TLOJ**

```
tloj:/etc/asterisk/fuentesip# sh esperar.sh 4002 3001
172.31.15.4
```

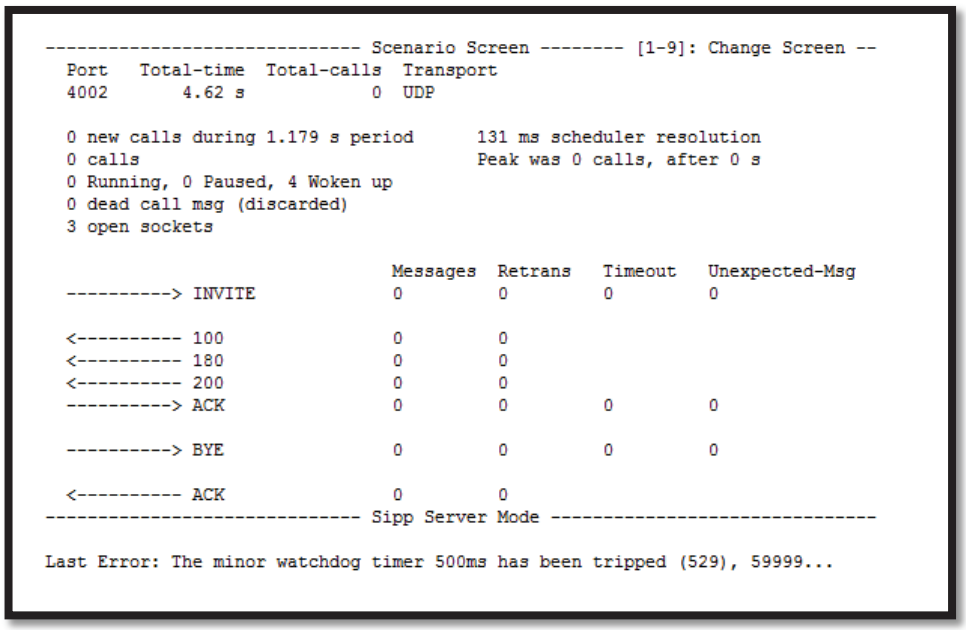


Figura 3.56 Escenario esperar.xml

Como se puede observar en el gráfico 3.56 se inicia el escenario esperar.xml en el cliente TLOJ. El cliente TLOJ espera la llegada de un mensaje *INVITE* para iniciar el proceso. En la columna Messages se indica el éxito de envío y recepción de mensajes, al iniciar el escenario esta columna presentará cero, después al efectuarse la llamada deberá cambiar de valor a uno para indicar que el proceso ha llegado a su fin.

```
tloj # netsat -a |grep 4002
```

Figura 3.57 Verificación de puerto para esperar llamada

En el cliente TESM se ejecuta el script de inicio de llamada denominado llamar.sh desde la máquina que simula el teléfono en Esmeraldas, TESM, con destino la extensión registrada en Loja, desde el nodo cliente.

- **Host cliente TESM**

```
tesm:/etc/asterisk/fuentesip# sh llamar.sh 9050 4003 3003
192.168.1.1
```

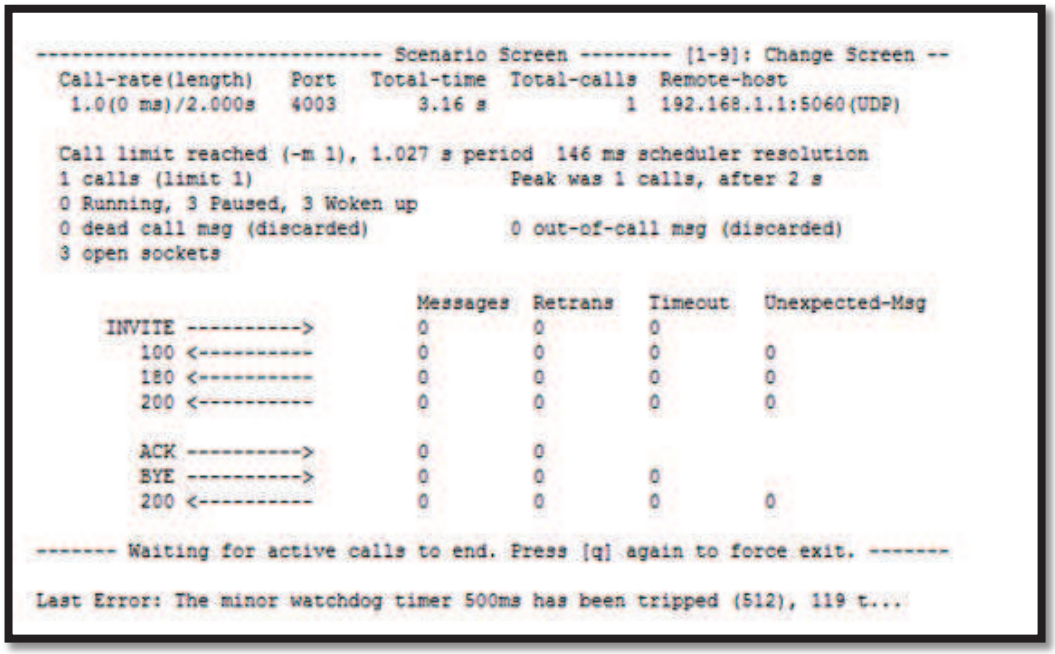


Figura 3.58 Inicio del escenario de llamada en TESM

Se verifica el éxito de la llamada en los hosts clientes TESM y TLOJ al igual que en las correspondientes centrales de telefonía IP, revisando los resultados de los scripts llamar.xml y esperar.xml.

- **Host cliente TLOJ**

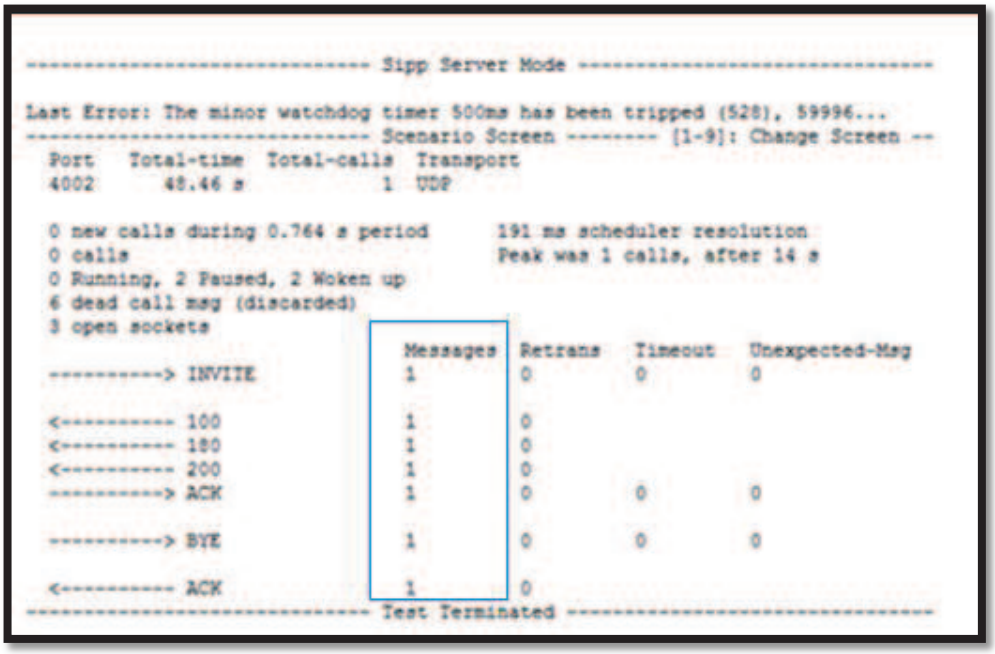


Figura 3.59.1 Verificación de llamada

- **Host cliente TESM**

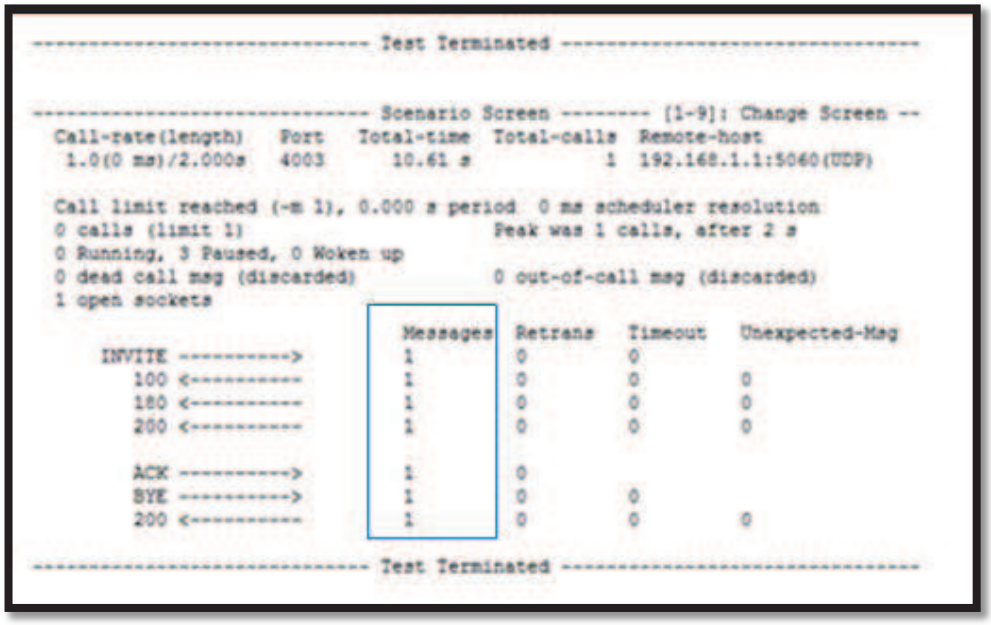


Figura 3.59.2 Verificación de llamada

- **Central Telefónica Loja**

```
    -- Remote UNIX connection

    -- Accepting AUTHENTICATED call from 10.10.1.1:

       > requested format = ulaw,

       > requested prefs = (),

       > actual format = ulaw,

       > host prefs = (ulaw|alaw),

       > priority = mine

    -- Executing [9050@registrosip:1] NoOp("IAX2/dundi-3798", "9050") in new
stack

    -- Executing [9050@registrosip:2] Dial("IAX2/dundi-3798", "SIP/9050|10") in
new stack

    -- Called 9050

    -- SIP/9050-00000002 is ringing

    -- SIP/9050-00000002 answered IAX2/dundi-3798

  == Spawn extension (registrosip, 9050, 2) exited non-zero on 'IAX2/dundi-3798'

    -- Hungup 'IAX2/dundi-3798'
```

Figura 3.59.3 Verificación de llamada

- **Central telefónica Esmeraldas**

```
  -- Remote UNIX connection

  -- Executing [9050@local:1] Dial("SIP/1050-00000002", "SIP/9050") in new stack

  [Jul 30 20:44:16] WARNING[2706]: chan_sip.c:3094 create_addr: No such host: 9050

  [Jul 30 20:44:16] WARNING[2706]: app_dial.c:1296 dial_exec_full: Unable to create
  channel of type 'SIP' (cause 20 - Unknown)

  == Everyone is busy/congested at this time (1:0/0/1)

  -- Executing [9050@local:2] System("SIP/1050-00000002", "echo $(date) La
  extension 9050 no se encuentra localmente registrada >>
  /etc/asterisk/fuentesip/redireccion.log") in new stack

  -- Executing [9050@local:3] System("SIP/1050-00000002", "echo $(date) Iniciando
  BUSQUEDA en el dominio DUNDI >> /etc/asterisk/fuentesip/redireccion.log") in new
  stack

  -- Executing [9050@local:4] Macro("SIP/1050-00000002", "buscar-dundi|9050") in
  new stack

  -- Executing [s@macro-buscar-dundi:1] Goto("SIP/1050-00000002", "9050|1") in new
  stack

  -- Goto (macro-buscar-dundi,9050,1)

  -- Called dundi:aOfcPPEwJTNBbdJbLTy8JA==@10.10.9.2/9050

  -- Call accepted by 10.10.9.2 (format ulaw)
```

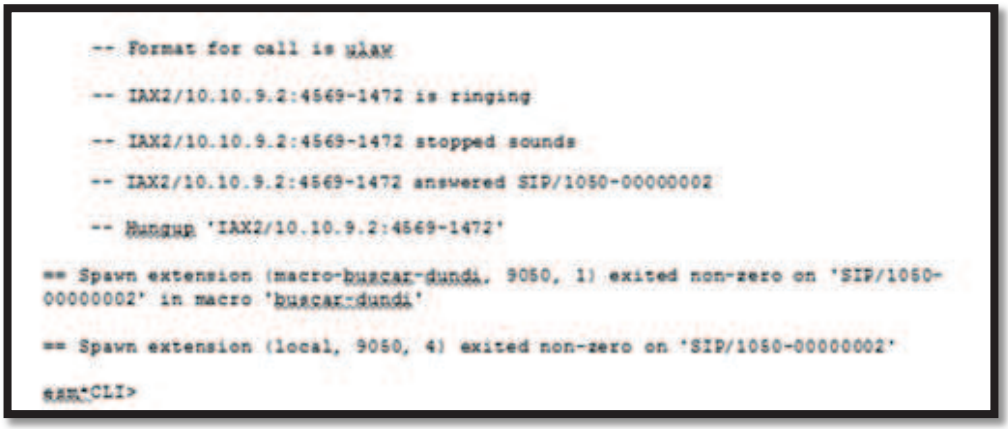


Figura 3.59.4 Verificación de llamada

Se obtiene como resultado éxito en el establecimiento de la llamada desde el usuario TESM al usuario TLOS tal como lo demuestran las ventanas de las figuras 3.59.1, 3.59.2, 3.59.3, 3.59.4.

3.4.4 PROCESO DE ENROLAMIENTO DE NÚMEROS DE EXTENSIÓN PERSONALES (MÓVILES)

Para llevar a cabo este proceso se sigue el diagrama de flujo de la de la figura 3.61. Desde el teléfono TESM se enrola la extensión personal 1150 en la central telefónica ESM, usando el script llamar.sh.

```
tesm:/etc/asterisk/fuentesip# sh llamar.sh  4561150 4100 3100
192.168.1.1 1050
```

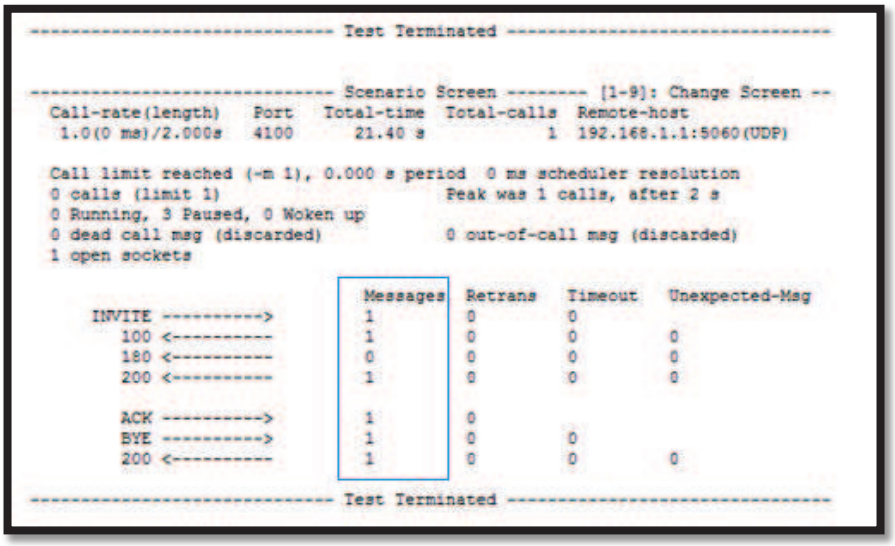


Figura 3.60 Proceso de enrolamiento en TESM

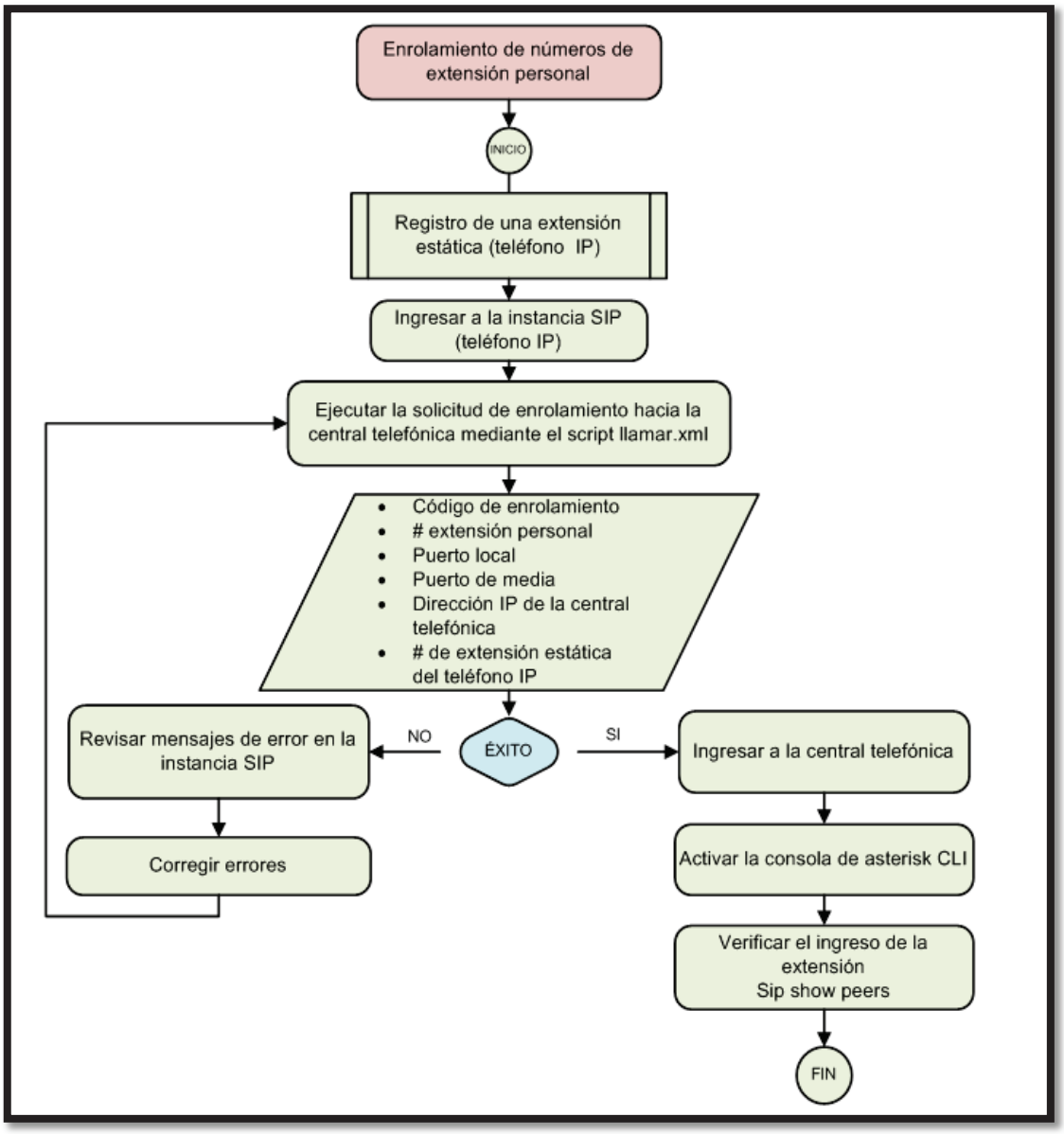


Figura 3.61 Diagrama de flujo del proceso enrolamiento de números de extensión personales

Desde el teléfono TESM se enrola la extensión personal 1150 en la central telefónica ESM, usando el script llamar.sh.

```
tesm:/etc/asterisk/fuentesip# sh llamar.sh  4561150 4100 3100
192.168.1.1 1050
```

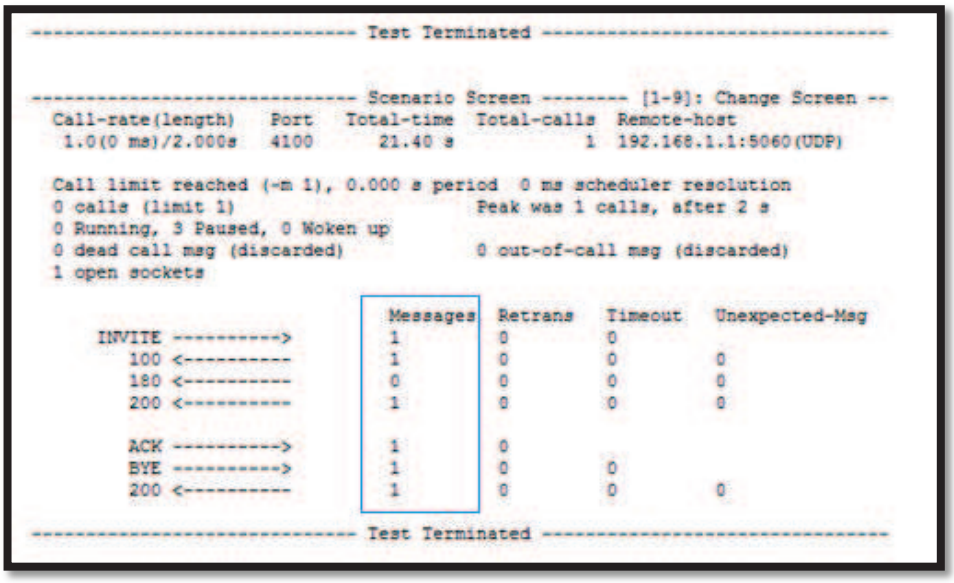


Figura 3.62 Proceso de enrolamiento en TESM

Se verifica, mediante registros de la CLI de Asterisk en la central telefónica de Esmeraldas, que el proceso se haya cumplido.

```
esm*CLI> sip show
```

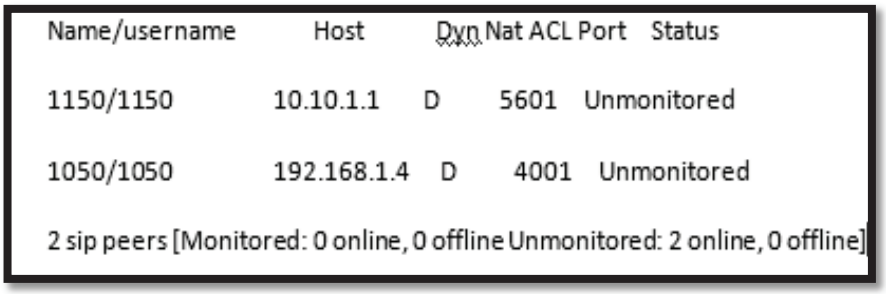


Figura 3.63 Verificación del proceso de enrolamiento TESM

Desde el teléfono TLOJ se enrola (llamar.sh) la extensión personal 9150 en la central LOJ y se realiza la verificación respectiva.

```
tloj:/etc/asterisk/fuentesip# sh llamar.sh  4569150 4200 3200
172.31.15.1 9500
```

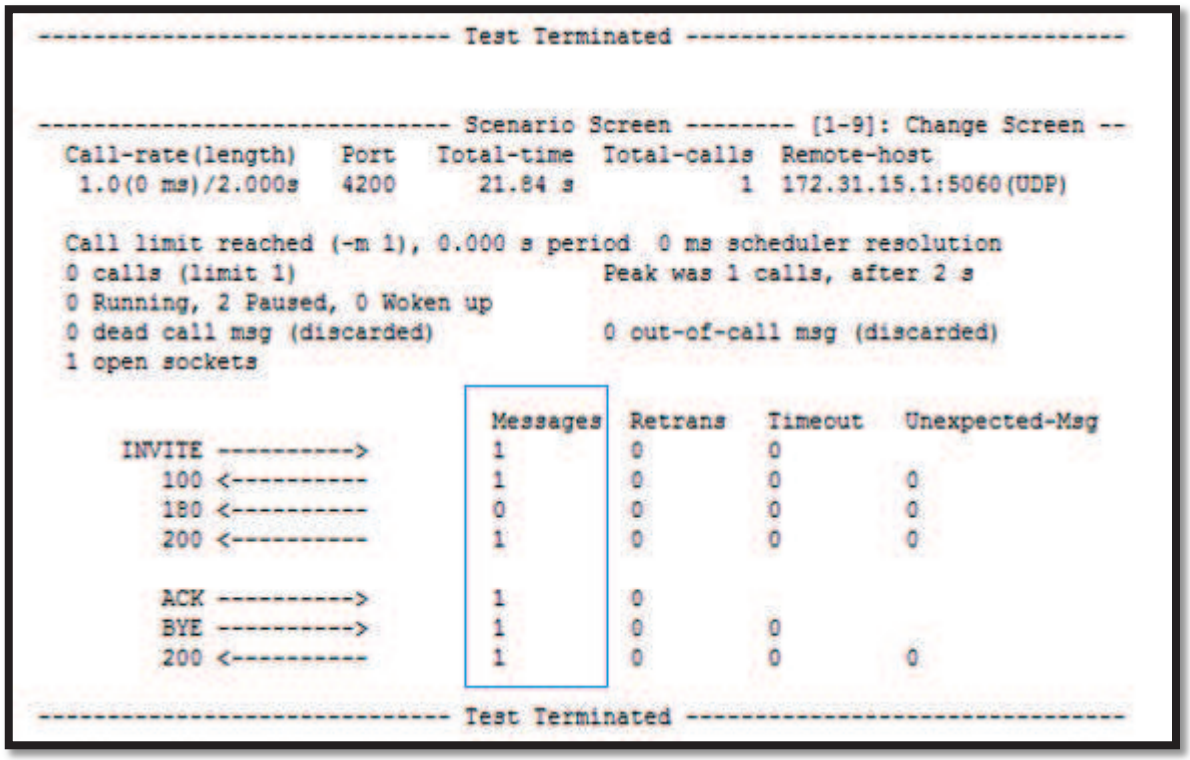


Figura 3.64 Proceso de enrolamiento en TLOJ

Se verifica, mediante registros de la CLI de Asterisk en la central telefónica de Loja, que el proceso se haya cumplido.

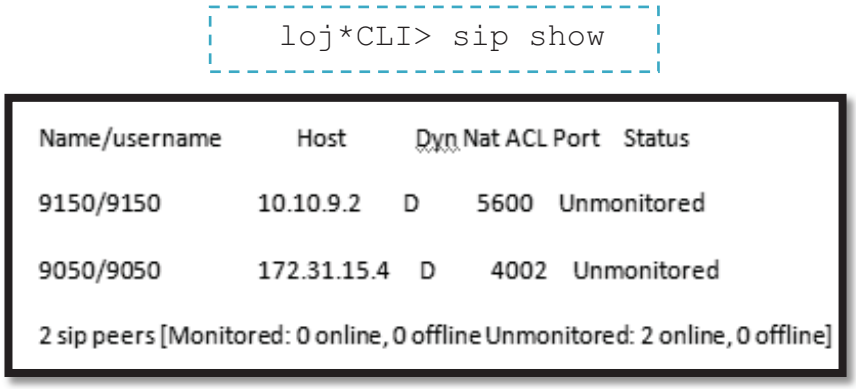


Figura 3.65 Verificación del proceso de enrolamiento TLOJ

3.4.5 PROCESO DE LLAMADA Y ESPERA DE LLAMADA ENTRE EXTENSIONES ENROLADAS

El proceso de llamada entre dos extensiones personales se detalla en el diagrama de flujo de la figura 3.66.

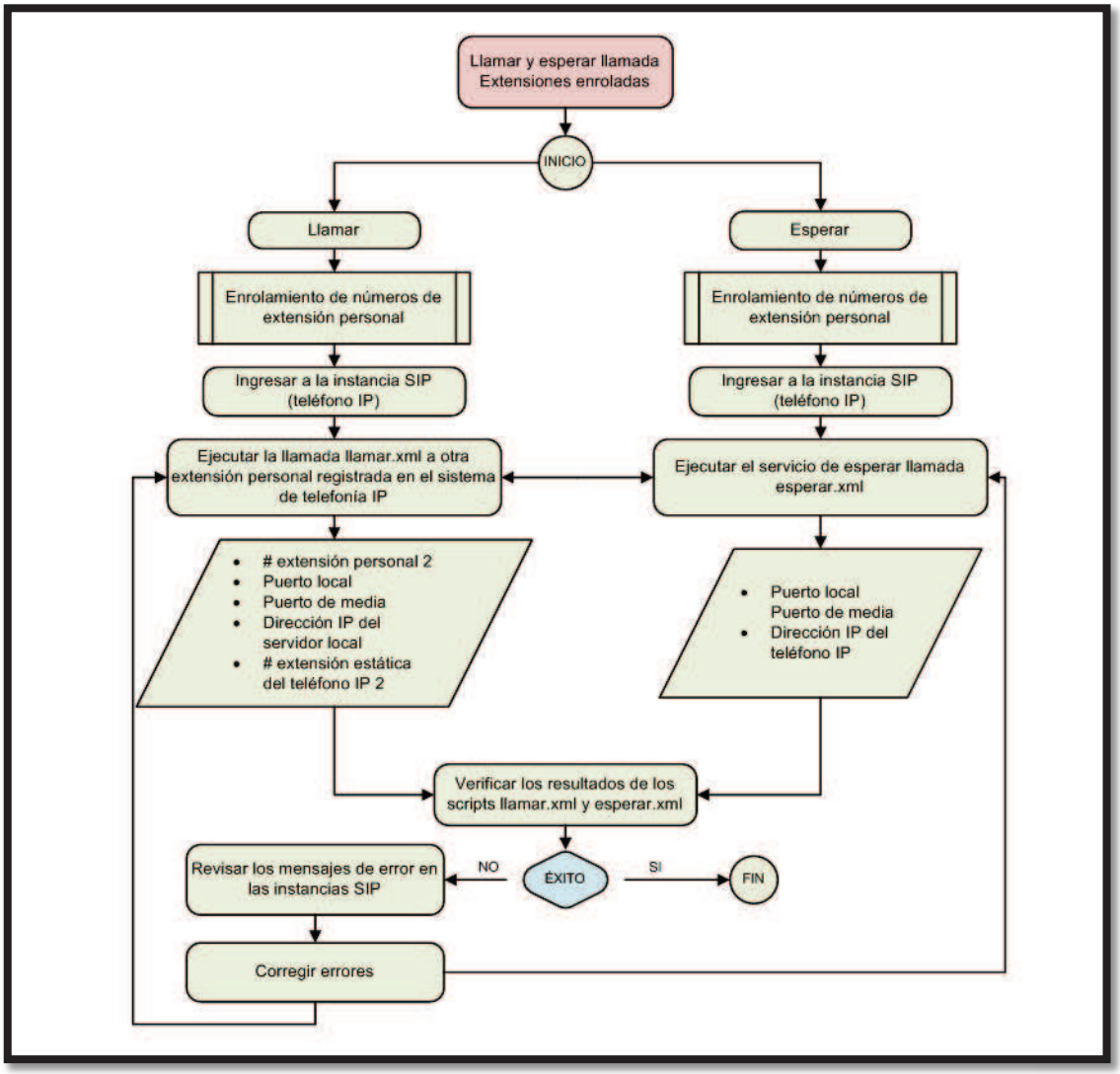


Figura 3.66 Diagrama de flujo del proceso llamar y esperar llamada entre extensiones personales enroladas

Conforme al diagrama, enrolados los números de extensión personal 1150 y 9150 en las centrales telefónicas de Esmeraldas y Loja respectivamente, se procede a comprobar la realización de llamada entre las dos extensiones. La llamada será realizada desde la extensión personal 1150 hacia la extensión personal 9150.

- **Host cliente TLOJ**

Se inicia el script de espera de llamada esperar.shen el teléfono TLOJ.

```
tloj:/etc/asterisk/fuentesip# sh esperar.sh 4002 3001
172.31.15.4
```

- **Host cliente tesm**

Se ejecuta el script de llamada desde el teléfono TESM con destino la extensión 9150 y se verifica que la comunicación llegue a su destino en la central LOJ.

```
tesm:/etc/asterisk/fuentesip# sh llamar.sh 9150 4300 3300
192.168.1.1 9050
```

Se verifica el éxito de la llamada en los resultados de los escenarios xml de cada cliente.

- **Host cliente TLOJ**

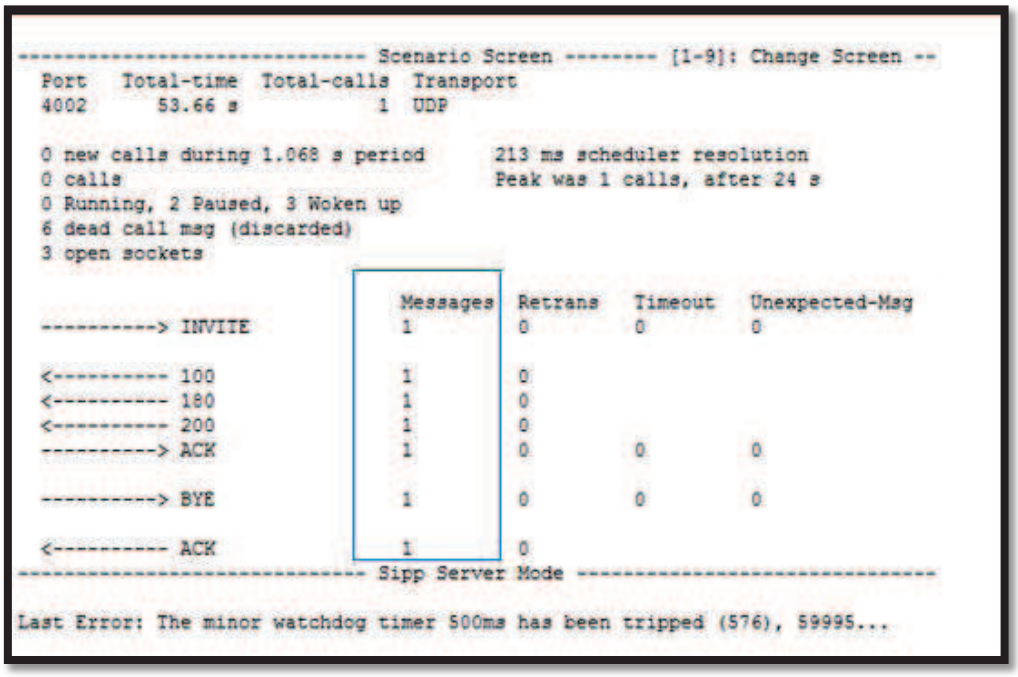


Figura 3.67 Llamadas entre extensiones enroladas TLOJ

- **Host cliente TESM**

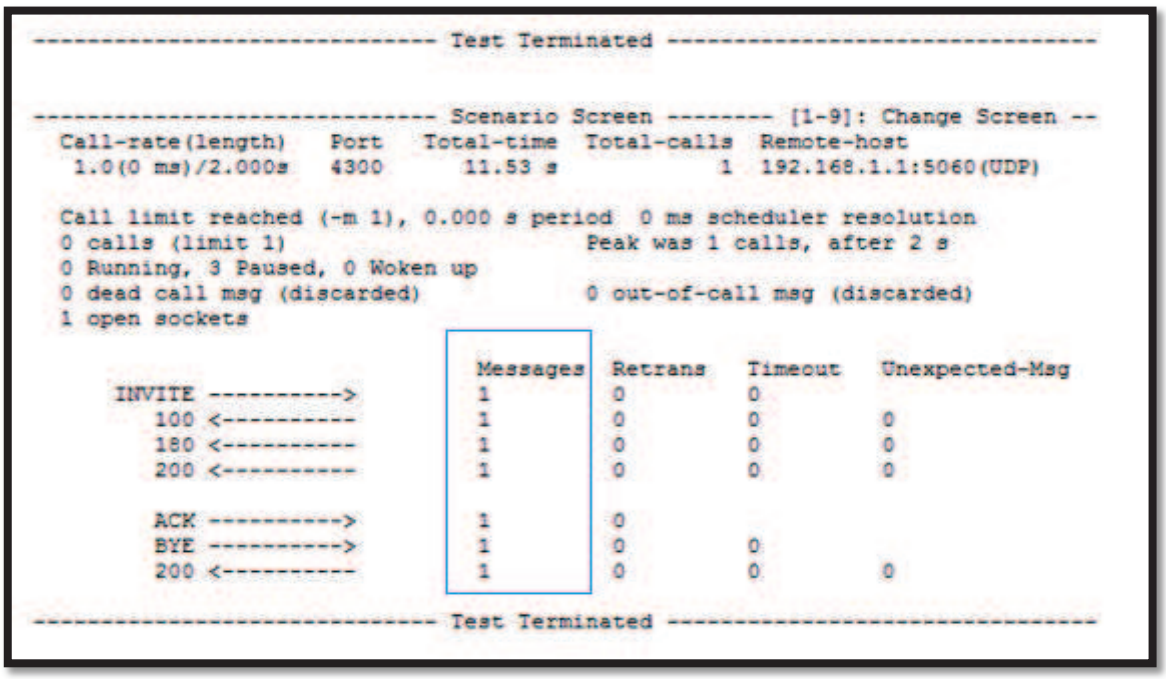


Figura 3.68 Llamadas entre extensiones enroladas TESM

Como resultado se obtiene el establecimiento de la llamada entre las extensiones personales 1150 y 9150 tal como lo demuestran las figuras 3.67 y 3.68, las columnas Messages en cada escenario toman los valores uno mostrando la correcta recepción de la petición de llamada.

3.4.6 PROCESO DE DES-ENROLAMIENTO DE NÚMEROS DE EXTENSIÓN PERSONALES

Siguiendo el diagrama de flujo de la figura 3.69 se realizan los siguientes pasos para llevar a cabo el des-enrolamiento una extensión personal en el sistema de telefonía IP. Los números de extensión personales a des-enrolar son 1150 y 9150, los cuales han sido previamente enrolados en el sistema.

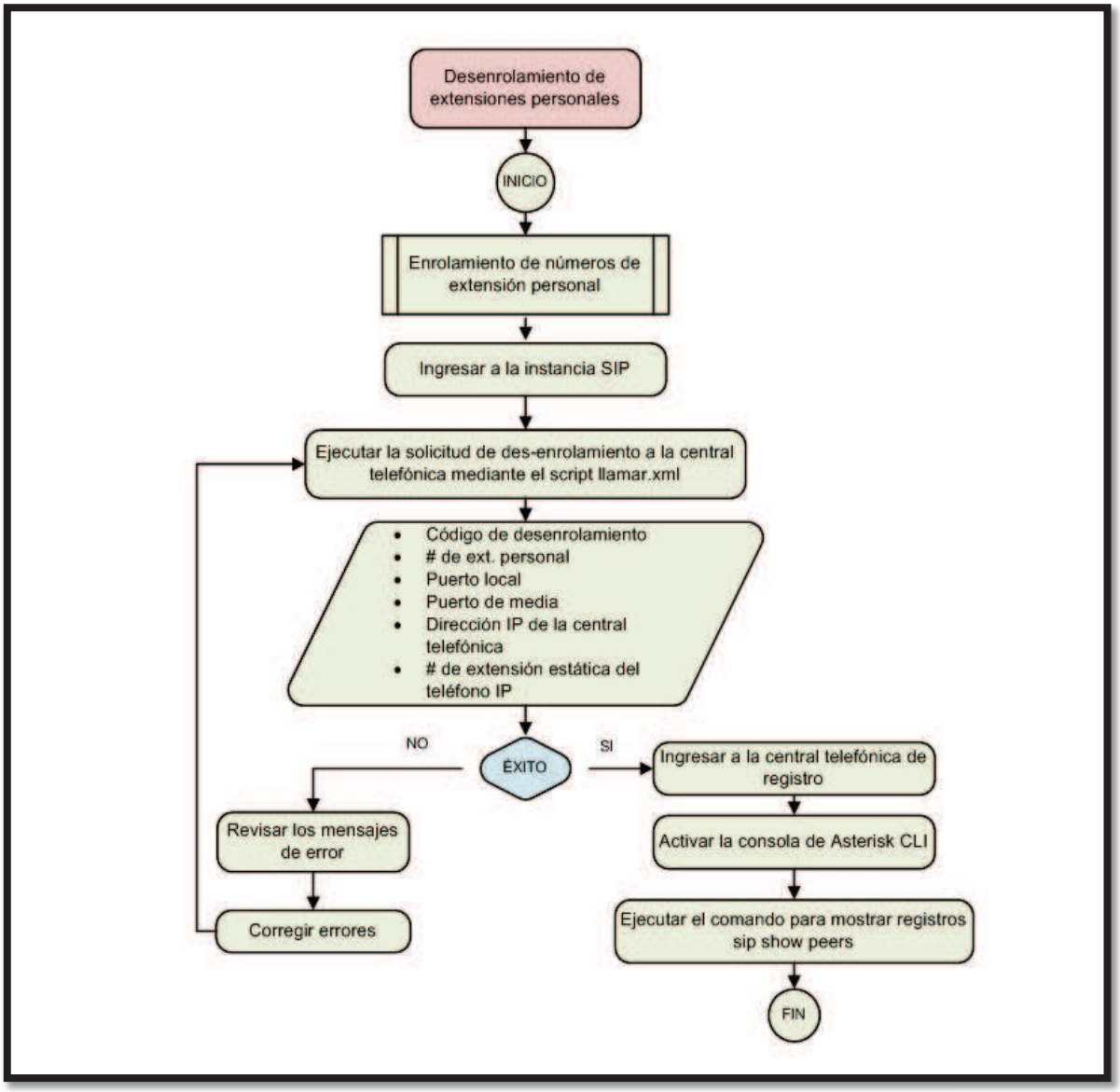


Figura 3.69 Diagrama de flujo del proceso de des-enrolamiento de números de extensión personal

Desde el cliente TESM se ejecuta la llamada de des-enrolamiento, utilizando el código correspondiente en el script de llamada en este caso **123**.

- **Host cliente TESM**

```
tesm:/etc/asterisk/fuentesip# sh llamar.sh 1231150 4500 3500
192.168.1.1 1050
```

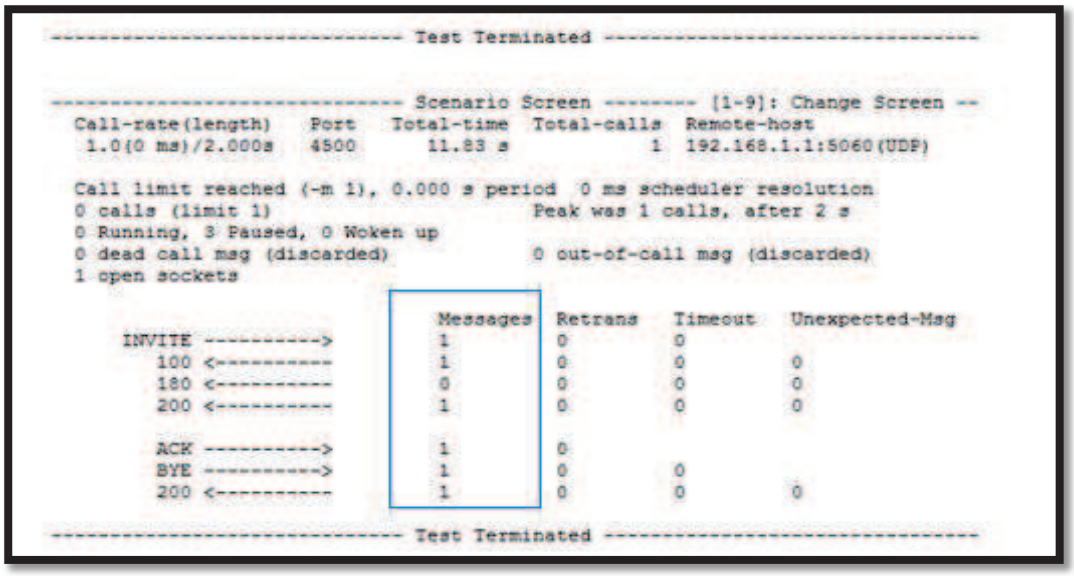


Figura 3.70 Proceso de des-enrolamiento en TESM

- **Central telefónica Esmeraldas**

Se verifica el proceso de des-enrolamiento de la extensión personal mediante el monitoreo de los logs de Asterisk en el servidor.

```
esm*CLI> sip show
```

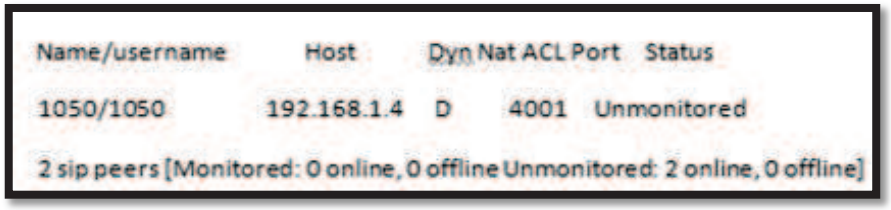


Figura 3.71 Extensión personal des-enrolada en ESM

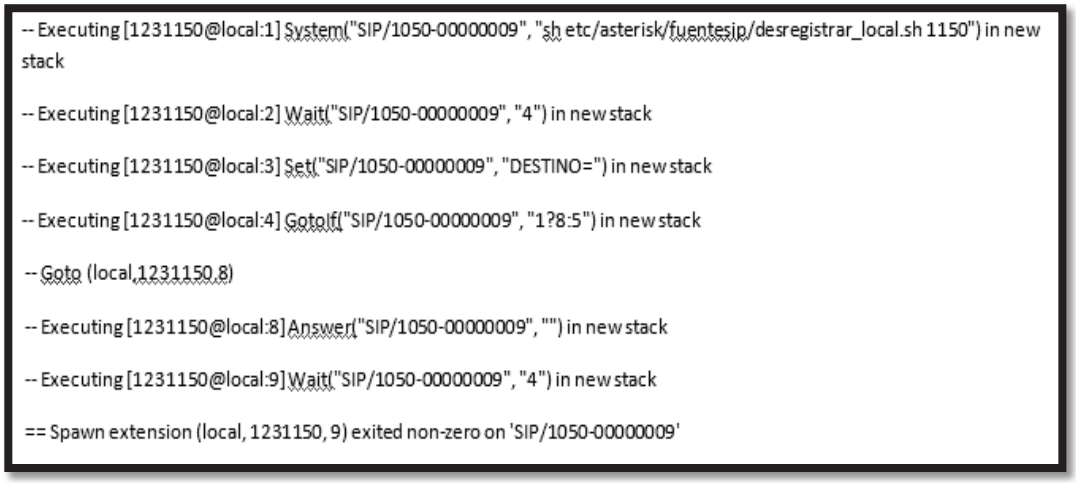


Figura 3.72 Proceso de des-enrolamiento en ESM

- **Host cliente TLOJ**

Desde el cliente TLOJ se ejecuta la llamada de des-enrolamiento, utilizando el código correspondiente en el script de llamada. En este caso la extensión persona la des-enrolar es la 9150, la cual está ingresada en la central telefónica de Loja.

```
tloj:/etc/asterisk/fuentesip# sh llamar.sh 1239150
                4600 3600 172.31.15.1 9050
```

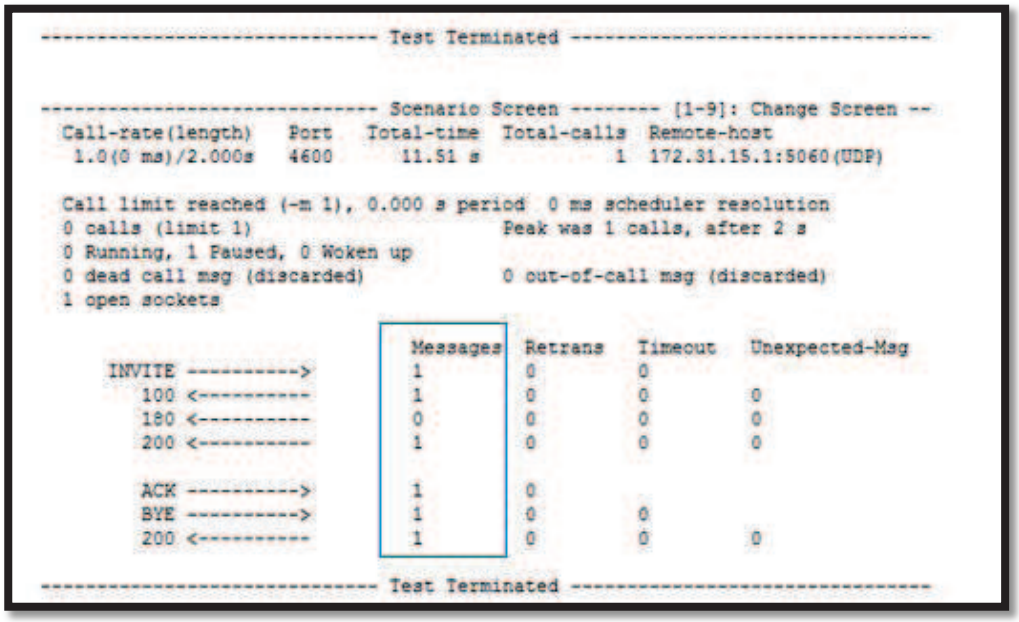


Figura 3.73 Proceso de des-enrolamiento en TLOJ

- **Central telefónica Loja**

Se comprueba el des-enrolamiento de la extensión 9150 en el dial-plan ingresado al CLI de asterisk en la central telefónica.

```
loj*CLI> sip show peers
```

```
Name/username        Host        Dyn Nat ACL Port    Status
9050/9050            172.31.15.4  D      4002    Unmonitored
1 sip peers [Monitored: 0 online, 0 offline Unmonitored: 1 online, 0 offline]
```

Figura 3.74Extensión personal des-enrolada en LOJ

Se verifica el proceso mediante el monitoreo de los logs de Asterisk en la central telefónica conectada al cliente.

```
-- Executing [1239150@local:1] System("SIP/9050-00000004", "sh /etc/asterisk/fuentesip/desregistrar_local.sh 9150") in new stack
  -- Executing [1239150@local:2] Wait("SIP/9050-00000004", "4") in new stack
  -- Executing [1239150@local:3] Set("SIP/9050-00000004", "DESTINO=") in new stack
  -- Executing [1239150@local:4] GotoIf("SIP/9050-00000004", "1?8:5") in new stack
  -- Goto (local,1239150,8)
  -- Executing [1239150@local:8] Answer("SIP/9050-00000004", "") in new stack
  -- Executing [1239150@local:9] Wait("SIP/9050-00000004", "4") in new stack
 == Spawn extension (local, 1239150, 9) exited non-zero on 'SIP/9050-00000004'
```

Figura 3.75 Proceso de des-enrolamiento en LOJ

3.5 VERIFICACIÓN DE FUNCIONALIDAD DE RECUPERACIÓN DEL SISTEMA EN CASO DE FALLA EN LA RED

3.5.1 FALLA DE UN ENLACE INTERMEDIO

Tomando en cuenta la comunicación entre los teléfonos TESM y TLOJ en el esquema, a través de los nodos ESM y LOJ y la red mallada, es comprensible que la ruta que toma esta comunicación incluye el nodo GYE en su interfaz eth1 (existen menos saltos al destino). Eso efectivamente es lo que refleja el comando ***traceroute*** desde ESM a LOJ.

- **Central telefónica ESM**

```
esm:~# traceroute 10.10.9.2
```

```
esm:~# traceroute 10.10.9.2
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.3.2)  67.284 ms  67.136 ms  67.017 ms
 2   (10.10.6.2)  66.900 ms  66.781 ms  66.661 ms
 3   (10.10.9.2)  129.033 ms  128.914 ms  128.794 ms
```

Figura 3.76 Ruta desde Esmeraldas a Loja

Para simular la falla de un enlace intermedio en la malla, en este caso la red 3, se deshabilitará dicha red en el nodo GYE. Se termina el proceso correspondiente a la red 3 desde el sistema *host anfitrión* (Máquina física).

- **Máquina física**

  - Se busca el PID del proceso correspondiente a la red 3.

```
root@servidor-desktop:/home/servidor# ps -aux | grep Net
```

```
ks/Net 2.ctl
root     20587  0.0  0.0   1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 3.ctl
root     20624  0.0  0.0   1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 4.ctl
root     20661  0.0  0.0   1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 5.ctl
root     20702  0.0  0.0   1672   464 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 6.ctl
root     20705  0.0  0.0   1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 7.ctl
```

Figura 3.77 Procesos del esquema de simulación red 3

- Se mata el proceso de red3 correspondiente a 2904

```
root@servidor-desktop:/home/servidor# kill -9 20587
```

Terminado el proceso se verifican los resultados

- **<u>Central telefónica Esmeraldas</u>**

  La central telefónica de Esmeraldas se conecta directamente con la central de Guayaquil por medio de la red 3, al fallar el enlace no debe haber conectividad.

```
esm:~# ping  10.10.3.2
```

```
esm:~# ping 10.10.3.2
PING 10.10.3.2 (10.10.3.2) 56(84) bytes of data.
From 10.10.3.1 icmp_seq=1 Destination Host Unreachable
From 10.10.3.1 icmp_seq=2 Destination Host Unreachable
From 10.10.3.1 icmp_seq=3 Destination Host Unreachable
From 10.10.3.1 icmp_seq=7 Destination Host Unreachable
```

Figura 3.78 Conectividad entre UIO y GYE

A pesar de fallar un enlace intermedio la central telefónica de Esmeraldas se puede comunicar con la central de Loja debido ya que existe redundancia en el sistema.

```
esm:~# ping 10.10.9.2
```

```
esm:~# ping 10.10.9.2

PING 10.10.9.2 (10.10.9.2) 56(84) bytes of data.

64 bytes from 10.10.9.2: icmp_seq=1 ttl=60 time=361 ms

64 bytes from 10.10.9.2: icmp_seq=2 ttl=60 time=225 ms

64 bytes from 10.10.9.2: icmp_seq=3 ttl=60 time=383 ms

64 bytes from 10.10.9.2: icmp_seq=4 ttl=60 time=239 ms

64 bytes from 10.10.9.2: icmp_seq=5 ttl=60 time=330 ms
```

Figura 3.79 Conectividad entre ESM y LOJ

La ruta alterna para llegar desde Esmeraldas a Loja es la siguiente:

```
esm:~# traceroute 10.10.9.2
```

```
esm:~# traceroute 10.10.9.2
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.1.2)  68.245 ms  68.096 ms  67.964 ms
 2   (10.10.11.1)  67.841 ms  67.722 ms  67.600 ms
 3   (10.10.6.2)  168.206 ms  168.092 ms  167.970 ms
 4   (10.10.9.2)  167.854 ms  167.732 ms  96.796 ms
```

Figura 3.80 Ruta alterna desde ESM a LOJ

Los pasos a ejecutarse para esta prueba serán:

- Registrar el teléfono TESM en ESM.

```
tesm:/etc/asterisk/fuentesip#  sh  registrar.sh  1050  4001
3002 192.168.1.1
```

- Registrar el teléfono TLOJ en LOJ.

```
tloj:# sh registrar.sh 9050 4002 3001 172.31.15.1
```

- Enrolar el número 1150 desde TESM en la central ESM.

```
tesm:# sh llamar.sh  4561150 4100 3100 192.168.1.1 1050
```

- Enrolar el número 9150 desde TLOJ en la central LOJ.

```
tloj:# sh llamar.sh 4569150 4200 3200 172.31.15.1 9050
```

- Verificación de resultados

  - **Central telefónica Esmeraldas**

```
esm*CLI> sip show peers
Name/username        Host        Dyn Nat ACL Port    Status
1150/1150            10.10.1.1    D       5600    Unmonitored
1050/1050            192.168.1.4  D       4001    Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.81 Resultados de registro y enrolamiento en ESM

  - **Central telefónica Loja**

```
esm*CLI> sip show peers
Name/username        Host        Dyn Nat ACL Port    Status
1150/1150            10.10.1.1    D       5600    Unmonitored
1050/1050            192.168.1.4  D       4001    Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.82 Resultados de registro y enrolamiento en LOJ

- Llamada desde TESM a TLOJ, para ello se ejecuta el script de esperar llamada en TLOJ y el script de llamar en TESM con los respectivos datos.

- **Host cliente TLOJ**

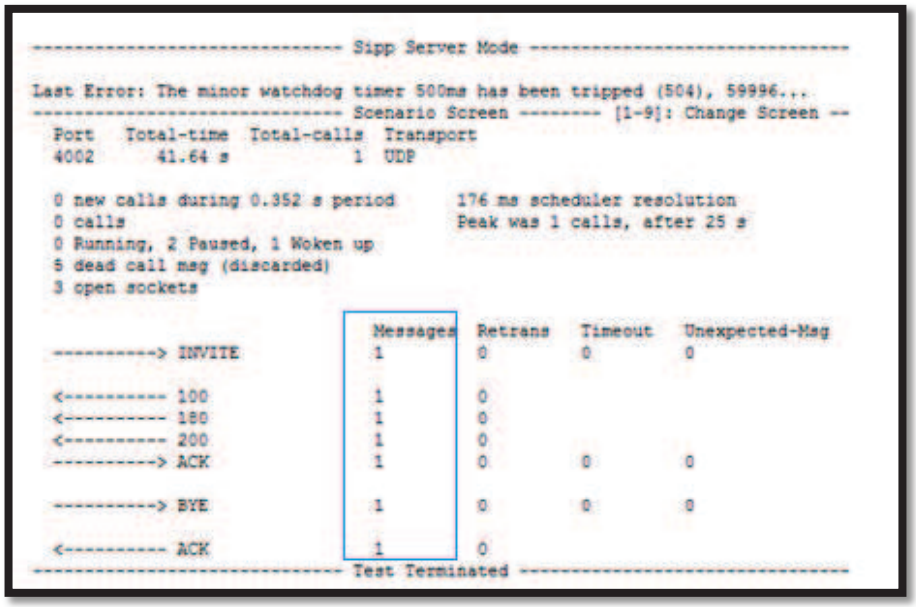


Figura 3.83 Proceso de llamada en TLOJ

- **Host cliente TESM**

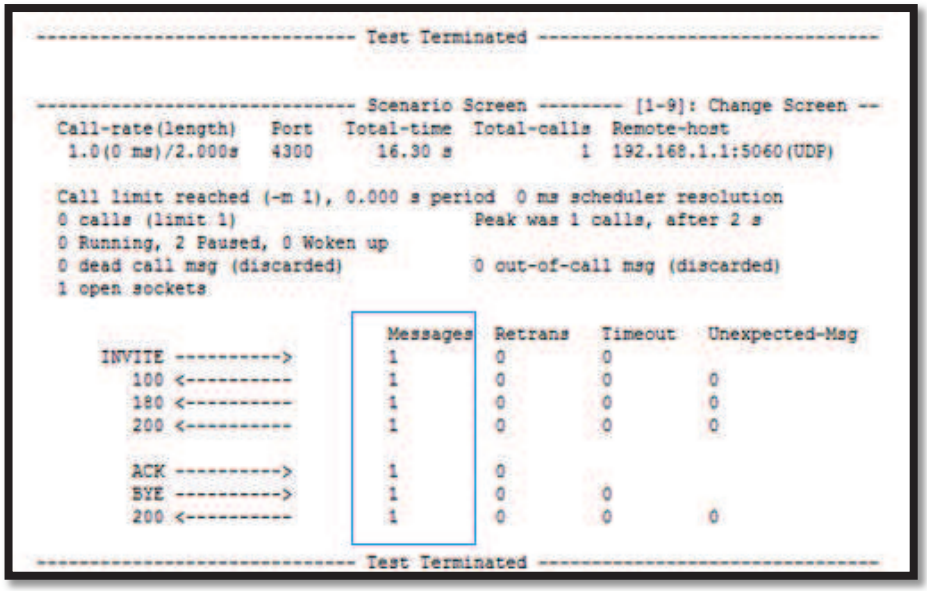


Figura 3.84 Proceso de llamada en TESM

Como se puede comprobar en las figuras 3.83 y 3.84 la llamada entre los usuarios TESM y TLOJ llegan a su fin con éxito como lo demuestran la columna de mensajes que toman el valor de 1.

Para una mejor comprensión de la prueba a continuación se detalla el diagrama de flujo del proceso de prueba expuesto y la respectiva demostración en el esquema de red.

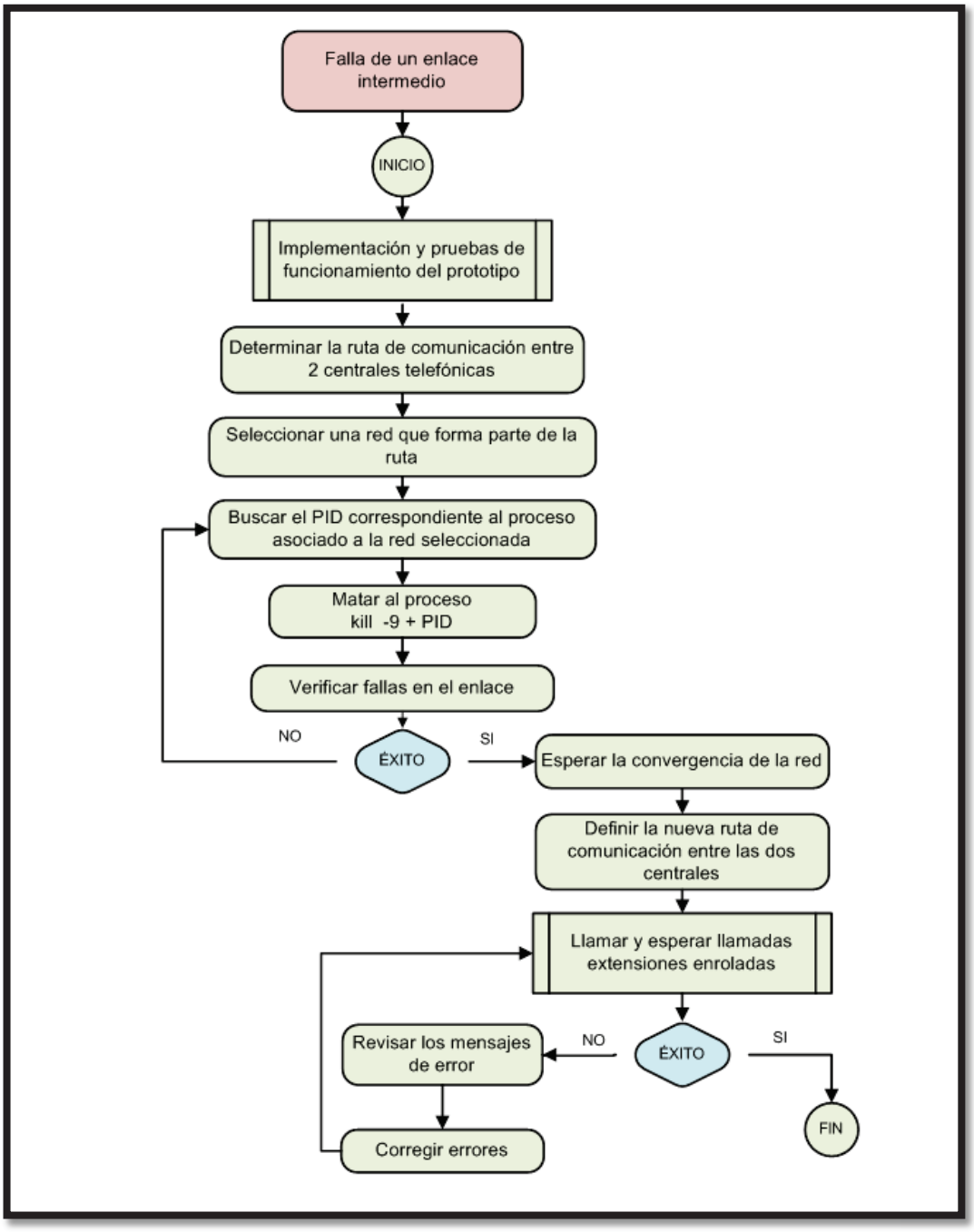
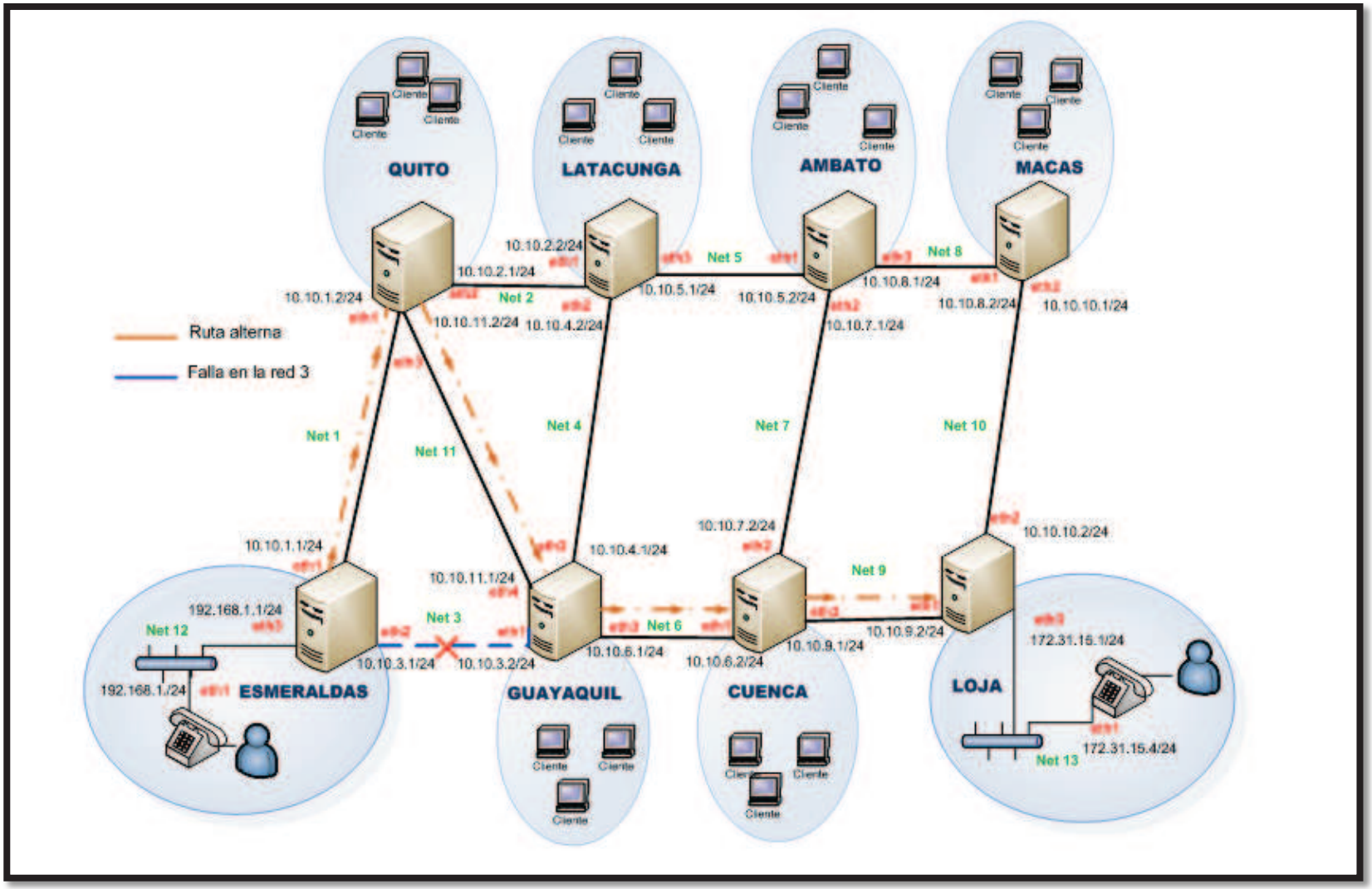


Figura 3.85 Diagrama de flujo de la prueba falla de una red en el sistema de telefonía IP

Figura 3.86 Falla en un enlace de la red (red 3)



3.5.2 FALLA DE DOS ENLACES INTERMEDIOS

Para comunicarse entre los clientes TESM y TLOJ una vez que se ha deshabilitado la red 3, se establece la siguiente ruta.

- **Central telefónica Esmeraldas**

```
esm:~# traceroute 10.10.9.2
```

```
esm:~# traceroute 10.10.9.2
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.1.2)  68.245 ms  68.096 ms  67.964 ms
 2   (10.10.11.1)  67.841 ms  67.722 ms  67.600 ms
 3   (10.10.6.2)  168.206 ms  168.092 ms  167.970 ms
 4   (10.10.9.2)  167.854 ms  167.732 ms  96.796 ms
```

Figura 3.87 Ruta desde ESM a LOJ sin la red 3

Como se puede observar esta ruta depende de la red 11 la cual interconecta las centrales de telefonía de Quito y Guayaquil. Al deshabilitar esta red la comunicación deberá ser redirigida por otra red de modo que la comunicación llegue a su fin.

- **Máquina física**

```
root@servidor-desktop:/home/servidor# ps -aux | grep Net
```

```
root     20782  0.0  0.0   1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net10.ctl
root     20819  0.0  0.0   1672   464 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net11.ctl
root     20857  0.0  0.0   1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net12.ctl
root     20893  0.0  0.0   1672   464 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net13.ctl
root     21051  6.0  1.7  67388 58900 ?        Ss   19:19   9:31 /usr/share/vnuml
```

Figura 3.88 Procesos del esquema de simulación red 11

Identificado el PID del proceso correspondiente a la red 11 se lo elimina.

```
root@servidor-desktop:/home/servidor# kill -9 20819
```

Se procede a la verificación de conectividad entre la central de Esmeraldas y Loja.

- **Central telefónica Quito**

  La central telefónica de Quito depende de la red 11 para su conectividad con la central de Guayaquil, al fallar el enlace no existe conectividad.

```
uio:~# ping 10.10.11.1
```

```
uio:~# ping 10.10.11.1
PING 10.10.11.1 (10.10.11.1) 56(84) bytes of data.
From 10.10.11.2 icmp_seq=1 Destination Host Unreachable
From 10.10.11.2 icmp_seq=4 Destination Host Unreachable
From 10.10.11.2 icmp_seq=5 Destination Host Unreachable
```

Figura 3.89 Conectividad entre UIO y GYE

- **Central telefónica Esmeraldas**

  A pesar de fallar dos enlaces, es decir la red 3 y 11 sigue existiendo conectividad entre la central de Esmeraldas y Loja, con lo cual se verifica la funcionalidad del sistema redundante.

```
esm:~# ping 10.10.9.2
```

```
esm:~# ping 10.10.9.2
PING 10.10.9.2 (10.10.9.2) 56(84) bytes of data.
64 bytes from 10.10.9.2: icmp_seq=1 ttl=60 time=161 ms
64 bytes from 10.10.9.2: icmp_seq=2 ttl=60 time=282 ms
64 bytes from 10.10.9.2: icmp_seq=3 ttl=60 time=503 ms
```

Figura 3.90 Conectividad entre ESM y LOJ

La ruta alterna es:

```
esm:~# traceroute 10.10.9.2
```

```
esm:~# traceroute 10.10.9.2
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.1.2)  62.421 ms  62.277 ms  62.154 ms
 2   (10.10.2.2)  139.052 ms  138.936 ms  138.824 ms
 3   (10.10.4.1)  162.852 ms  162.734 ms  162.611 ms
 4   (10.10.6.2)  295.875 ms  295.758 ms  194.776 ms
 5   (10.10.9.2)  315.995 ms  315.859 ms  315.743 ms
```

Figura 3.91 Ruta desde ESM hacia LOJ sin la red 3 y 11

Los pasos a ejecutarse para esta prueba serán:

- Registrar el teléfono TESM en ESM.

```
tesm:# sh registrar.sh 1050 4001 3002 192.168.1.1
```

- Registrar el teléfono TLOJ en LOJ.

```
tloj:# sh registrar.sh 9050 4002 3001 172.31.15.1
```

- Enrolar el número 1150 desde TESM en la central ESM.

```
tesm:# sh llamar.sh  4561150 4100 3100 192.168.1.1 1050
```

- Enrolar el número 9150 desde TLOJ en la central LOJ.

```
tloj:# sh llamar.sh 4569150 4200 3200 172.31.15.1 9050
```

- Verificación de resultados.

  - **Central telefónica Esmeraldas**

```
esm*CLI> sip show peers
Name/username        Host        Dyn Nat ACL Port   Status
1150/1150            10.10.1.1   D       5600   Unmonitored
1050/1050            192.168.1.4 D       4001   Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.92 Resultados de registro y enrolamiento en ESM

- **Central telefónica Loja**

```
loj*CLI> sip show peers
Name/username        Host        Dyn Nat ACL Port    Status
9150/9150            10.10.9.2   D       5600        Unmonitored
9050/9050            172.31.15.4 D       4002        Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.93 Resultados de registro y enrolamiento en LOJ

- Se verifica el funcionamiento de llamadas en el sistema. Se realiza la llamada desde TESM hacia TLOJ. En el cliente TLOJ, se ejecuta el script de esperar llamada, mientras que en el cliente TESM se ejecuta el script llamar con sus respectivos datos cada uno.

- **Cliente host TLOJ**

```
------------------------------ Scenario Screen -------- [1-9]: Change Screen --
  Port   Total-time  Total-calls  Transport
  4002     120.03 s            1  UDP

  0 new calls during 1.284 s period      183 ms scheduler resolution
  0 calls                                Peak was 1 calls, after 41 s
  0 Running, 0 Paused, 3 Woken up
  6 dead call msg (discarded)
  3 open sockets

                                 Messages  Retrans   Timeout   Unexpected-Msg
  ----------> INVITE            1         0         0         0

  <---------- 100               1         0
  <---------- 180               1         0
  <---------- 200               1         0
  ----------> ACK               1         0         0         0

  ----------> BYE               1         0         0         0

  <---------- ACK               1         0
------------------------------ Sipp Server Mode -------------------------------
```

Figura 3.94 Proceso de llamada en TLOJ sin la red 3 y 11

- **Cliente host TESM**

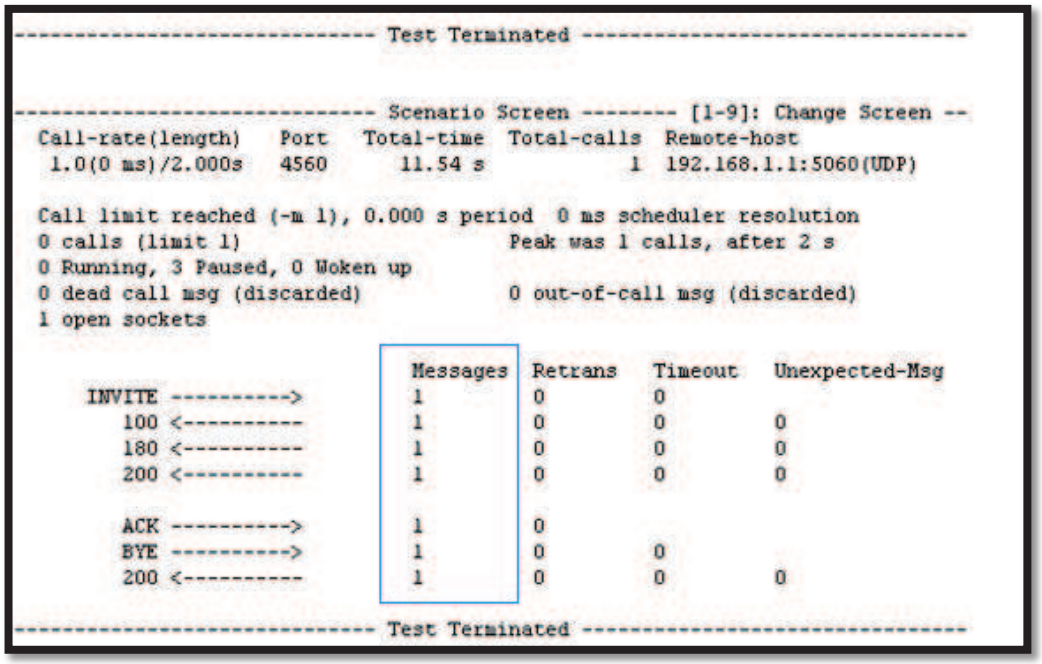


Figura 3.95 Proceso de llamada en TESM sin la red 3 y 11

Las figuras 3.94 y 3.95 permiten apreciar que el proceso de llamada entre dos usuarios del sistema llega a su fin con éxito incluso al fallar dos redes que forman parte de la ruta de comunicación, con lo cual se verifica la redundancia en el sistema de telefonía IP.

De igual forma de la prueba realizada en caso de falla de una red en el sistema de telefonía IP, se realiza un diagrama de flujo para explicar el proceso de la prueba en caso de falla de dos redes en el sistema, los cuales de detallan a continuación.

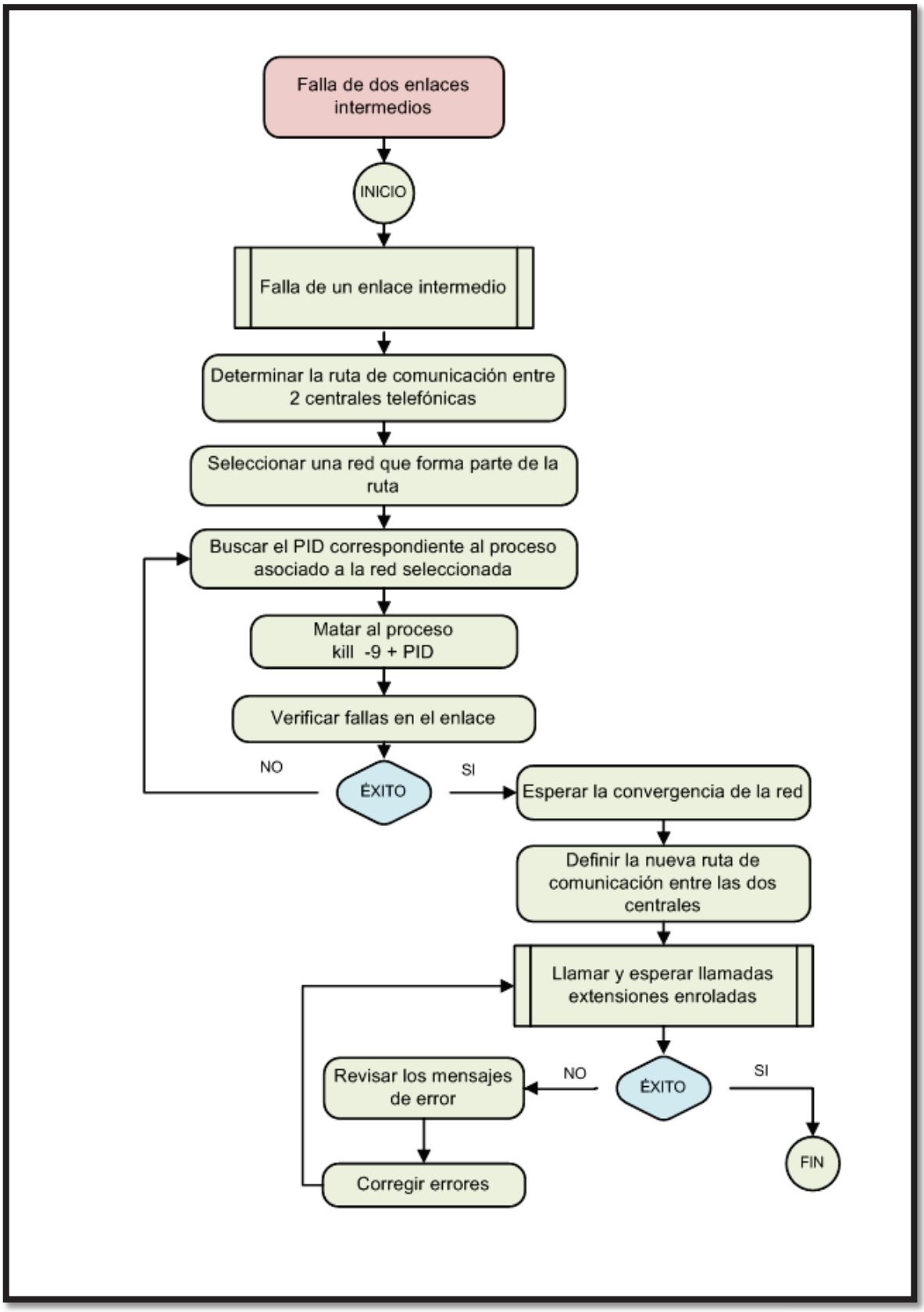
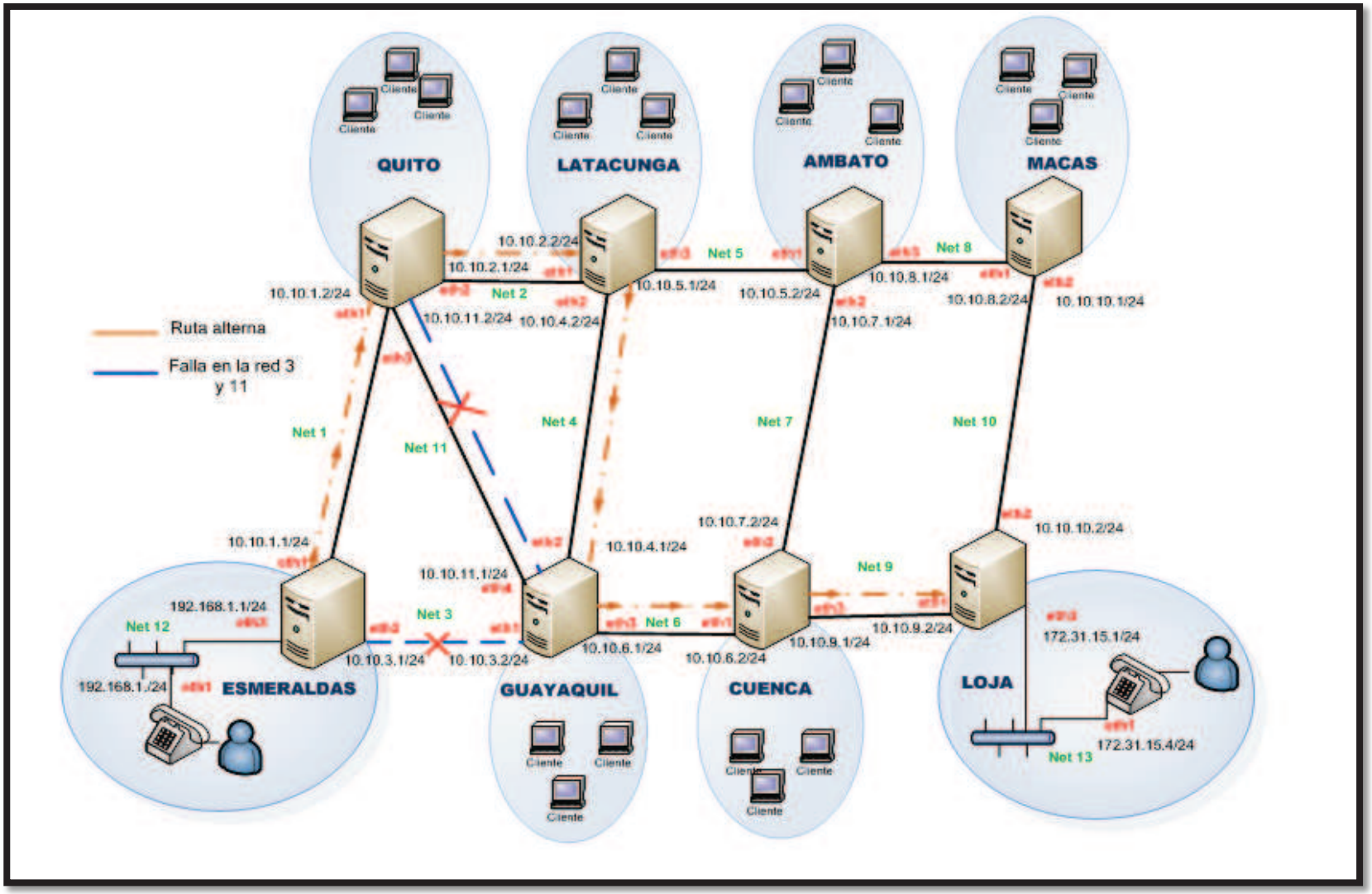


Figura 3.96 Diagrama de flujo de la prueba falla de dos enlaces intermedios en el sistema de telefonía IP

Figura 3.97 Falla de dos enlaces en la red (red 3 y red 11)



3.5.3 FALLA DE TRES ENLACES INTERMEDIOS

Para comunicarse entre los clientes TESM y TLOJ una vez que se ha deshabilitado la red 3y 11, se establece la siguiente ruta.

- **Central telefónica Esmeraldas**

```
esm:~# traceroute 10.10.9.2
```

```
esm:~# traceroute 10.10.9.2
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.1.2)  62.421 ms  62.277 ms  62.154 ms
 2   (10.10.2.2)  139.052 ms  138.936 ms  138.824 ms
 3   (10.10.4.1)  162.852 ms  162.734 ms  162.611 ms
 4   (10.10.6.2)  295.875 ms  295.758 ms  194.776 ms
 5   (10.10.9.2)  315.995 ms  315.859 ms  315.743 ms
```

Figura 3.98 Ruta desde ESM a LOJ sin la red 3

En este caso la ruta depende de la red 4 la cual interconecta las centrales telefónicas de Latacunga y Guayaquil. Al deshabilitar esta red la comunicación deberá ser redirigida por otra red para que la comunicación tenga éxito.

- **Máquina física**

```
root@servidor-desktop:/home/servidor# ps -aux | grep Net
```

```
root     20587  0.0  0.0  1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 3.ctl
root     20624  0.0  0.0  1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 4.ctl
root     20661  0.0  0.0  1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 5.ctl
root     20702  0.0  0.0  1672   464 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 6.ctl
root     20705  0.0  0.0  1672   468 ?        Ss   19:19   0:00 /usr/bin/uml_swi
ks/Net 7.ctl
```

Figura 3.99 Procesos del esquema de simulación red 4

Identificado el PID del proceso correspondiente a la red 11 se lo elimina.

```
root@servidor-desktop:/home/servidor# kill -9 20624
```

Se procede a la verificación de conectividad entre la central de Esmeraldas y Loja.

- **Central telefónica Latacunga**

  La central telefónica de Latacunga depende de la red 4 para su conectividad con la central de Guayaquil, al fallar el enlace no posee conectividad.

```
lat:~# ping 10.10.4.1
```

```
lat:~# ping 10.10.4.1
PING 10.10.4.1 (10.10.4.1) 56(84) bytes of data.
From 10.10.4.2 icmp_seq=3 Destination Host Unreachable
From 10.10.4.2 icmp_seq=4 Destination Host Unreachable
```

Figura 3.100 Conectividad entre LAT y GYE

- **Central telefónica Esmeraldas**

  Deshabilitadas las tres redes 3,11 y 4 en el esquema, sigue existiendo conectividad entre la central telefónica de Esmeraldas y Loja, conb lo cual queda comprobada la redundancia en el sistema.

```
esm:~# ping 10.10.9.2
```

```
esm:~# ping 10.10.9.2
PING 10.10.9.2 (10.10.9.2) 56(84) bytes of data.
64 bytes from 10.10.9.2: icmp_seq=1 ttl=60 time=161 ms
64 bytes from 10.10.9.2: icmp_seq=2 ttl=60 time=282 ms
64 bytes from 10.10.9.2: icmp_seq=3 ttl=60 time=503 ms
```

Figura 3.101Conectividad entre ESM y LOJ

La ruta alterna es:

```
esm:~# traceroute 10.10.9.2
```

```
esm:~# traceroute 10.10.9.2
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.1.2)  36.370 ms  36.220 ms  36.101 ms
 2   (10.10.2.2)  140.258 ms  140.140 ms  140.021 ms
 3   (10.10.5.2)  192.589 ms  192.469 ms  192.352 ms
 4   (10.10.7.2)  283.207 ms  283.089 ms *
 5   (10.10.9.2)  329.687 ms  329.558 ms  329.436 ms
```

Figura 3.102 Ruta desde ESM hacia LOJ sin la red 3,11 y 4

Los pasos a ejecutarse para esta prueba serán:

- Registrar el teléfono TESM en ESM.

```
tesm:# sh registrar.sh 1050 4001 3002 192.168.1.1
```

- Registrar el teléfono TLOJ en LOJ.

```
tloj:# sh registrar.sh 9050 4002 3001 172.31.15.1
```

- Enrolar el número 1150 desde TESM en la central ESM.

```
tesm:# sh llamar.sh  4561150 4100 3100 192.168.1.1 1050
```

- Enrolar el número 9150 desde TLOJ en la central LOJ.

```
tloj:# sh llamar.sh 4569150 4200 3200 172.31.15.1 9050
```

- Verificación de resultados.

  - **Central telefónica Esmeraldas**

```
esm*CLI> sip show peers
Name/username       Host        Dyn Nat ACL Port     Status
1150/1150           10.10.1.1   D       5600         Unmonitored
1050/1050           192.168.1.4 D       4001         Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.103 Resultados de registro y enrolamiento en ESM

- **Central telefónica Loja**

```
loj*CLI> sip show peers

Name/username        Host        Dyn Nat ACL Port   Status

9150/9150            10.10.9.2   D       5600   Unmonitored

9050/9050            172.31.15.4 D       4002   Unmonitored

2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.104 Resultados de registro y enrolamiento en LOJ

- Se verifica el funcionamiento de llamadas en el sistema. Se realiza la llamada desde TESM hacia TLOJ, con sus respectivos datos cada uno.

- **Cliente host TLOJ**

```
------------------------------ Scenario Screen -------- [1-9]: Change Screen --
  Port   Total-time  Total-calls  Transport
  4002     120.03 s            1  UDP

  0 new calls during 1.284 s period      183 ms scheduler resolution
  0 calls                                Peak was 1 calls, after 41 s
  0 Running, 0 Paused, 3 Woken up
  6 dead call msg (discarded)
  3 open sockets

                                 Messages  Retrans   Timeout   Unexpected-Msg
  ----------> INVITE            1         0         0         0

  <---------- 100               1         0
  <---------- 180               1         0
  <---------- 200               1         0
  ----------> ACK               1         0         0         0

  ----------> BYE               1         0         0         0

  <---------- ACK               1         0
------------------------------ Sipp Server Mode -------------------------------
```

Figura 3.105 Proceso de llamada en TLOJ sin la red 3, 11 y 4

- **Cliente host TESM**

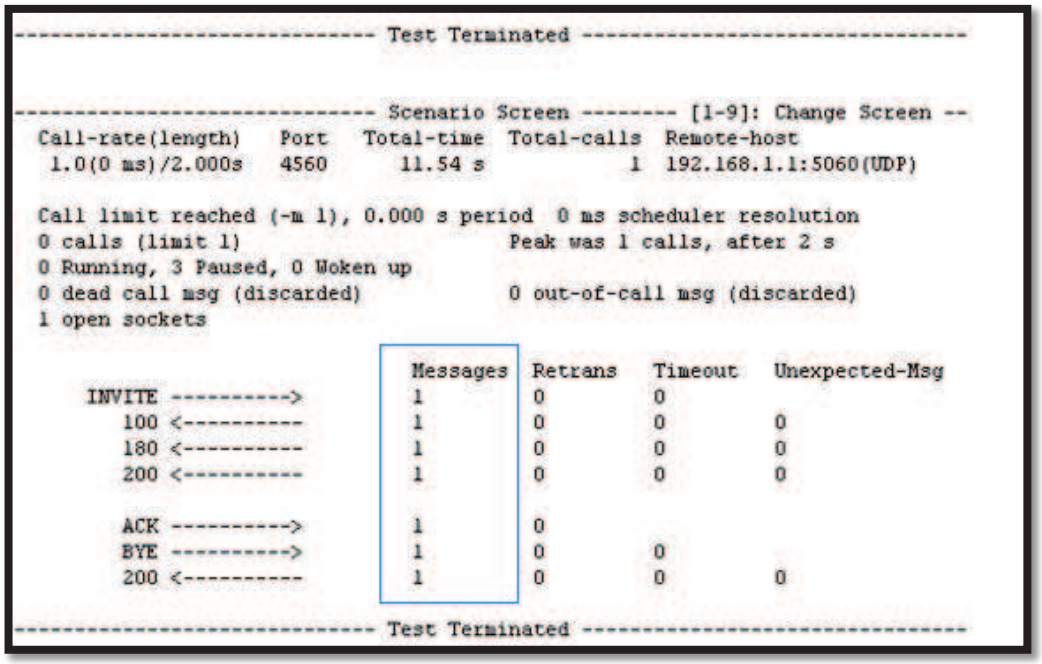


Figura 3.106 Proceso de llamada en TESM sin la red 3,11 y 4

Dentro del esquema de red en malla, como se puede verificar en esta prueba, se garantiza el éxito de la comunicación y por ende del establecimiento de llamada entre dos usuarios ubicados a los dos extremos de la red (TESM y TLOJ), a pesar de fallar simultáneamente un total de tres redes en la comunicación. Este es el caso extremo para un correcto funcionamiento de la llamada, debido que al fallar tres redes en la comunicación únicamente se dispone de una sola ruta alterna para interconectar a dos usuarios en el sistema.

Para apreciar el funcionamiento de la prueba a continuación se realiza el diagrama de flujo relacionado a este proceso y su respectiva gráfica en el esquema de red.

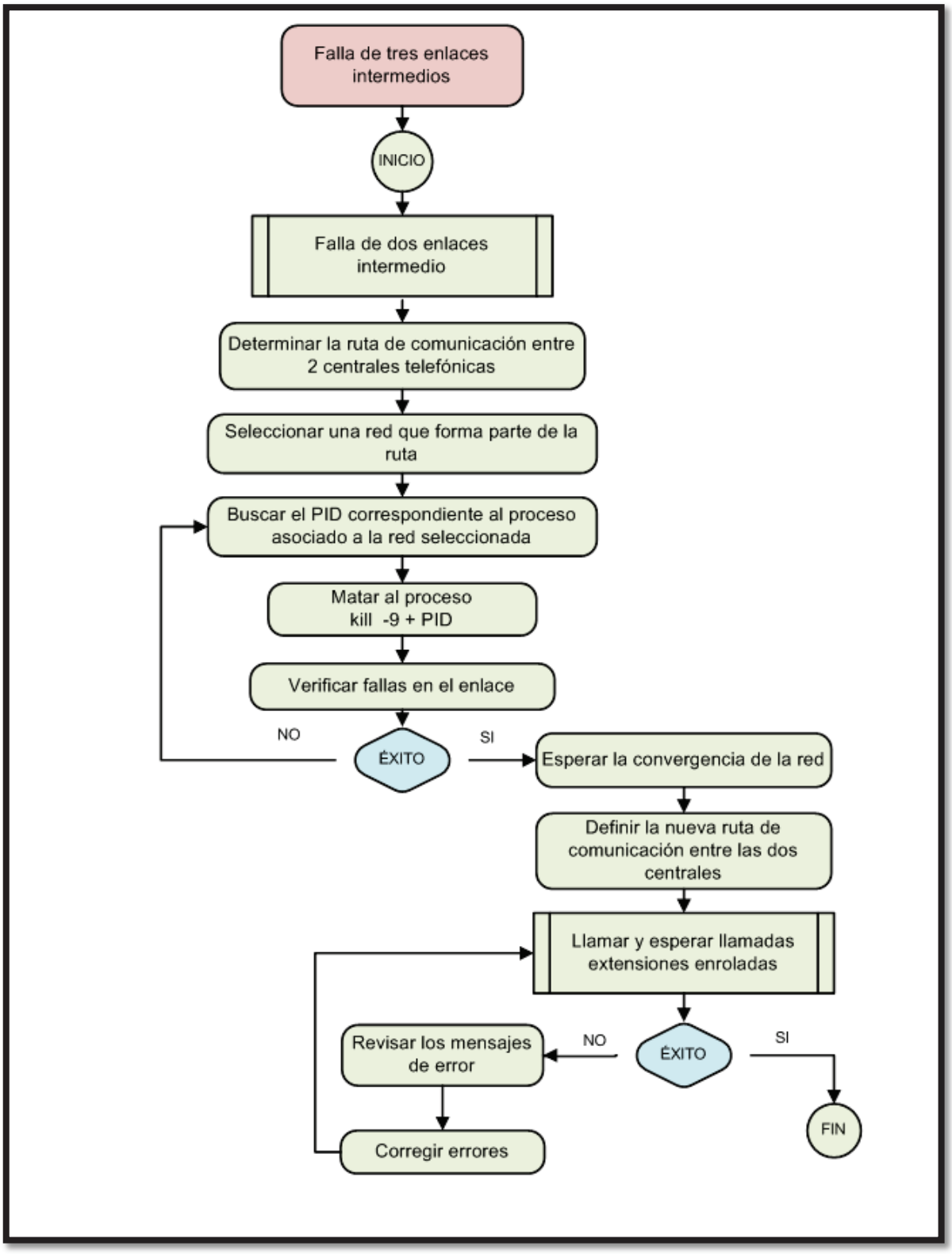
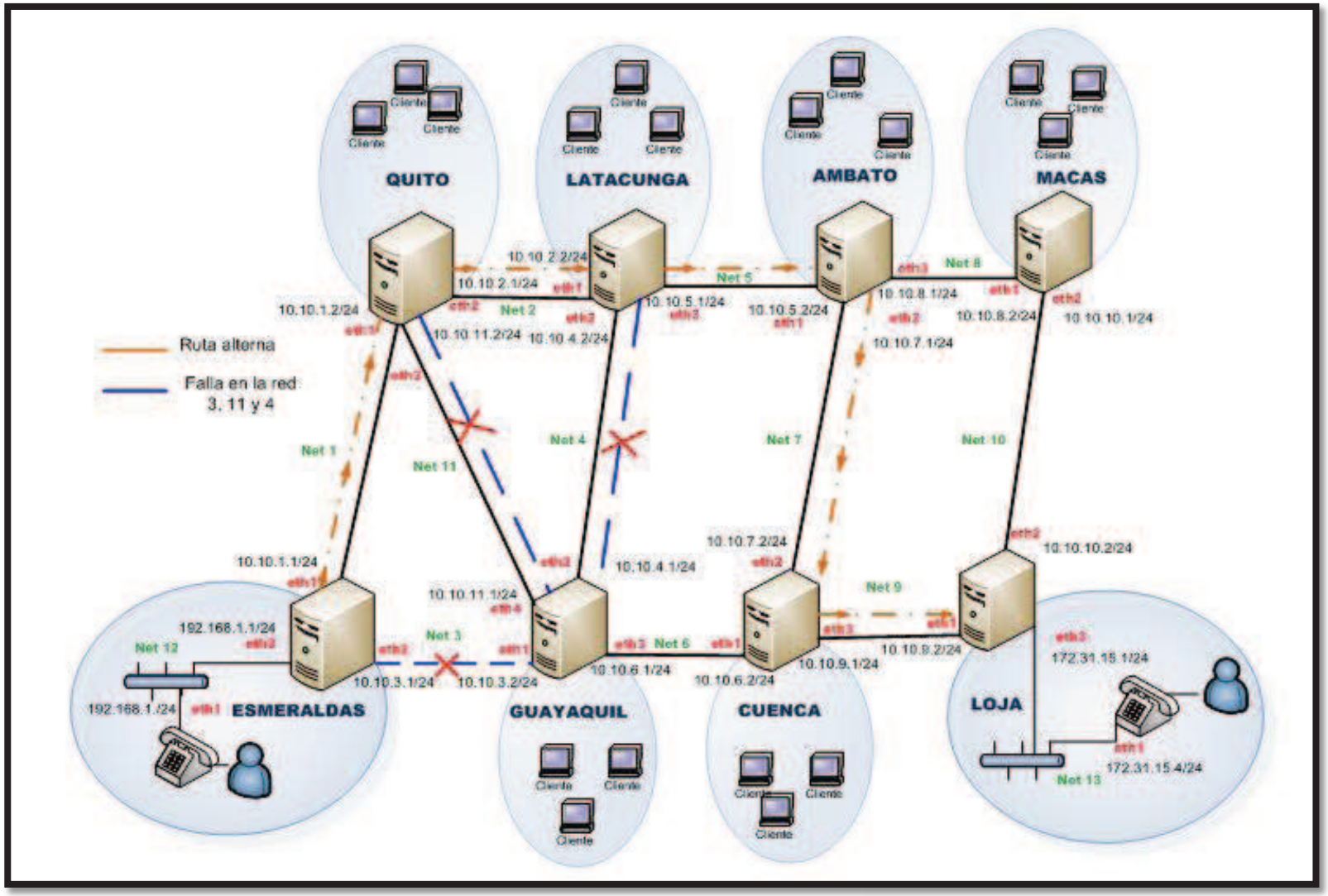


Figura 3.107 Diagrama de flujo de la prueba falla de tres enlaces intermedios en el sistema de telefonía IP

Figura 3.108 Falla de tres enlaces en la red 3,11 y 4



3.5.4 FALLA DE UN NODO INTERMEDIO

El otro escenario es la falla total del nodo GYE el cual forma parte de la ruta como lo indica el comando traceroute:

```
esm:~# traceroute 10.10.9.2
```

```
esm:~# traceroute 10.10.9.2
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.3.2)  67.284 ms  67.136 ms  67.017 ms
 2   (10.10.6.2)  66.900 ms  66.781 ms  66.661 ms
 3   (10.10.9.2)  129.033 ms  128.914 ms  128.794 ms
```

Figura 3.109 Ruta desde ESM a LOJ

Para verificar que hay redundancia en el sistema de telefonía IP, se provocará la falla del nodo GYE:

- Terminar los procesos correspondientes a la simulación del nodo GYE.

  - **Máquina física**

```
root@servidor-desktop:~# ps -aux |grep gye

root@servidor-desktop:~#kill -9 2358

root@servidor-desktop:~#kill -9 2402

root@servidor-desktop:~#kill -9 2403

root@servidor-desktop:~#kill -9 2404
```

- Se comprueba la des-habilitación del nodo GYE.

  - **Máquina física**

```
root@servidor-desktop:~# ssh gye
```

```
root@servidor-desktop:~# ssh gye
ssh: connect to host gye port 22: No route to host
```

Figura 3.110Fallo en la Central telefónica Guayaquil

- Esperar un tiempo para que vuelva a convergir la red, tanto a nivel de enrutamiento IP como de enrutamiento de llamada. Verificar la nueva ruta desde Esmeraldas a Loja.

  - **Central telefónica Esmeraldas**

```
esm:~# traceroute 10.10.9.2
```

```
esm:~# traceroute 10.10.9.2
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.1.2)  0.202 ms  0.117 ms  5.709 ms
 2   (10.10.2.2)  5.869 ms  5.755 ms  17.304 ms
 3   (10.10.5.2)  17.192 ms  17.077 ms  16.963 ms
 4   (10.10.7.2)  16.849 ms  16.734 ms  16.617 ms
 5   (10.10.9.2)  18.533 ms  13.841 ms  17.526 ms
```

Figura 3.111 Ruta alterna desde ESM a LOJ sin el nodo GYE

- Verificar las nuevas tablas de enrutamiento en el esquema de red. Se comprueban las tablas de rutas de los nodos adyacentes a Guayaquil (GYE), los cuales alteran su funcionamiento en caso de fallar el nodo GYE. Se observará como cambia el Gateway para algunas rutas.

  - **Central telefónica Quito**

```
uio:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.104   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       10.10.2.2       255.255.255.0   UG    4      0        0 eth2
10.10.7.0       10.10.2.2       255.255.255.0   UG    3      0        0 eth2
10.10.4.0       10.10.2.2       255.255.255.0   UG    2      0        0 eth2
10.10.5.0       10.10.2.2       255.255.255.0   UG    2      0        0 eth2
10.10.2.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.3.0       10.10.1.1       255.255.255.0   UG    2      0        0 eth1
10.10.1.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.10.0      10.10.2.2       255.255.255.0   UG    4      0        0 eth2
10.10.11.0      0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.8.0       10.10.2.2       255.255.255.0   UG    3      0        0 eth2
10.10.9.0       10.10.2.2       255.255.255.0   UG    4      0        0 eth2
```

Figura 3.112 Tabla de rutas de UIO sin el nodo GYE

- **Central telefónica Latacunga**

```
lat:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.108   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       10.10.5.2       255.255.255.0   UG    3      0        0 eth3
10.10.7.0       10.10.5.2       255.255.255.0   UG    2      0        0 eth3
10.10.4.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.5.0       0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.2.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.3.0       10.10.2.1       255.255.255.0   UG    3      0        0 eth1
10.10.1.0       10.10.2.1       255.255.255.0   UG    2      0        0 eth1
10.10.10.0      10.10.5.2       255.255.255.0   UG    3      0        0 eth3
10.10.11.0      10.10.2.1       255.255.255.0   UG    2      0        0 eth1
10.10.8.0       10.10.5.2       255.255.255.0   UG    2      0        0 eth3
10.10.9.0       10.10.5.2       255.255.255.0   UG    3      0        0 eth3
```

Figura 3.113Tabla de rutas de LAT sin el nodo GYE

- **Central telefónica Cuenca**

```
cue:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.116   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.7.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.4.0       10.10.7.1       255.255.255.0   UG    3      0        0 eth2
10.10.5.0       10.10.7.1       255.255.255.0   UG    2      0        0 eth2
10.10.2.0       10.10.7.1       255.255.255.0   UG    3      0        0 eth2
10.10.3.0       10.10.7.1       255.255.255.0   UG    5      0        0 eth2
10.10.1.0       10.10.7.1       255.255.255.0   UG    4      0        0 eth2
10.10.10.0      10.10.9.2       255.255.255.0   UG    2      0        0 eth3
10.10.11.0      10.10.7.1       255.255.255.0   UG    4      0        0 eth2
10.10.8.0       10.10.7.1       255.255.255.0   UG    2      0        0 eth2
10.10.9.0       0.0.0.0         255.255.255.0   U     0      0        0 eth3
```

Figura 3.114 Tabla de rutas de CUE sin el nodo GYE

- **Central telefónica Esmeraldas**

```
esm:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.100   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.6.0       10.10.1.2       255.255.255.0   UG    5      0        0 eth1
10.10.7.0       10.10.1.2       255.255.255.0   UG    4      0        0 eth1
10.10.4.0       10.10.1.2       255.255.255.0   UG    3      0        0 eth1
10.10.5.0       10.10.1.2       255.255.255.0   UG    3      0        0 eth1
10.10.2.0       10.10.1.2       255.255.255.0   UG    2      0        0 eth1
10.10.3.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.1.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.10.0      10.10.1.2       255.255.255.0   UG    5      0        0 eth1
10.10.11.0      10.10.1.2       255.255.255.0   UG    2      0        0 eth1
10.10.8.0       10.10.1.2       255.255.255.0   UG    4      0        0 eth1
10.10.9.0       10.10.1.2       255.255.255.0   UG    5      0        0 eth1
```

Figura 3.115 Tabla de rutas de ESM sin el nodo GYE

- Verificar la conectividad entre los nodos extremos.

  - **De Esmeraldas a Loja**

```
esm:~# ping 10.10.9.2
```

```
esm:~# ping 10.10.9.2
PING 10.10.9.2 (10.10.9.2) 56(84) bytes of data.
64 bytes from 10.10.9.2: icmp_seq=1 ttl=60 time=76.2 ms
64 bytes from 10.10.9.2: icmp_seq=2 ttl=60 time=10.9 ms
64 bytes from 10.10.9.2: icmp_seq=3 ttl=60 time=0.699 ms
```

Figura 3.116 Conectividad de ESM a LOJ sin el nodo GYE

  - **De Loja a Esmeraldas**

```
loj:~# ping 10.10.1.1
```

```
loj:~# ping 10.10.1.1
PING 10.10.1.1 (10.10.1.1) 56(84) bytes of data.
64 bytes from 10.10.1.1: icmp_seq=1 ttl=60 time=93.7 ms
64 bytes from 10.10.1.1: icmp_seq=2 ttl=60 time=0.608 ms
64 bytes from 10.10.1.1: icmp seq=3 ttl=60 time=0.617 ms
```

Figura 3.117 Conectividad de LOJ a ESM sin el nodo GYE

- Verificar las adyacencias *DUNDi* entre los nodos activos y la falta de adyacencia con el nodo desactivado (GYE).

  - **Central telefónica Quito**

```
uio*CLI> dundi show peers
```

```
uio*CLI> dundi show peers
EID                 Host                 Model      AvgTime  Status
fe:fd:00:00:04:00   10.10.11.1      (S) Symmetric  Unavail  UNREACHABLE
fe:fd:00:00:03:00   10.10.2.2       (S) Symmetric  Unavail  OK (1084 ms)
fe:fd:00:00:01:00   10.10.1.1       (S) Symmetric  Unavail  OK (3 ms)
3 dundi peers [2 online, 1 offline, 0 unmonitored]
```

Figura 3.118 Nodos DUNDi alcanzados por UIO

- **Central telefónica Latacunga**

```
lat*CLI> dundi show peers
```

```
lat*CLI> dundi show peers
EID                 Host                 Model      AvgTime  Status
fe:fd:00:00:02:00   10.10.2.1       (S)  Symmetric  Unavail  OK (69 ms)
fe:fd:00:00:04:00   10.10.4.1       (S)  Symmetric  Unavail  UNREACHABLE
fe:fd:00:00:06:00   10.10.5.2       (S)  Symmetric  Unavail  OK (27 ms)
3 dundi peers [2 online, 1 offline, 0 unmonitored]
```

Figura 3.119 Nodos DUNDi alcanzados por LAT

- **Central telefónica Cuenca**

```
cue*CLI> dundi show peers
```

```
cue*CLI> dundi show peers
EID                 Host                 Model      AvgTime  Status
fe:fd:00:00:08:00   10.10.9.2       (S)  Symmetric  Unavail  OK (68 ms)
fe:fd:00:00:04:00   10.10.6.1       (S)  Symmetric  Unavail  UNREACHABLE
fe:fd:00:00:06:00   10.10.7.1       (S)  Symmetric  Unavail  OK (68 ms)
3 dundi peers [2 online, 1 offline, 0 unmonitored]
```

Figura 3.120Nodos DUNDi alcanzados por CUE

- **Central telefónica Esmeraldas**

```
esm*CLI> dundi show peers
```

```
esm*CLI> dundi show peers
EID                 Host                 Model      AvgTime  Status
fe:fd:00:00:04:00   10.10.3.2       (S)  Symmetric  Unavail  UNREACHABLE
fe:fd:00:00:02:00   10.10.1.2       (S)  Symmetric  Unavail  OK (6 ms)
2 dundi peers [1 online, 1 offline, 0 unmonitored]
```

Figura 3.121 Nodos DUNDi alcanzados por ESM

- Verificar que la llamada entre TLOJ y TESM pueda completarse, aún a través de la malla modificada, de modo que se compruebe el funcionamiento del servidor de redirección que depende del protocolo *DUNDi* y de la herramienta *SIPP*.

- **Central telefónica Esmeraldas**

  Se realiza el registro de la extensión estática 1050 y el enrolamiento de la extensión personal 1150.

```
esm*CLI>sip show peers
```

```
esm*CLI> sip show peers
Name/username              Host            Dyn Nat ACL Port     Status
1150/1150                  10.10.1.1        D          5600     Unmonitored
1050/1050                  192.168.1.4      D          4001     Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.122 Extensiones registradas en ESM

- **Central telefónica Loja**

  Se realiza el registro de la extensión estática 9050 y el enrolamiento de la extensión personal 9150.

```
loj*CLI>sip show peers
```

```
loj*CLI> sip show peers
Name/username              Host            Dyn Nat ACL Port     Status
9150/9150                  10.10.9.2        D          5601     Unmonitored
9050/9050                  172.31.15.4      D          4002     Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.123 Extensiones registradas en LOJ

- Registradas las extensiones se procede a realizar la llamada de prueba desde el cliente TLOJ hacia el cliente TESM. Este proceso implica iniciar el script esperar llamada en TESM y ejecutar el script de llamada en TLOJ. Se toman para este ejemplo los datos utilizados en las anteriores pruebas llevadas a cabo en el sistema.

- **Host cliente TESM**

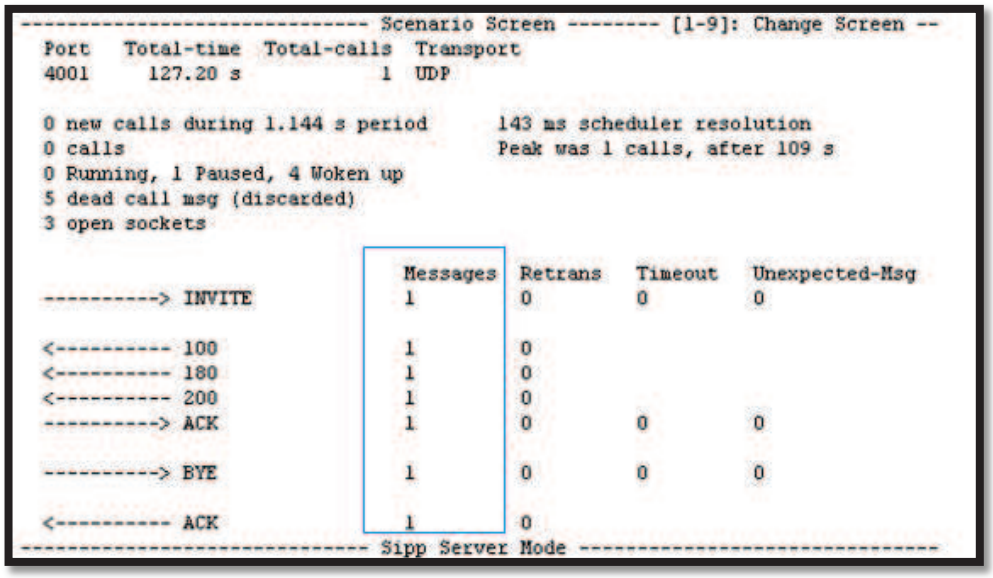


Figura 3.124 Proceso de llamada en TESM sin el nodo GYE

- **Host cliente TLOJ**

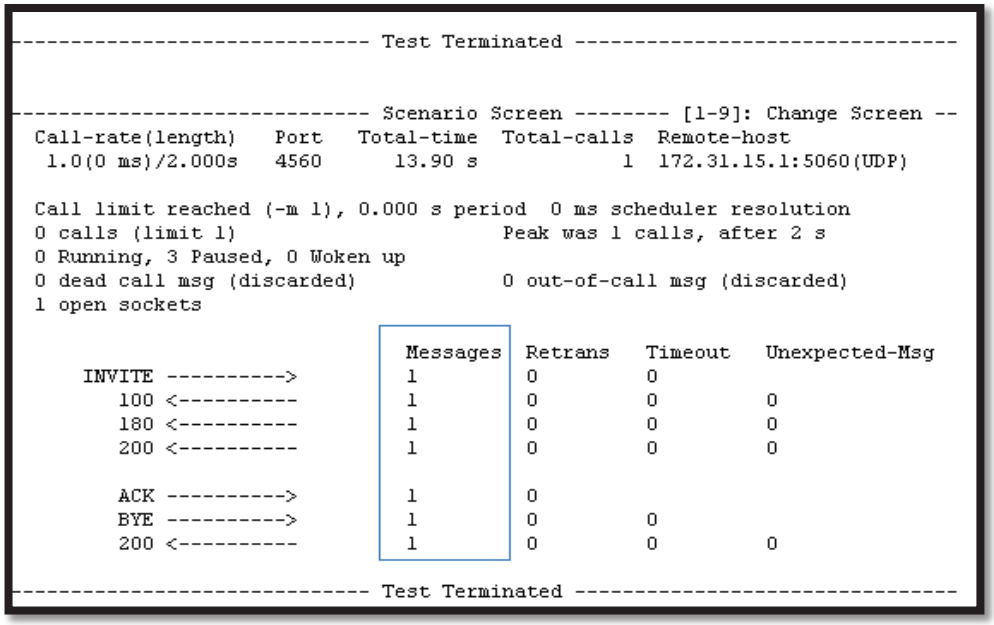


Figura 3.125 Proceso de llamada en TLOJ sin el nodo GYE

De este modo se ha llegado al fin de la prueba comprobando el correcto establecimiento de la llamada entre los usuarios de TESM y TLOJ, tomando en cuenta la falla de un nodo en la red es decir la central de telefonía de IP en

Guayaquil (GYE) ha quedado fuera de uso, sin embargo este suceso no alteró el enrutamiento de la llamada en el sistema de telefonía IP.Seguido de detalla el diagrama de flujo del funcionamiento de esta prueba y su respectivo esquema.

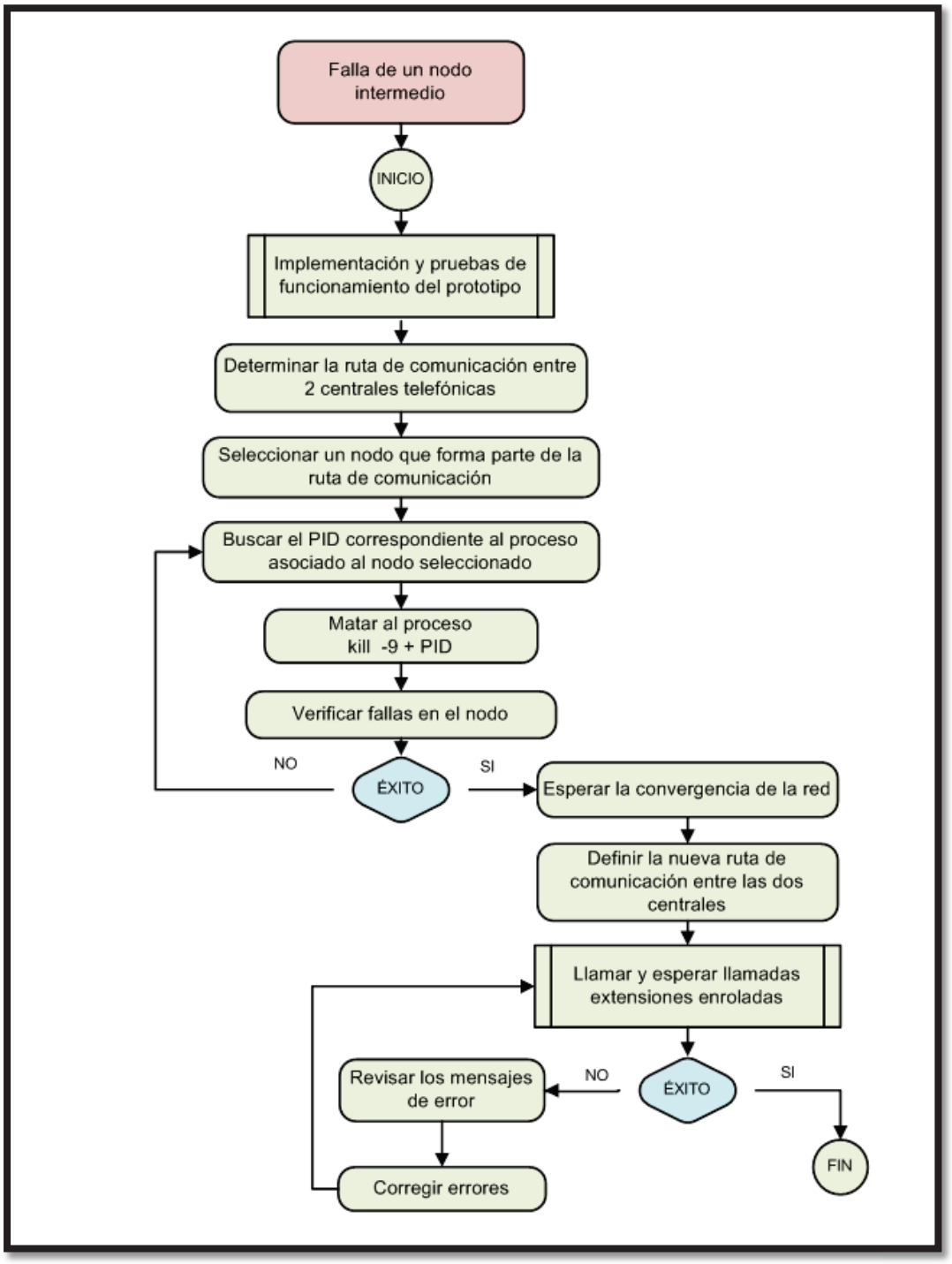
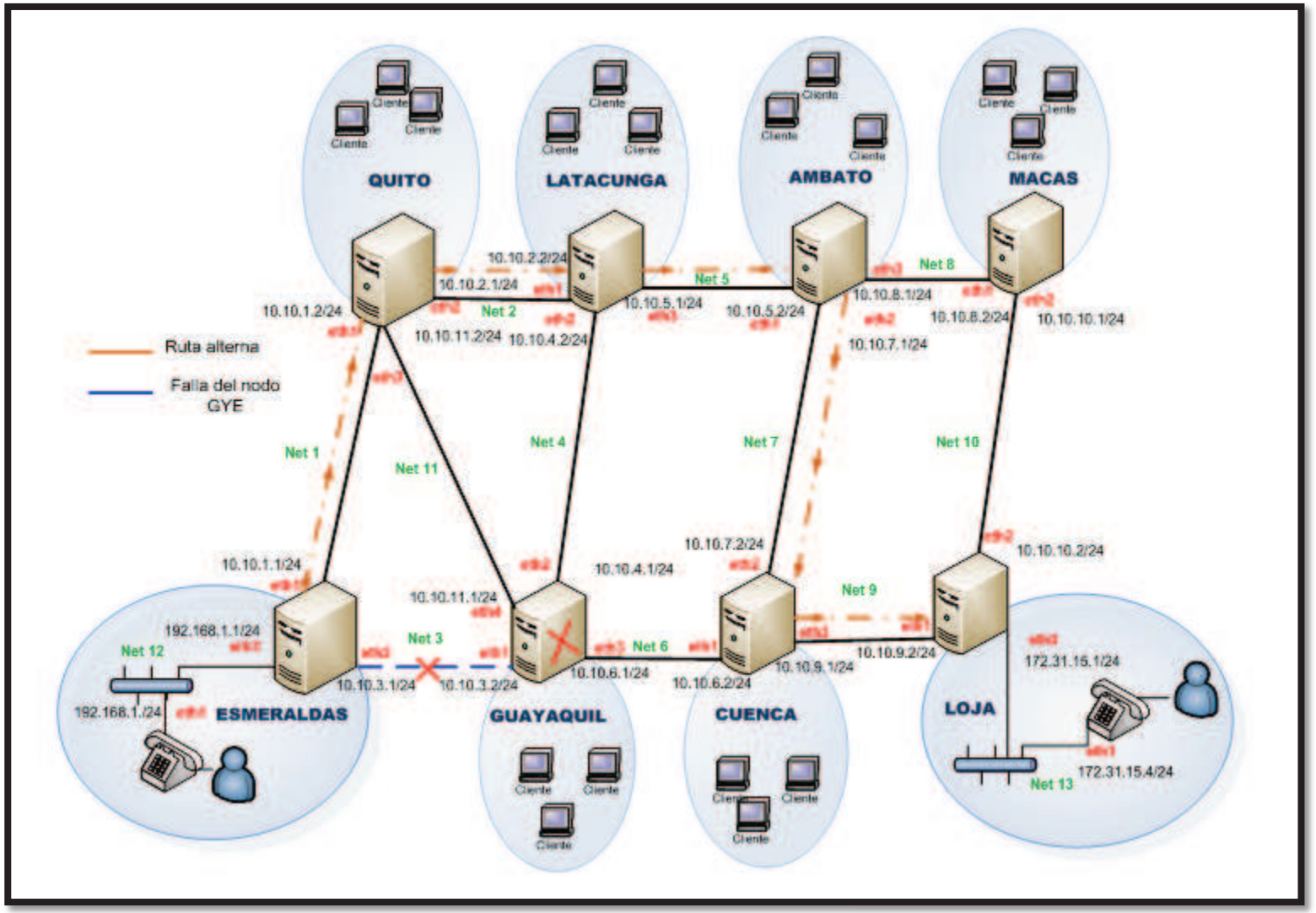


Figura 3.126 Diagrama de flujo de la prueba falla de un nodo intermedio en el sistema de telefonía IP

Figura 3.127 Falla de un nodo intermedio en la red (nodo GYE)



3.5.5 FALLA DE DOS NODOS INTERMEDIOS

Al fallar GYE se tomó otra ruta alterna como lo muestra traceroute, la cual va por CUE

- **Host cliente TESM**

```
esm:~# traceroute
```

```
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.1.2)  11.697 ms  11.553 ms  11.430 ms
 2   (10.10.2.2)  32.256 ms  32.138 ms  32.014 ms
 3   (10.10.5.2)  44.942 ms  44.826 ms  66.328 ms
 4   (10.10.7.2)  124.892 ms  124.776 ms  124.661 ms
 5   (10.10.9.2)  204.514 ms  204.400 ms  204.281 ms
```

Figura 3.128 Ruta desde ESM a LOJ sin nodo GYE

Para verificar que hay redundancia en el sistema de telefonía IP, se provocará la falla del nodo CUE, por lo tanto la nueva ruta debe ser por Macas:

- Terminar los procesos correspondientes a la simulación del nodo CUE.

  - **Máquina física**

```
root@servidor-desktop:~# ps -aux |grep cue

root@servidor-desktop:~#kill -9 5407

root@servidor-desktop:~#kill -9 5457

root@servidor-desktop:~#kill -9 5458

root@servidor-desktop:~#kill -9 5459
```

Se comprueba la falla del nodo MAC al ser imposible el acceso.

```
root@servidor-desktop:~# sshcue
```

```
root@servidor-desktop:/home/servidor# ssh cue
ssh: connect to host cue port 22: No route to host
```

Figura 3.129 Falla del nodo CUE

- Esperar un tiempo para que vuelva a convergir la red, tanto a nivel de enrutamiento IP como de enrutamiento de llamada. Verificar la nueva ruta desde Esmeraldas a Loja.

  - **Central telefónica Esmeraldas**

```
esm:~# traceroute 10.10.9.2
```

```
traceroute to 10.10.9.2 (10.10.9.2), 30 hops max, 60 byte packets
 1   (10.10.1.2)  22.436 ms  22.292 ms  22.163 ms
 2   (10.10.2.2)  71.414 ms  71.293 ms  71.174 ms
 3   (10.10.5.2)  71.043 ms  70.947 ms  70.827 ms
 4   (10.10.8.2)  155.260 ms  155.140 ms  155.016 ms
 5   (10.10.9.2)  253.293 ms  253.157 ms  253.034 ms
```

Figura 3.130 Ruta desde ESM a LOJ sin nodo GYE y CUE

- Verificar las nuevas tablas de enrutamiento en el esquema de red, de los nodos directamente conectados con el nodo CUE.

  - **Central telefónica Ambato**

```
amb:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.120   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.7.0       0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.4.0       10.10.5.1       255.255.255.0   UG    2      0        0 eth1
10.10.5.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.10.2.0       10.10.5.1       255.255.255.0   UG    2      0        0 eth1
10.10.3.0       10.10.5.1       255.255.255.0   UG    4      0        0 eth1
10.10.1.0       10.10.5.1       255.255.255.0   UG    3      0        0 eth1
10.10.10.0      10.10.8.2       255.255.255.0   UG    2      0        0 eth3
10.10.11.0      10.10.5.1       255.255.255.0   UG    3      0        0 eth1
10.10.8.0       0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.9.0       10.10.8.2       255.255.255.0   UG    3      0        0 eth3
```

Figura 3.131Tabla de rutas en AMB sin nodo GYE y CUE

- **Central telefónica LOJA**

```
loj:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.128   0.0.0.0         255.255.255.252 U     0      0        0 eth0
10.10.7.0       10.10.10.1      255.255.255.0   UG    3      0        0 eth2
10.10.4.0       10.10.10.1      255.255.255.0   UG    4      0        0 eth2
172.31.15.0     0.0.0.0         255.255.255.0   U     0      0        0 eth3
10.10.5.0       10.10.10.1      255.255.255.0   UG    3      0        0 eth2
10.10.2.0       10.10.10.1      255.255.255.0   UG    4      0        0 eth2
10.10.3.0       10.10.10.1      255.255.255.0   UG    6      0        0 eth2
10.10.1.0       10.10.10.1      255.255.255.0   UG    5      0        0 eth2
10.10.10.0      0.0.0.0         255.255.255.0   U     0      0        0 eth2
10.10.11.0      10.10.10.1      255.255.255.0   UG    5      0        0 eth2
10.10.8.0       10.10.10.1      255.255.255.0   UG    2      0        0 eth2
10.10.9.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
```

Figura 3.132 Tabla de rutas en LOJ sin nodo GYE y CUE

- Verificar la conectividad entre los nodos extremos.

  - **De Esmeraldas a Loja**

```
esm:~# ping 10.10.9.2
```

```
esm:~# ping 10.10.9.2
PING 10.10.9.2 (10.10.9.2) 56(84) bytes of data.
64 bytes from 10.10.9.2: icmp_seq=1 ttl=60 time=271 ms
64 bytes from 10.10.9.2: icmp_seq=2 ttl=60 time=308 ms
```

Figura 3.133 Conectividad de ESM a LOJ sin nodo GYE y CUE

  - **De Loja a Esmeraldas**

```
loj:~# ping 10.10.1.1
```

```
loj:~# ping 10.10.1.1
PING 10.10.1.1 (10.10.1.1) 56(84) bytes of data.
64 bytes from 10.10.1.1: icmp_seq=1 ttl=60 time=176 ms
64 bytes from 10.10.1.1: icmp_seq=2 ttl=60 time=215 ms
```

Figura 3.134 Conectividad de LOJ a ESM sin nodo GYE y CUE

- Verificar las adyacencias DUNDI entre los nodos activos y la falta de adyacencia con el nodo desactivado CUE.

- **Central telefónica Loja**

```
loj*CLI> dundi show peers
```

```
loj*CLI> dundi show peers
EID                 Host                 Model      AvgTime  Status
fe:fd:00:00:05:00   10.10.9.1      (S)   Symmetric  2090 ms  UNREACHABLE
fe:fd:00:00:07:00   10.10.10.1     (S)   Symmetric  1875 ms  OK (116 ms)
2 dundi peers [1 online, 1 offline, 0 unmonitored]
```

Figura 3.135 Nodos DUNDi alcanzados por LOJ

- **Central telefónica amb**

```
amb*CLI> dundi show peers
```

```
amb*CLI> dundi show peers
EID                 Host                 Model      AvgTime  Status
fe:fd:00:00:07:00   10.10.8.2      (S)   Symmetric  1391 ms  OK (35 ms)
fe:fd:00:00:05:00   10.10.7.2      (S)   Symmetric  1491 ms  UNREACHABLE
fe:fd:00:00:03:00   10.10.5.1      (S)   Symmetric  1776 ms  OK (43 ms)
3 dundi peers [2 online, 1 offline, 0 unmonitored]
```

Figura 3.136 Nodos DUNDi alcanzados por AMB

- Verificar que la llamada entre TLOJ y TESM pueda completarse, aún a través de la malla modificada.

  - **Central telefónica Esmeraldas**

    Se procede a registrar la extensión estática 1050 y a enrolar la extensión personal 1150.

```
esm*CLI> dundi show peers
```

```
esm*CLI> sip show peers
Name/username              Host            Dyn Nat ACL Port     Status
1150/1150                  10.10.1.1        D          5601     Unmonitored
1050/1050                  192.168.1.4      D          4001     Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.137 Extensiones registradas en ESM

- **Central telefónica LOJA**

  Se procede a registrar la extensión estática 9050 y a enrolar la extensión personal 9150.

```
loj*CLI> dundi show peers
```

```
loj*CLI> sip show peers
Name/username              Host            Dyn Nat ACL Port     Status
9150/9150                  10.10.9.2        D          5600     Unmonitored
9050/9050                  172.31.15.4      D          4002     Unmonitored
2 sip peers [Monitored: 0 online, 0 offline Unmonitored: 2 online, 0 offline]
```

Figura 3.138 Extensiones registradas en LOJ

- Se comprueba el correcto establecimiento de llamada en el sistema de telefonía IP, realizando una llamada desde el cliente TLOJ al cliente TESM.

  - **Host cliente TESM**

```
------------------------------ Scenario Screen -------- [1-9]: Change Screen --
  Port   Total-time  Total-calls  Transport
  4001     112.04 s            1  UDP

  0 new calls during 1.028 s period      128 ms scheduler resolution
  0 calls                                Peak was 1 calls, after 34 s
  0 Running, 1 Paused, 3 Woken up
  6 dead call msg (discarded)
  3 open sockets

                                 Messages  Retrans   Timeout   Unexpected-Msg
  ----------> INVITE                   1        0         0         0

  <---------- 100                      1        0
  <---------- 180                      1        0
  <---------- 200                      1        0
  ----------> ACK                      1        0         0         0

  ----------> BYE                      1        0         0         0

  <---------- ACK                      1        0
------------------------------ Sipp Server Mode -------------------------------
```

Figura 3.139 Proceso de llamada en TESM sin nodo GYE y CUE

- **Host cliente TLOJ**

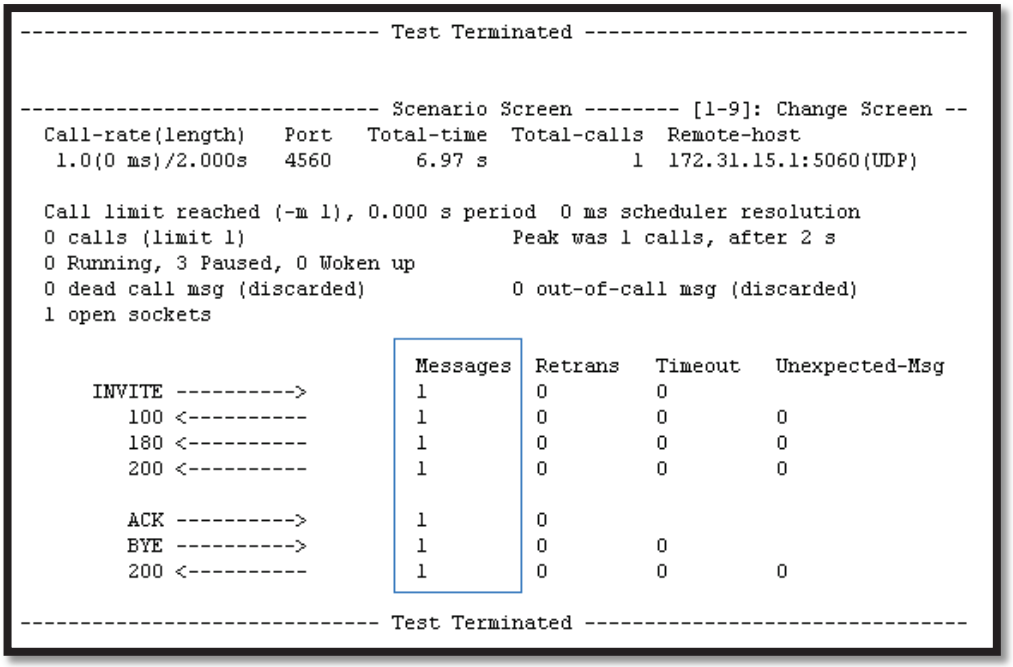


Figura 3.140 Proceso de llamada en TLOJ sin nodo GYE y CUE

Al fallar dos nodos en la red, en este caso Guayaquil (GYE) y Cuenca (CUE), el sistema de telefonía IP continúa garantizando el establecimiento de la comunicación, es decir de la llamada entre dos usuarios en el sistema, este hecho lo confirman las figuras 3.139 y 3.140 de las instancias SIP TESM y TLOJ. En el caso de fallar nodos en el sistema de telefonía IP, únicamente se garantiza el éxito de la comunicación entre dos usuarios de la red cuando fallan como máximo dos nodos que forman parte de la ruta de conexión, al fallar tres nodos el sistema se queda sin rutas alternas para ofrecer el servicio.

Al igual que en todas las pruebas expuestas se realiza el diagrama de flujo de la prueba y su esquema de red.

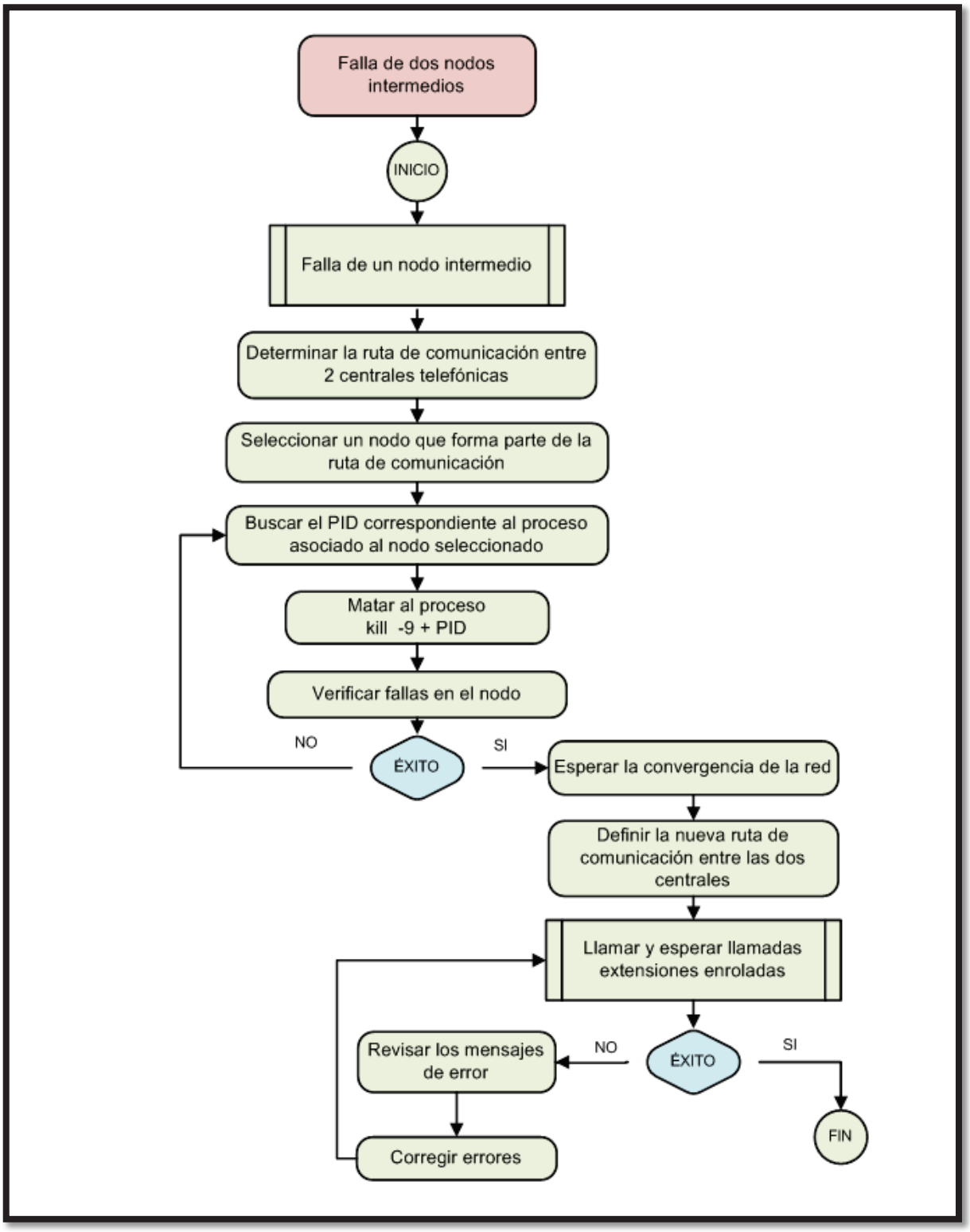
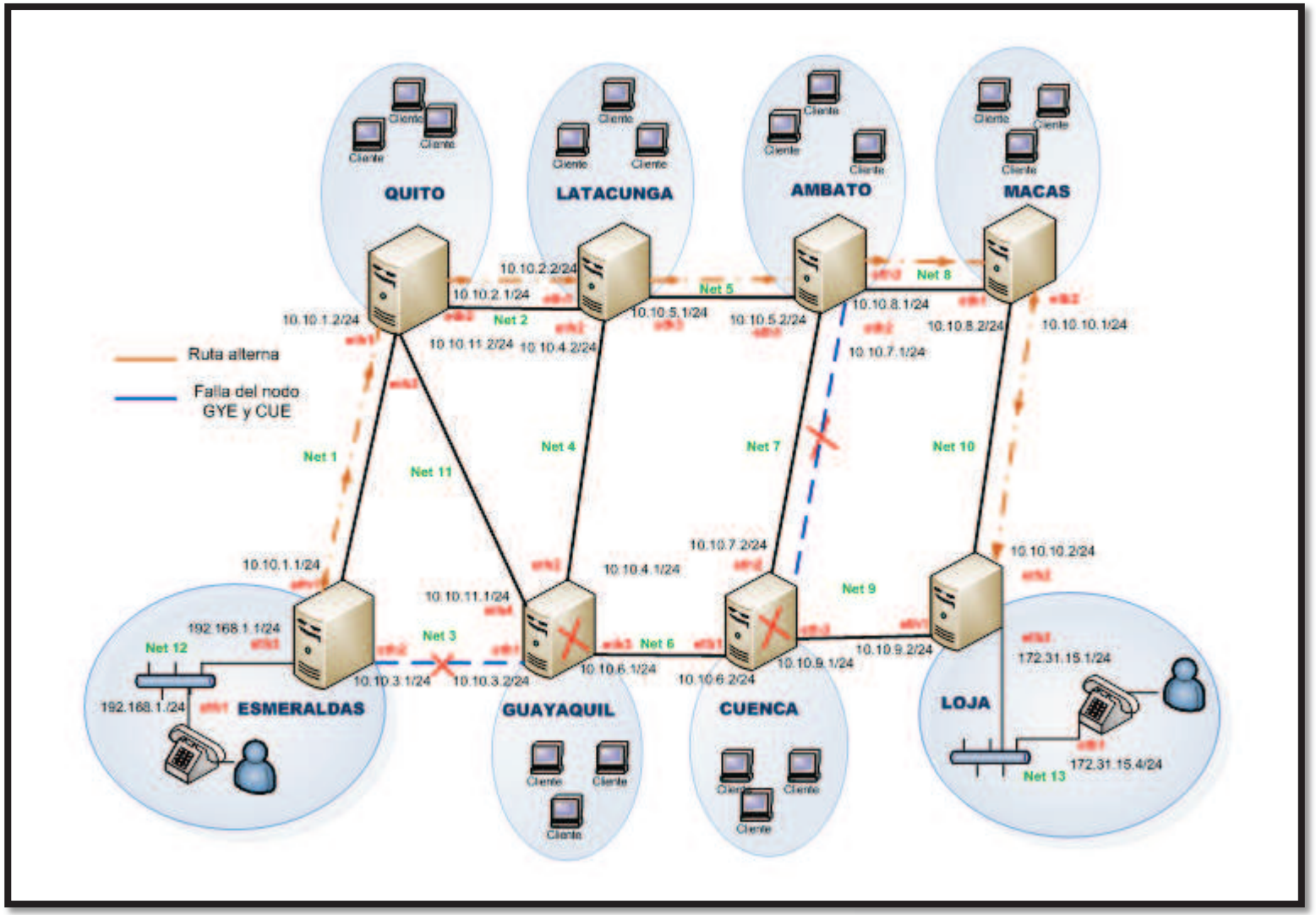


Figura 3.141 Diagrama de flujo de la prueba falla de dos nodos intermedios en el sistema de telefonía IP

Figura 3.142 Falla de dos nodos en la red (nodo GYE y CUE)



De este modo se llega al fin de las pruebas en el Prototipo del sistema de telefonía IP, cumpliendo con los objetivos planteados previos a la realización de este proyecto.

Las pruebas realizadas confirman la funcionalidad de Asterisk y el protocolo *DUNDI* sobre el esquema de red, obteniendo como resultado redundancia en el sistema de telefonía IP y movilidad de los usuarios entre las distintas centrales telefónicas con la funcionalidad adicional del servidor de redirección. Los casos de falla analizados permiten establecer la operatividad del sistema frente a problemas críticos de enlaces de red dañados y centrales telefónicas fuera de uso, afirmando a partir de los resultados, el correcto funcionamiento del servicio de telefonía IP en el sistema al fallar como máximo tres enlaces de red simultáneamente, y dos nodos de red que formen parte de la ruta de comunicación entre los usuarios que deseen establecer una conexión.

En un entorno real de trabajo a nivel empresarial siempre es recomendable realizar un respaldo (*backup*) de los servidores, en este caso de las centrales telefónicas, para minimizar los problemas de falla en los servicios cuando un servidor sufre daños. Una de las herramientas que se puede implementar en el sistema para garantizar un servicio de alta disponibilidad, tomando en cuenta el sistema operativo (Linux) que reside en las centrales telefónicas es *heartbeat. Heartbeat* provee una infraestructura de clúster entre dos servidores (original y respaldo) para mantener activos los servicios alos clientesque dependen de ellos.Este programa ofrece detección de servidores caídos, comunicación y gestión del clúster, de tal modo que cuando un servidor falle el respaldo entre en funcionamiento. En el caso de este proyecto se analizó el caso crítico de no poseer un servidor de respaldo en cada nodo, pues su objetivo fue verificar el descubrimiento de rutas de llamadas en el sistema de telefonía IP, en un ambiente de falla total del nodo únicamente por motivos prácticos. Para implementar el prototipo analizado se debe considerar esta observación, claro está que la decisión de adquirir un servidor de respaldo depende de la empresa que requiera implementar el servicio.

CAPÍTULO IV

COSTO REFERENCIAL DEL SISTEMA DE TELEFONÍA IP

4.1 INTRODUCCIÓN

Previo a evaluar el costo referencial que resultaría de implementar el prototipo del sistema de telefonía IP en un ambiente empresarial, es necesario detallar los conceptos bajo los cuales será determinado el costo.

4.1.1 CLASIFICACIÓN DE COSTOS

4.1.1.1 Costosde Inversión

4.1.1.1.1 Inversiones en activos fijos

Son todas aquellas que se realizan en los bienes tangibles que se utilizarán en el proceso de transformación de los insumos o que sirvan de apoyo a la operación normal del proyecto, son activos fijos entre otros los terrenos, las obras físicas, el equipamiento de la planta, oficinas (maquinaria, muebles, herramientas, vehículos) y la infraestructura de servicios de apoyo (agua potable, desagües, red eléctrica, etc.). ref [15]

4.1.1.1.2 Inversiones en activos nominales o diferidos

Son todas aquellas que se realizan sobre activos constituidos por los servicios o derechos adquiridos necesarios para la puesta en marcha del proyecto. Los principales ítems que configuran esta inversión son los gastos de organización, las patentes y licencias, los gastos de puesta en marcha, capacitación, los imprevistos, los intereses y cargos financieros pre-operativos, etc. ref [15]

4.1.1.1.3 Inversiones en capital de trabajo

Constituye el conjunto de recursos necesarios, en la forma de activos corrientes, para la operación normal del proyecto durante un ciclo productivo, para una capacidad y tamaño determinados. ref [15]

4.1.1.2 CostosOperativos

4.1.1.2.1 Gastos técnicos y administrativos

Son aquellos que representan la estructura ejecutiva, técnica y administrativa de una empresa, tales como, jefes de compras, almacenistas, mecánicos, veladores, dibujantes, ayudantes, mozos de limpieza y envíos, etc.[ref [16]]

4.1.1.2.2 Alquileres y/o depreciaciones

Son aquellos gastos por conceptos de bienes muebles e inmuebles, así como servicios necesarios para el buen desempeño de las funciones ejecutivas, técnicas y administrativas de una empresa, tales como: rentas de oficinas y almacenes, servicios de teléfonos, etc.[ref [16]]

4.1.1.2.3 Obligaciones y seguros

Son aquellos gastos obligatorios para la operación de la empresa y convenientes para la dilución de riesgos a través de seguros que impidan una súbita descapitalización por siniestros.[ref [16]]

4.1.1.2.4 Materiales de consumo

Son aquellos gastos en artículos de consumo, necesarios para el funcionamiento de la empresa, tales como: combustibles y lubricantes de automóviles y camionetas al servicio de las oficinas de la planta, gastos de papelería impresa, artículos de oficina, etc.[ref [16]]

4.1.1.2.5 Capacitación y Promoción

Todo colaborador tiene el derecho de capacitarse y pensamos en que tanto éste lo haga, en esa misma medida o mayor aún, la empresa mejorará su productividad. Entre los gastos de capacitación y promoción se puede mencionar: cursos a obreros y empleados, gastos de actividades deportivas, de celebraciones de oficinas, etc.[ref [16]]

Ref [15] http://eqaula.org/eva/mod/resource/view.php?id=6112

Ref [16] http://www.ingenieria.unam.mx/~jkuri/Apunt_Planeacion_internet/TEMAVI.4.pdf

Para determinar el costo referencial del prototipo se analizarán únicamente los costos de inversión, con el fin de presentar una propuesta económica detallada de la inversión general que se debe realizar para implementar el prototipo del sistema de telefonía IP, en un ambiente empresarial. Por esta razón no se estudiarán los costos operativos.

4.2 REQUERIMIENTOS DE TELEFONÍA

El prototipo desarrollado en este proyecto se implementó sobre una red en malla, compuesta por ocho nodos, los cuales simulan estar ubicados en diferentes ciudades del Ecuador. Con el fin de dimensionar los equipos que se requieren para implementar el Prototipo del Sistema de Telefonía IP en un ambiente empresarial se asume las siguientes consideraciones:

- La solución de telefonía IP se implementará sobre la red de datos de la empresa, la cual está funcional y operativa. La velocidad de transmisión de los enlaces WAN es 100 [Mbps].
- Cada sucursal tendrá su propia central de telefonía IP basada en Asterisk 1.4.
- El máximo número de usuarios al que se dará servicio por cada central de telefonía IP, será cincuenta (50).
- El 50 % de los usuarios en el sistema de telefonía IP emplearán para la comunicación teléfonos IP a nivel de hardware, el otro 50 % seguirá utilizando los mismos teléfonos tradicionales analógicos, empleando para su funcionamiento convertidores ATA[15].
- Los usuarios del sistema de telefonía IP, además de comunicarse internamente entre todos los miembros del sistema se comunicarán con la PSTN (*Public Switched Telephone Network*) de la CNT (*Corporación Nacional de Telecomunicaciones*), con el fin de realizar llamadas locales, regionales, internacionales y a la telefonía celular.

ATA [15] Es un adaptador analógico para telefonía IP, que permite convertir un teléfono analógico Standard o un FAX cualquier marca, en un sistema de telefonía VoIP (Voz sobre IP / Voz sobre Internet).

4.3 CLASIFICACIÓN DE COSTOS Y DIMENSIONAMIENTO DE EQUIPOS

De acuerdo al diseño de red establecido para el sistema de telefonía IP y de sus requerimientos se dimensionan los siguientes equipos, para llevar a cabo la implementación:

COSTOS DE INVERSION			
TIPO	**DESCRIPCIÓN**	**APLICA**	**CANTIDAD**
ACTIVOS NOMINALES	Licencias de software	No	X
	Patentes	No	X
ACTIVOS FIJOS	Servidores	Si	8
	Teléfonos IP	Si	200
	Tarjetas interfaces *FXO*	Si	8
	Convertidores ATA	Si	100
	Líneas telefónicas *CNT*	Si	40
CAPITAL DE TRABAJO	Aporte de accionistas	No	X

Tabla 4.1 Clasificación de costos de inversión

4.4 CARACTERÍSITICAS DE LOS EQUIPOS

4.4.1 CENTRAL DE TELEFONÍA IP

El sistema operativo y las aplicaciones adicionales que se instalarán en cada central telefónica, en este caso Ubuntu 9.10, Asterisk 1.4 y SIPP, no consumen en gran medida los recursos del servidor, sin embargo se deben destinar recursos adicionales para el procesamiento de las solicitudes de los usuarios. En función del número de usuarios a los cuales se dará servicio, y por ende el número de solicitudes que éste recibirá y deberá procesar, se dimensiona la memoria RAM y el procesador del servidor, considerando además la versión Asterisk que se utilizará(Las referencias utilizadas para este análisis de detallan en el anexo 5.1).

De este análisisse establecen las siguientes características mínimas que debe poseer el servidor para garantizar un buen servicio.

CARÁCTERISTICAS DEL SERVIDOR
• Procesador Intel® Core™2 Duo
• Memoria RAM de 8 GB
• Disco Duro de 120 GB
• Tarjeta de Red 10/100 Mbps

Tabla 4.2 Características del servidor IP

El servidor que cumple con estos requisitos es el ***HP ProLiant ML310 G5p Server***, además de tener un costo conveniente tomando en cuenta la garantía a nivel mundial de la marca reconocida ***HP (***Hewlett-Packard***).***La descripción del servidor seleccionado se detalla en el anexo 5.1.

Figura 4.1 Servidor HP ProLiant ML 310 G5p

4.4.2 TELÉFONO IP

La selección del teléfono IP depende de los servicios finales que se prestarán al usuario tales como: llamada en espera, transferencia de llamada, buzón de voz entre otros. De acuerdo a ello se establecen las siguientes características básicas.

CARÁCTERISTICAS DEL TELÉFONO IP
Soporte para codecs G711 (A-law y u-law)
Puerto Ethernet RJ45 100 Mbps
Indicador de voicemail

Tabla 4.3 Características del Teléfono IP

En función de las características planteadas el teléfono IP seleccionado es el ***Grandstream BT-200*** cuyas especificaciones se adjuntan en el anexo 5.2.

Figura 4.2 Grandstream BT-200

4.4.3 TARJETA FXO

La central de telefonía IP para permitir cursar tráfico telefónico local, nacional, regional, internacional y de telefonía celular deberá estar conectada a la PSTN de CNT. Para ello se emplea una tarjeta con interfaces FXO (*Foreign Exchange Office*), la cual se conecta en la ranura PCI [15] (*Peripheral Component Interconnect*) de la central telefónica. En las interfaces *FXO* de la tarjeta se conectarán las líneas telefónicas proporcionadas por la *CNT,* de este modo la telefonía IP interactúa con la telefonía tradicional ref[17].

Para determinar el número de líneas telefónicas que se deben solicitar a la *CNT,* y con este resultado el número de interfaces *FXO* que debe poseer la tarjeta, se debe realizar un análisis de la intensidad de tráfico telefónico que generarán los

usuarios de cada central telefónica, con un mínimo de pérdida establecido. Se aplica entonces la fórmula *Erlang B* y sus respectivas tablas para calcular el número de troncales que se requieren.

- **Cálculo del tráfico en erlangs**

Para realizar el cálculo de erlangs[16] se utiliza la siguiente fórmula:

$$A = \frac{1}{T}(n * m) = \#[erlangs]$$

Fórmula 4.1 ref [17]

VARIABLES	DESCRIPCIÓN	VALORES ASUMIDOS
T	Período de evaluación	1 hora = 3600 [segundos]
n	Número de llamadas total realizadas en un período T	25 llamadas (50% de los usuarios utilizan el servicio simultáneamente)
m	Tiempo de duración promedio de una llamada	3 minutos = 180[segundos]

Tabla 4.4 Tabla de variables utilizados en la fórmula de Erlang

Para este el cálculo se asumirán los valores tomando en cuenta un ambiente empresarial estándar, debido a que el Prototipo realizado constituye una simulación que no puede ser monitoreada.

Al reemplazar los valores se obtiene:

$$A = \frac{1}{T}(n * tm) = \frac{1}{3600[s]}(25 * 180[s]) = 1.25[erlangs]$$

Fórmula 4.2 ref [17]

Determinado la cantidad de erlangs, se debe asignar el grado de calidad de servicio (*GOS*) que se brindará al sistema de telefonía IP. Se asume una calidad de servicio de 0.01 la cual corresponde a 1 llamada perdida por cada 100 llamadas realizadas (lo cual resulta tolerable en un ambiente empresarial).

Con estos datos se verifica tabla Erlang B para comprobar el número de líneas telefónicas necesarias (tabla anexo 5.3), se utiliza para este fin la aplicación Erlang B Calculator disponible en internet:

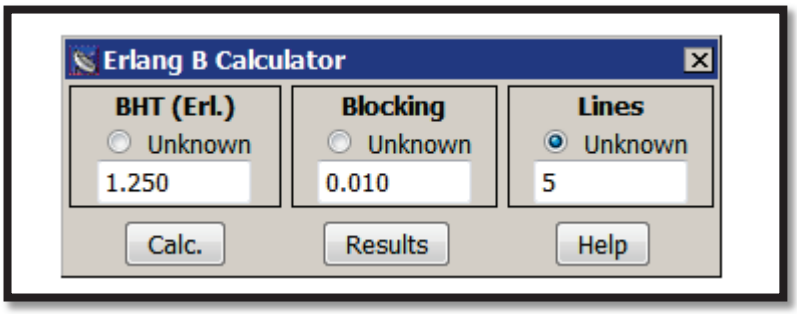


Figura 4.3 Cálculo de Erlang B Ref [18]

Como resultado final cada central requiere 5 líneas telefónicas de la *CNT,* es decir una tarjeta con al menos 5 interfaces *FXO.*

CARACTERÍSTICAS DE TARJETA
Tarjeta de comunicación para sistemas Asterisk
5 Puertos FXO

Tabla 4.5 Características de la tarjeta

Al analizar el precio y características se escoge como mejor solución la tarjeta ***TDM800P – Digium***. En el mercado se dispone de tarjetas de 2, 4 y 8 interfaces *FXO*, se seleccionó la tarjeta de 8 interfaces para cumplir con los requisitos expuestos en la tabla 4.5. La descripción de esta tarjeta se detalla en el anexo5.4.

PCI [15] (Interconexión de Componentes Periféricos) Consiste en un bus de ordenador estándar para conectar dispositivos periféricos directamente a su placa base.

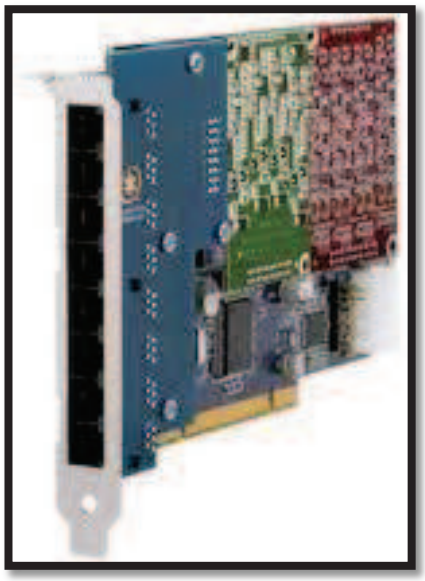


Figura 4.4 TDM800P – Digium

4.4.4 ADAPTADOR ATA

Un adaptador ATA transforma la telefonía convencional en telefonía IP, este dispositivo permite conectar dos teléfonos analógicos al sistema de telefonía IP. Las características del adaptador son:

CARÁCTERISTICAS DEL ADAPTADOR ATA
2 Puertos telefónicos RJ11
Puerto Ethernet RJ45
Soporte para codecs G711 (A-law y u-law)

Tabla 4.6 Características del convertidor ATA

Erlang [16] Es una unidad adimensional utilizada en telefonía como una medida estadística del volumen de tráfico.

Ref [17] http://dspace.epn.edu.ec/bitstream/123456789/550/5/T10467CAP2.pdf

Ref [18] http://www.erlang.com/calculator/erlb/

En el mercado existen convertidores ATA de dos marcas reconocidas CISCO y LINKSYS que cumplen con estas características, por motivos de precio se toma como mejor opción el convertidor ***LINKSYS PAP 2 NA***, sus especificaciones se describen en el anexo 5.5.

Figura 4.5 ATA Linksys PAP2-NA

4.5 COSTO GENERAL DEL SISTEMA DE TELEFONÍA IP

4.5.1 PROFORMA DE LOS EQUIPOS

CANTIDAD	EQUIPO	PRECIO UNITARIO	PRECIO TOTAL
8	Servidores ProLiant ML310 G5p server HP	905,0	7240,0
200	Teléfonos IP Grandstream	129,0	25.800,00
8	Tarjetas analógicasTDM800PDigium	980,0	7.840,0
100	Convertidores ATA PAP2-NA	70,0	7.000,0
	Subtotal		47.880,0
Iva 12%			5.745,6
Total USD $			53.625,6

Tabla 4.7 Proforma de equipos

Los costos fueron tomados como referencia de las proformas solicitadas a SIDEVOX SA, una empresa ecuatoriana dedicada a la venta configuración de equipos de telefonía IP, y de los precios ofertados en Internet para el Ecuador. Las proformas respectivas se adjuntan en el anexo5.6.

4.5.2 PROFORMA DE CONFIGURACIÓN DE LOS EQUIPOS

CANTIDAD	DESCRIPCIÓN	PRECIO UNITARIO	PRECIO TOTAL
8	Configuración de Central de Telefonía IP	1.000,0	8.000,0
200	Configuración de teléfono IP	12,0	2.400,0
100	Instalación de convertidor*ATA PAP2-NA*	10,0	1.000,0
8	Instalación de tarjeta*FXO*	45,0	360,0
Subtotal			11.760,0
Iva 12%			1.411,2
		Total USD $	13.171,2

Tabla 4.8 Proforma de configuración de equipos

Para justificar el precio de configuración asignado en la proforma, se realiza el desglose de los costosasociados a la implementación del sistema de telefonía IP. Se asume un tiempo de duración de la obra de 2 meses y la contratación de 20 técnicos operativos para brindar soporte y realizar las instalaciones y las configuraciones en las ocho ciudades que forman la red. Cada técnico recibe su remuneración salarial, gastos de transporte y alimentación.

CANTIDAD	DESCRIPCIÓN	PRECIO UNITARIO	PRECIO TOTAL
1	Servidor de pruebas	700,0	700,0
32	Servicios profesionales	60,0	1.920,0
20	Mano de obra del personal de soporte	350,0	7.000,0
1	Movilización del personal	800,0	800,0
1	Alimentación del personal	1.280,0	1.280,0
1	Publicidad en Internet	60,0	60,0
Subtotal			11.760,0
		Iva 12%	1.411,2
		Total USD $	13.171,2

Tabla 4.9 Proforma de costos previstos para la configuración de equipos

4.5.3 PROFORMA GENERAL

CANTIDAD	DESCRIPCIÓN	PRECIO UNITARIO	PRECIO TOTAL
8	Servidores ProLiant ML310 G5p server HP	905,0	7.240
200	Teléfonos IP Grandstream	129,0	25.800,0
8	Tarjetas TDM800PDigium	980,0	7.840,0
100	Convertidores ATA PAP2-NA	70,0	7.000,0
8	Configuración de Central de Telefonía IP	1.000,0	8.000,0
200	Configuración de teléfono IP	12,0	2.400,0
100	Instalación de convertidor*ATA PAP2-NA*	10,0	1.000,0
		Subtotal	59.280,0
		Iva 12%	7.113,6
Total USD $			66.393,60

Tabla 4.10 Proforma general de costos del sistema de telefonía IP

Como resultado se obtiene el costo total de 66.393,60 dólares, para llevar a cabo la implementación del prototipo de telefonía IP en una empresa con las características descritas en el literal 4.2.

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

- El prototipo desarrollado e implementado, permitió estudiar el funcionamiento de un sistema de telefonía IP distribuido que brinda movilidad a los usuarios, mediante el descubrimiento de rutas de llamadas y la aplicación de un servidor de redirección. La aplicación principal del prototipo, radica en la fase de estudio del funcionamiento del sistema de telefonía IP previo a instalación física de los dispositivos, pues permite probar, modificar y eliminar configuraciones periódicamente con el fin de cubrir los requerimientos del cliente. Por esta razón se puede afirmar que la realización del prototipo resulta útil y óptima al momento de brindar soluciones de telefonía IP, que se ajusten a las necesidades de empresas y entidades públicas.

- El esquema de configuración propuesto en el prototipo del sistema de telefonía IP, representa una solución aplicable a una red de nodos o enlaces intermitentemente activos, que gracias a la distribución redundante permite el establecimiento de las llamadas extremo a extremo siempre que exista una ruta disponible, tanto a nivel de red como a nivel de llamada.

- El prototipo presentado en este proyecto, tiene la limitante de no permitir realizar pruebas de dimensionamiento de tráfico telefónico en un ambiente empresarial real, por esta razón en el Capítulo IV fue necesario asumir valores referenciales tomados de estudios estadísticos en esquemas similares.

- A pesar de que ENUM resulta ser un mecanismo sencillo y conciso para la localización de *gateways* en Internet hacia servicios de telefonía, el monopolio de registradores de la información de mapeo, permite que se

implementen esquemas demasiado centralizados en relación a esta autoridad. Este monopolio en el registro deriva en la imposición de tarifas, impuestos y una exagerada dependencia del funcionamiento de la entidad que administra estos servicios.

- El protocolo DUNDi es una aproximación excelente para agregar la funcionalidad de clúster y confiabilidad en un esquema distribuido de nodos, sin la necesidad de depender de un registrador único de mapeo.

- La implementación de DUNDi representa un complemento perfecto del protocolo de enrutamiento dinámico RIP pues compensa la capacidad de descubrimiento de rutas de red con la de descubrimiento de rutas de llamada.

- *VNUML* es una herramienta de virtualización *open source* que permite la definición rápida de escenarios de red complejos. Resulta una herramienta de gran ayuda para la realización de pruebas de aplicaciones y servicios de red sobre esquemas complejos conformados incluso por decenas de nodos dentro de una sola máquina física.

- La flexibilidad de la solución *open source,* Asterisk, permite que, con el apoyo de otras aplicaciones compatibles como *SIPP*, se agreguen funcionalidades adicionales que de manera posterior podrían integrarse directamente al código fuente.

- La movilidad a nivel de usuario se consiguió mediante la utilización de *SIPP*y la implementación de un servicio adicional en los nodos de Asterisk que permita la redirección de las llamadas hacia el número de extensión personal, donde se encuentre enrolado el usuario; evitando de esta manera que la asignación estática que representa el registro inicial del dispositivo cliente, restrinja la posibilidad de movimiento del usuario a lo largo del dominio de VoIP.

- La implementación de movilidad y enrutamiento de llamada, permite que el sistema sea más fácilmente accesible para el usuario final, pues éste maneja un número de identificación único en todo el dominio DUNDi cuya ubicación depende solamente del proceso de enrolamiento y en caso de falla de un enlace o nodo de la red en malla, permite la recuperación de las rutas.

- *SIPP* constituye una excelente herramienta*open source* para la realización de pruebas de carga sobre dispositivos de VoIP que manejen protocolo SIP. Asimismo, la gran flexibilidad de su configuración en base a archivos XML, permite una evaluación muy granular de la funcionalidad de SIP.

- *SIPP* es la base, en este esquema, de la implementación de movilidad, permitiendo la simulación de los procesos de registro y des-registro de extensiones y enrolamiento y des-enrolamiento de usuarios. Asimismo permitió la simulación de las llamadas, en función de las cuales se logró hacer las pruebas pertinentes de interconexión a nivel de VoIP. En conclusión *SIPP* permitió una simulación bastante completa de los dispositivos cliente de VoIP.

- *VNUML* a diferencia de otras herramientas de virtualización brinda la gran ventaja de simular y probar complejos esquemas de red en un solocomputador anfitrión, al aprovechar y optimizar los recursos del computador. Además al ser *VNUML* un software libre, probar los prototipos de simulación tales como el descrito en el desarrollo de este proyecto resulta totalmente gratuito para los usuarios.

- Las pruebas respecto de la funcionalidad del sistema se realizaron con éxito, permitiendo comprobar los procesos de enrolamiento de usuarios en varios nodos simulados, asociando su número personal a diferentes dispositivos físicos (simulados igualmente) a través de los servicios de redirección en cada uno de los nodos de VoIP.

- El gran potencial de DUNDi actualmente se encuentra en escenarios de recuperación de desastres, gracias a la facilidad que brinda en la replicación de planes de marcado en un esquema de funcionalidad completamente basada en VoIP.

- Lamentablemente, la adopción de DUNDi como producto que revolucione el *peering* basado en VoIP se ve menguada por el interés de empresas gigantescas que manejan un monopolio cerrado del estándar ENUM. Por esa razón, el proceso de adopción a gran escala es muy lento.

5.2 RECOMENDACIONES

- Es recomendable utilizar *VNUML* para la creación de entornos de red virtuales que necesiten ser flexibles, sencillos y de eficiente consumo de recursos. Es, sin lugar a dudas una herramienta didáctica y de simulación que vale la pena incorporar a las prácticas sobre redes y sistemas operativos en la universidad.

- La herramienta *SIPP* es una aplicación muy útil para realizar pruebas automatizadas de carga y funcionalidad de servicios de telefonía IP basados en SIP. Estas funciones están desde luego potencializadas por la flexibilidad que nos brinda la línea de comandos de Linux y la sintaxis de los archivos XML. Sin embargo es recomendable probar la funcionalidad de la interfaz gráfica de *SIPP*, *SIPP* GUI, para evaluar los beneficios que esta herramienta podría brindar.

- El tiempo que toma el esquema simulado de 8 nodos en inicializarse es de 300 segundos en promedio, siempre y cuando los servicios principales (asterisk, quagga) estén desactivados. Es recomendable iniciar los servicios luego de que el entorno de red se haya iniciado para evitar el retardo de arranque.

- En el escenario propuesto se utiliza como protocolo de enrutamiento en capa 3 al protocolo RIP, implementado a través de la herramienta quagga, sin embargo, sería recomendable probar con OSPF en caso de disponer de un esquema de malla más complejo.

- En el caso de que la capacidad física del *host* no sea suficiente, se recomienda el uso de un equipo *host* adicional, pues *VNUML* permite la interconexión de esquemas virtuales, a través de un enlace físico existente. Se puede consultar la página web de *VNUML* para mayor información relacionada.

- Al arrancar el escenario de simulación mediante el comando vnumlparser.pl es recomendable utilizar la opción –w configurada a cero, (-w=0) la cual permite que no se llegue al tiempo límite de simulación establecido por defecto en VNUML, de este modo se logra evitar que se detenga la creación del escenario, en el caso de poseer un esquema de red complejo con varios nodos interconectados, el cual necesita un tiempo prudencial para terminar todos los procesos.

- Se recomienda también el uso de la solución de virtualización XEN para la evaluación del consumo de recursos en un esquema equivalente, y que permita una comparación más sustentada entre estas dos soluciones *open source.*

- En un entorno no simulado y de producción, se recomienda agregar mensajes de voz para la confirmación a los usuarios de los procesos de enrolamiento y des-enrolamiento, de modo que estos estén conscientes cuando la operación se ejecutó con éxito o cuando no.

- En relación a los *scripts* de registro de extensión y llamada sería recomendable fusionarlos en uno solo pues el registro de la extensión normalmente implica que el dispositivo cliente (en este caso simulado por varias instancias de *SIPP*) estará listo para recibir una llamada.

- En el esquema de simulación, del mismo modo como se lo hace en un escenario de red físico, es recomendable la detección de errores probando la funcionalidad desde las capas bajas del modelo OSI hacia las superiores.

- Se recomienda implementar un sistema de *firewall* que impida que conexiones no permitidas tengan acceso a los nodos en cada una de las sucursales.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

CAPÍTULO I

CAPÍTULO II

CAPÍTULO III

CAPÍTULO IV

ÍNDICE DE FIGURAS

CAPÍTULO I