Guía de la API REST

← Volver a la Documentación Principal

Esta guía proporciona documentación completa para la **API REST** de OmniSS7 y **Swagger UI**.

Tabla de Contenidos

Descripción General

OmniSS7 proporciona una API REST para el acceso programático a las operaciones MAP (Mobile Application Part). La API permite:

- Enviar solicitudes MAP (SRI, SRI-for-SM, UpdateLocation, etc.)
- Recuperar respuestas MAP
- Monitorear métricas del sistema a través de Prometheus

Arquitectura de la API

Configuración del Servidor HTTP

Detalles del Servidor

Parámetro	Valor	Configurable
Protocolo	HTTP	No
Dirección IP	0.0.0.0 (todas las interfaces)	Solo a través de código
Puerto	8080	Solo a través de código
Transporte	Plug.Cowboy	No

URL de Acceso: `http://[server-ip]:8080`

Habilitar/Deshabilitar el Servidor HTTP

Controla si el servidor HTTP se inicia:

```
config :omniss7,
  start_http_server: true  # Establecer en false para deshabilitar
```

Predeterminado: `true` (habilitado)

Cuando está Deshabilitado: El servidor HTTP no se iniciará, y la API REST/Swagger UI no estará disponible.

Swagger UI

La API incluye un **Swagger UI** para documentación interactiva de la API y pruebas.

Accediendo a Swagger UI

URL: `http://[server-ip]:8080/swagger`

Características:

- Documentación interactiva de la API
- Funcionalidad de prueba para probar puntos finales
- Esquemas de solicitud/respuesta
- Cargas útiles de ejemplo

Swagger JSON

La especificación OpenAPI está disponible en:

URL: `http://[server-ip]:8080/swagger.json`

Casos de Uso:

- Importar en Postman u otros clientes de API
- Generar bibliotecas de cliente
- Automatización de documentación de API

Puntos Finales de la API

Todos los puntos finales de operaciones MAP siguen el patrón: `POST /api/{operation}`

Resumen de Puntos Finales

Punto Final	Método	Propósito	Tiempo de Espera
`/api/sri`	POST	Enviar Información de Enrutamiento	10s
`/api/sri-for-sm`	POST	Enviar Información de Enrutamiento para SM	10s
`/api/send-auth-info`	POST	Enviar Información de Autenticación	10s
`/api/MT-forwardSM`	POST	Enviar SMS Terminado a Móvil	10s
`/api/forwardSM`	POST	Reenviar SMS	10s
`/api/updateLocation`	POST	Actualizar Ubicación	10s
`/api/prn`	POST	Proporcionar Número de Roaming	10s
`/api/ussd/send`	POST	Envío de USSD originado en la red	10s
`/metrics`	GET	Métricas de Prometheus	N/A
`/swagger`	GET	Swagger UI	N/A
`/swagger.json`	GET	Especificación OpenAPI	N/A

Nota: Todas las solicitudes MAP tienen un **tiempo de espera de 10 segundos codificado**.

SendRoutingInfo (SRI)

Recuperar información de enrutamiento para establecer una llamada a un suscriptor móvil.

Punto Final: `POST /api/sri`

Cuerpo de la Solicitud:

```
{
  "msisdn": "1234567890",
  "gmsc": "5551234567"
}
```

Parámetros:

Campo	Tipo	Requerido	Descripción
`msisdn`	String	Sí	MSISDN de la parte llamada
`gmsc`	String	Sí	Título Global del MSC de la puerta de enlace

Respuesta (200 OK):

```
{
  "result": {
    "imsi": "001001234567890",
    "msrn": "5551234999",
    "vlr_number": "5551234800",
    ...
  }
}
```

Error (504 Gateway Timeout):

```
{
  "error": "timeout"
}
```

Ejemplo de cURL:

```
curl -X POST http://localhost:8080/api/sri \
  -H "Content-Type: application/json" \
  -d '{
    "msisdn": "1234567890",
    "gmsc": "5551234567"
  }'
```

SendRoutingInfoForSM (SRI-for-SM)

Recuperar información de enrutamiento para entregar un SMS a un suscriptor móvil.

Punto Final: `POST /api/sri-for-sm`

Cuerpo de la Solicitud:

```
{
  "msisdn": "1234567890",
  "service_center": "5551234567"
}
```

Parámetros:

Campo	Tipo	Requerido	Descripción
`msisdn`	String	Sí	MSISDN de destino
`service_center`	String	Sí	Título Global del Centro de Servicio

Respuesta (200 OK):

```
{
  "result": {
    "imsi": "001001234567890",
    "msc_number": "5551234800",
    "location_info": {...},
    ...
  }
}
```

Ejemplo de cURL:

```
curl -X POST http://localhost:8080/api/sri-for-sm \
  -H "Content-Type: application/json" \
  -d '{
    "msisdn": "1234567890",
    "service_center": "5551234567"
  }'
```

SendAuthenticationInfo

Solicitar vectores de autenticación para un suscriptor.

Punto Final: `POST /api/send-auth-info`

Cuerpo de la Solicitud:

```
{
  "imsi": "001001234567890",
  "vectors": 3
}
```

Parámetros:

Campo	Tipo	Requerido	Descripción
`imsi`	String	Sí	IMSI del suscriptor
`vectors`	Integer	Sí	Número de vectores de autenticación a generar

Respuesta (200 OK):

```
{
  "result": {
    "authentication_sets": [
      {
        "rand": "0123456789ABCDEF...",
        "xres": "...",
        "ck": "...",
        "ik": "...",
        "autn": "..."
      }
    ],
    ...
  }
}
```

Ejemplo de cURL:

```
curl -X POST http://localhost:8080/api/send-auth-info \
  -H "Content-Type: application/json" \
  -d '{
    "imsi": "001001234567890",
    "vectors": 3
  }'
```

MT-ForwardSM

Entregar un SMS Terminado a Móvil a un suscriptor.

Punto Final: `POST /api/MT-forwardSM`

Cuerpo de la Solicitud:

```
{
  "imsi": "001001234567890",
  "destination_service_centre": "5551234567",
  "originating_service_center": "5551234568",
  "smsPDU": "0001000A8121436587F900001C48656C6C6F20576F726C64"
}
```

Parámetros:

Campo	Tipo	Requerido	Descripción
`imsi`	String	Sí	IMSI del suscriptor de destino
`destination_service_centre`	String	Sí	GT del centro de servicio de destino
`originating_service_center`	String	Sí	GT del centro de servicio de origen
`smsPDU`	String	Sí	SMS TPDU en formato hexadecimal

Nota: `smsPDU` debe ser una cadena codificada en hex (mayúsculas o minúsculas).

Respuesta (200 OK):

```
{
  "result": {
    "delivery_status": "success",
    ...
  }
}
```

Ejemplo de cURL:

```
curl -X POST http://localhost:8080/api/MT-forwardSM \
  -H "Content-Type: application/json" \
  -d '{
    "imsi": "001001234567890",
    "destination_service_centre": "5551234567",
    "originating_service_center": "5551234568",
    "smsPDU": "0001000A8121436587F900001C48656C6C6F20576F726C64"
  }'
```

ForwardSM

Reenviar un mensaje SMS (MO-SMS del suscriptor).

Punto Final: `POST /api/forwardSM`

Cuerpo de la Solicitud: Igual que MT-ForwardSM

Ejemplo de cURL:

```
curl -X POST http://localhost:8080/api/forwardSM \
  -H "Content-Type: application/json" \
  -d '{
    "imsi": "001001234567890",
    "destination_service_centre": "5551234567",
    "originating_service_center": "5551234568",
    "smsPDU": "0001000A8121436587F900001C48656C6C6F20576F726C64"
  }'
```

UpdateLocation

Notificar al HLR del cambio de ubicación del suscriptor (registro VLR).

Punto Final: `POST /api/updateLocation`

Cuerpo de la Solicitud:

```
{
  "imsi": "001001234567890",
  "vlr": "5551234800"
}
```

Parámetros:

Campo	Tipo	Requerido	Descripción
`imsi`	String	Sí	IMSI del suscriptor
`vlr`	String	Sí	Dirección del Título Global del VLR

Respuesta (200 OK):

```
{
  "result": {
    "hlr_number": "5551234567",
    "subscriber_data": {...},
    ...
  }
}
```

Nota: En modo HLR, esto activa la secuencia InsertSubscriberData (ISD) con un tiempo de espera de 10 segundos por ISD.

Ejemplo de cURL:

```
curl -X POST http://localhost:8080/api/updateLocation \
  -H "Content-Type: application/json" \
  -d '{
    "imsi": "001001234567890",
    "vlr": "5551234800"
  }'
```

ProvideRoamingNumber (PRN)

Solicitar MSRN (Número de Roaming de Estación Móvil) para el enrutamiento de llamadas a suscriptores en roaming.

Punto Final: `POST /api/prn`

Cuerpo de la Solicitud:

```
{
  "msisdn": "1234567890",
  "gmsc": "5551234567",
  "msc_number": "5551234800",
  "imsi": "001001234567890"
}
```

Parámetros:

Campo	Tipo	Requerido	Descripción
`msisdn`	String	Sí	MSISDN del suscriptor
`gmsc`	String	Sí	GT del MSC de puerta de enlace
`msc_number`	String	Sí	Número del MSC para el suscriptor
`imsi`	String	Sí	IMSI del suscriptor

Respuesta (200 OK):

```
{
  "result": {
    "msrn": "5551234999",
    ...
  }
}
```

Ejemplo de cURL:

```
curl -X POST http://localhost:8080/api/prn \
  -H "Content-Type: application/json" \
  -d '{
    "msisdn": "1234567890",
    "gmsc": "5551234567",
    "msc_number": "5551234800",
    "imsi": "001001234567890"
  }'
```

USSD Send (Originado en la Red)

Enviar un mensaje USSD a un suscriptor. Requiere `ussd_gateway_enabled: true`. Para la documentación completa del USSD Gateway, incluyendo el protocolo de callback y el ciclo de vida de la sesión, consulte la Guía del USSD Gateway.

Punto Final: `POST /api/ussd/send`

Cuerpo de la Solicitud:

```
{
  "msisdn": "+254712345678",
  "text": "Tienes una factura pendiente. Responde 1 para pagar.",
  "callback_url": "http://billing-app:9000/ussd"
}
```

Parámetros:

Campo	Tipo	Requerido	Descripción
`msisdn`	String	Sí	MSISDN del suscriptor de destino
`text`	String	Sí	Texto USSD a mostrar (solo caracteres GSM de 7 bits)
`callback_url`	String	Sí	URL para recibir respuestas del suscriptor a través de HTTP POST

Respuesta (200 OK):

```
{
  "session_id": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
  "status": "sent"
}
```

Error (503 Servicio No Disponible):

```
{
  "error": "USSD gateway not enabled"
}
```

Ejemplo de cURL:

```
curl -X POST http://localhost:8080/api/ussd/send \
  -H "Content-Type: application/json" \
  -d '{
    "msisdn": "+254712345678",
    "text": "Tienes una factura pendiente. Responde 1 para
pagar.",
    "callback_url": "http://billing-app:9000/ussd"
  }'
```

Autenticación

Estado Actual: La API **no requiere autenticación**.

Consideraciones de Seguridad:

- La API está destinada para uso en redes internas/confiables
- Considere usar reglas de firewall para restringir el acceso
- Para implementaciones en producción, considere implementar middleware de autenticación

---

Formatos de Respuesta

Todas las respuestas utilizan el formato **JSON**.

Respuesta de Éxito

Estado HTTP: 200 OK

Estructura:

```
{
  "result": {
    // Datos de respuesta específicos de la operación
  }
}
```

Respuesta de Error

Estado HTTP:

- 400 Bad Request - Cuerpo de solicitud no válido
- 504 Gateway Timeout - Tiempo de espera de solicitud MAP (10 segundos)
- 404 Not Found - Punto final no válido

Estructura:

```
{
  "error": "timeout"
}
```

o

```
{
  "error": "invalid request"
}
```

Manejo de Errores

Errores Comunes

Error	Código HTTP	Descripción	Solución
JSON Inválido	400	El cuerpo de la solicitud no es un JSON válido	Verifique la sintaxis JSON
Campos Faltantes	400	Faltan campos requeridos	Incluya todos los parámetros requeridos
Tiempo de Espera	504	La solicitud MAP excedió el tiempo de espera de 10s	Verifique la conectividad M3UA, disponibilidad de HLR/VLR
No Encontrado	404	Punto final no válido	Verifique la URL del punto final

Comportamiento de Tiempo de Espera

Todas las solicitudes MAP tienen un **tiempo de espera de 10 segundos codificado**:

1. Solicitud enviada al MapClient GenServer
2. Espera respuesta hasta 10 segundos
3. Si no hay respuesta → devuelve 504 Gateway Timeout
4. Si se recibe respuesta → devuelve 200 OK con resultado

Solución de Problemas de Tiempos de Espera:

- Verifique el estado de conexión M3UA (Interfaz Web → página M3UA)
- Verifique que el elemento de red (HLR/VLR/MSC) sea accesible
- Verifique la configuración de enrutamiento
- Revise los registros de eventos SS7 en busca de errores

Métricas (Prometheus)

La API expone métricas de Prometheus para monitoreo.

Punto Final de Métricas

URL: `http://[server-ip]:8080/metrics`

Formato: Formato de texto de Prometheus

Ejemplo de Salida:

```
# HELP map_requests_total Total de solicitudes MAP
# TYPE map_requests_total counter
map_requests_total{operation="sri"} 42
map_requests_total{operation="sri_for_sm"} 158
map_requests_total{operation="updateLocation"} 23

# HELP cap_requests_total Total de solicitudes CAP
# TYPE cap_requests_total counter
cap_requests_total{operation="initialDP"} 87
cap_requests_total{operation="requestReportBCSMEvent"} 91

# HELP map_request_duration_milliseconds Duración de
solicitudes/respuestas MAP en ms
# TYPE map_request_duration_milliseconds histogram
map_request_duration_milliseconds_bucket{operation="sri",le="10"}
5
map_request_duration_milliseconds_bucket{operation="sri",le="50"}
12
map_request_duration_milliseconds_bucket{operation="sri",le="100"}
35
...

# HELP map_pending_requests Número de solicitudes MAP pendientes
# TYPE map_pending_requests gauge
map_pending_requests 3

# HELP omniss7_license_status Estado actual de la licencia (1 =
válido, 0 = inválido)
# TYPE omniss7_license_status gauge
omniss7_license_status 1
```

Métricas Disponibles

Métrica	Tipo	Etiquetas	Desc
`map_requests_total`	Contador	`operation`	Número solicitud tipo de c
`cap_requests_total`	Contador	`operation`	Número solicitud tipo de c
`map_request_duration_milliseconds`	Histograma	`operation`	Duración solicitud milisegu
`map_pending_requests`	Medidor	-	Número transacc pendient
`ussd_requests_total`	Contador	`direction`	Número solicitud (entrant
`ussd_active_sessions`	Medidor	-	Número USSD ac
`omniss7_license_status`	Medidor	-	Estado a licencia 0 = invá

Configuración de Prometheus

Agregue a su `prometheus.yml`:

```yaml
scrape_configs:
  - job_name: 'omniss7'
    static_configs:
      - targets: ['server-ip:8080']
    metrics_path: '/metrics'
    scrape_interval: 15s
```

Ejemplos de Solicitudes

Ejemplo en Python

```python
import requests
import json

# Solicitud SRI-for-SM
url = "http://localhost:8080/api/sri-for-sm"
payload = {
    "msisdn": "1234567890",
    "service_center": "5551234567"
}

response = requests.post(url, json=payload, timeout=15)

if response.status_code == 200:
    result = response.json()
    print(f"Éxito: {result}")
elif response.status_code == 504:
    print("Tiempo de espera - sin respuesta de la red")
else:
    print(f"Error: {response.status_code} - {response.text}")
```

Ejemplo en JavaScript

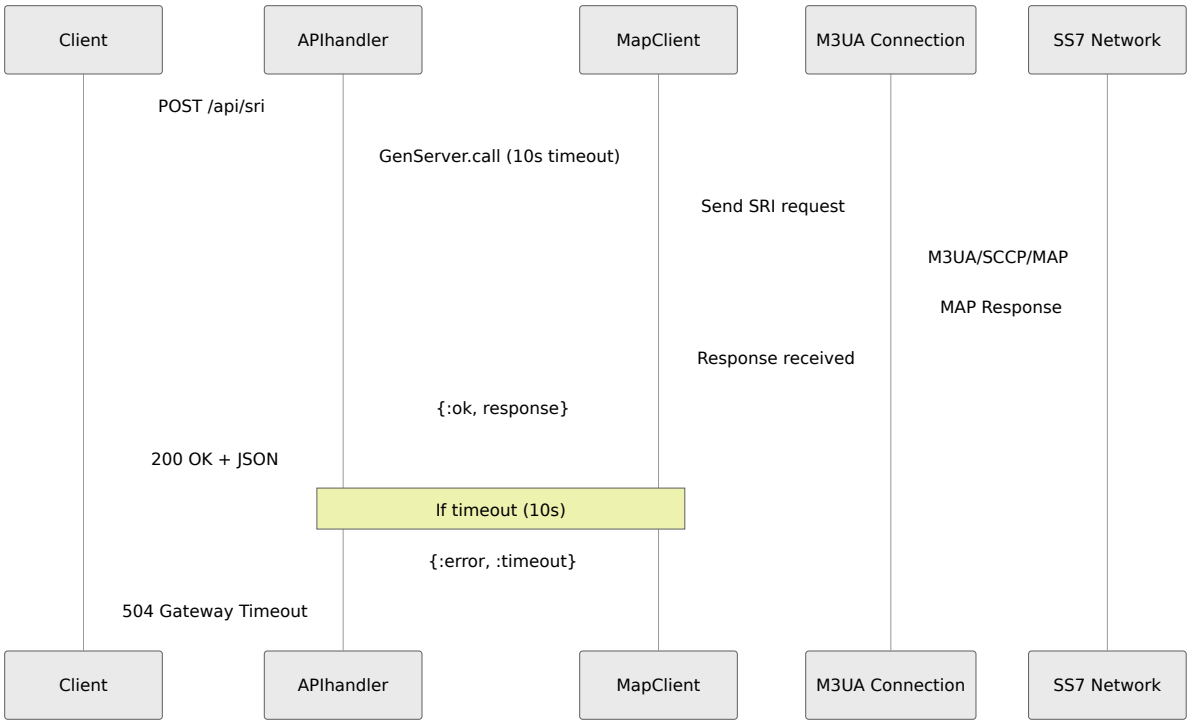
```
const axios = require('axios');

async function sendSRI() {
  try {
    const response = await
axios.post('http://localhost:8080/api/sri', {
      msisdn: '1234567890',
      gmsc: '5551234567'
    }, {
      timeout: 15000
    });

    console.log('Éxito:', response.data);
  } catch (error) {
    if (error.code === 'ECONNABORTED') {
      console.error('Tiempo de espera - sin respuesta de la red');
    } else {
      console.error('Error:', error.response?.data ||
error.message);
    }
  }
}

sendSRI();
```

Ejemplo en Bash/cURL

```
#!/bin/bash

# Solicitud UpdateLocation
response=$(curl -s -w "\n%{http_code}" -X POST
http://localhost:8080/api/updateLocation \
  -H "Content-Type: application/json" \
  -d '{
    "imsi": "001001234567890",
    "vlr": "5551234800"
  }')

http_code=$(echo "$response" | tail -n 1)
body=$(echo "$response" | sed '$d')

if [ "$http_code" -eq 200 ]; then
  echo "Éxito: $body"
elif [ "$http_code" -eq 504 ]; then
  echo "Tiempo de espera - sin respuesta de la red"
else
  echo "Error $http_code: $body"
fi
```

Diagramas de Flujo

Flujo de Solicitudes de API



Resumen

La API REST de OmniSS7 proporciona:

 Operaciones MAP - Soporte completo para SRI, SRI-for-SM, UpdateLocation, entrega de SMS, autenticación
 Swagger UI - Documentación y pruebas interactivas de la API
 Métricas de Prometheus - Monitoreo y observabilidad
 Tiempos de Espera Codificados - Tiempo de espera de 10 segundos para todas las solicitudes MAP
 Servidor HTTP - Se ejecuta en el puerto 8080 (configurable a través de `start_http_server`)

Para acceso a la Interfaz Web, consulte la Guía de la Interfaz Web.

Para detalles de configuración, consulte la Referencia de Configuración.

Referencia Técnica (Apéndice)

← Volver a la Documentación Principal

Referencia técnica para los protocolos SS7 y la implementación de OmniSS7.

Pila de Protocolos SS7

Códigos de Operación MAP

Operación	Opcode	Propósito
updateLocation	2	Registrar ubicación del suscriptor
cancelLocation	3	Darse de baja del VLR
provideRoamingNumber	4	Solicitar MSRN
sendRoutingInfo	22	Consultar enrutamiento de llamadas
mt-forwardSM	44	Entregar SMS al suscriptor
sendRoutingInfoForSM	45	Consultar enrutamiento de SMS
mo-forwardSM	46	Reenviar SMS del suscriptor
sendAuthenticationInfo	56	Solicitar vectores de autenticación

Tipos de Mensajes TCAP

- **BEGIN** - Iniciar transacción
- **CONTINUE** - A mitad de transacción
- **END** - Respuesta final
- **ABORT** - Cancelar transacción

Direccionamiento SCCP

Formatos de Título Global

- **E.164** - Número de teléfono internacional (por ejemplo, 447712345678)
- **E.212** - Formato IMSI (por ejemplo, 234509876543210)
- **E.214** - Formato de código de punto

Números de Subsistema (SSN)

- **SSN 6**: HLR
- **SSN 7**: VLR
- **SSN 8**: MSC/SMSC
- **SSN 9**: GMLC
- **SSN 10**: SGSN

TPDU de SMS

Tipos de Mensajes

- **SMS-DELIVER** (MT) - De la red al móvil
- **SMS-SUBMIT** (MO) - Del móvil a la red

- **SMS-STATUS-REPORT** - Estado de entrega
- **SMS-COMMAND** - Comando remoto

Codificaciones de Caracteres

- **GSM7** - Alfabeto GSM de 7 bits (160 caracteres por SMS)
- **UCS2** - Unicode de 16 bits (70 caracteres por SMS)
- **8-bit** - Datos binarios (140 bytes por SMS)

Estados M3UA

- **DOWN** - Sin conexión SCTP
- **CONNECTING** - Conectando SCTP
- **ASPUP_SENT** - Esperando ACK de ASPUP
- **INACTIVE** - ASP activo pero no en uso
- **ASPAC_SENT** - Esperando ACK de ASPAC
- **ACTIVE** - Listo para tráfico

Códigos de Punto Comunes SS7

Los códigos de punto son típicamente valores de 14 bits (ITU) o 24 bits (ANSI).

Formato de Ejemplo (ITU):

- Red: 3 bits
- Clúster: 8 bits
- Miembro: 3 bits

Códigos de Error SCCP

- **0** - Sin traducción para la dirección

- **1** - Sin traducción para dirección específica
- **2** - Congestión de subsistema
- **3** - Fallo de subsistema
- **4** - Usuario no equipado
- **5** - Fallo de MTP
- **6** - Congestión de red
- **7** - No calificado
- **8** - Error en el transporte de mensajes

Códigos de Error MAP

Código	Error	Descripción
1	unknownSubscriber	Suscriptor no en HLR
27	absentSubscriber	Suscriptor no alcanzable
34	systemFailure	Fallo de red
35	dataMissing	Datos requeridos no disponibles
36	unexpectedDataValue	Valor de parámetro inválido

Documentación Relacionada

- ← Volver a la Documentación Principal
- Guía STP
- Guía del Cliente MAP
- Guía del Centro de SMS
- Guía HLR
- Características Comunes

Guía de Configuración del Gateway CAMEL

Descripción general

El modo **Gateway CAMEL (CAMELGW)** transforma OmniSS7 en una plataforma de Red Inteligente (IN) que proporciona control de llamadas en tiempo real y servicios de facturación utilizando el protocolo CAMEL Application Part (CAP).

¿Qué es CAMEL?

CAMEL (Aplicaciones Personalizadas para Lógica Mejorada de Redes Móviles) es un conjunto de estándares diseñados para funcionar en una red central GSM o en una red UMTS. Permite a los operadores proporcionar servicios que requieren control en tiempo real de las llamadas, tales como:

- **Llamadas prepagadas** - Verificación y facturación de saldo en tiempo real
- **Servicios de tarifa premium** - Facturación especial para servicios de valor añadido
- **Control de enrutamiento de llamadas** - Enrutamiento dinámico de destinos basado en tiempo/ubicación
- **Redes privadas virtuales** - Planes de numeración corporativa
- **Filtrado de llamadas** - Permitir/bloquear llamadas según criterios

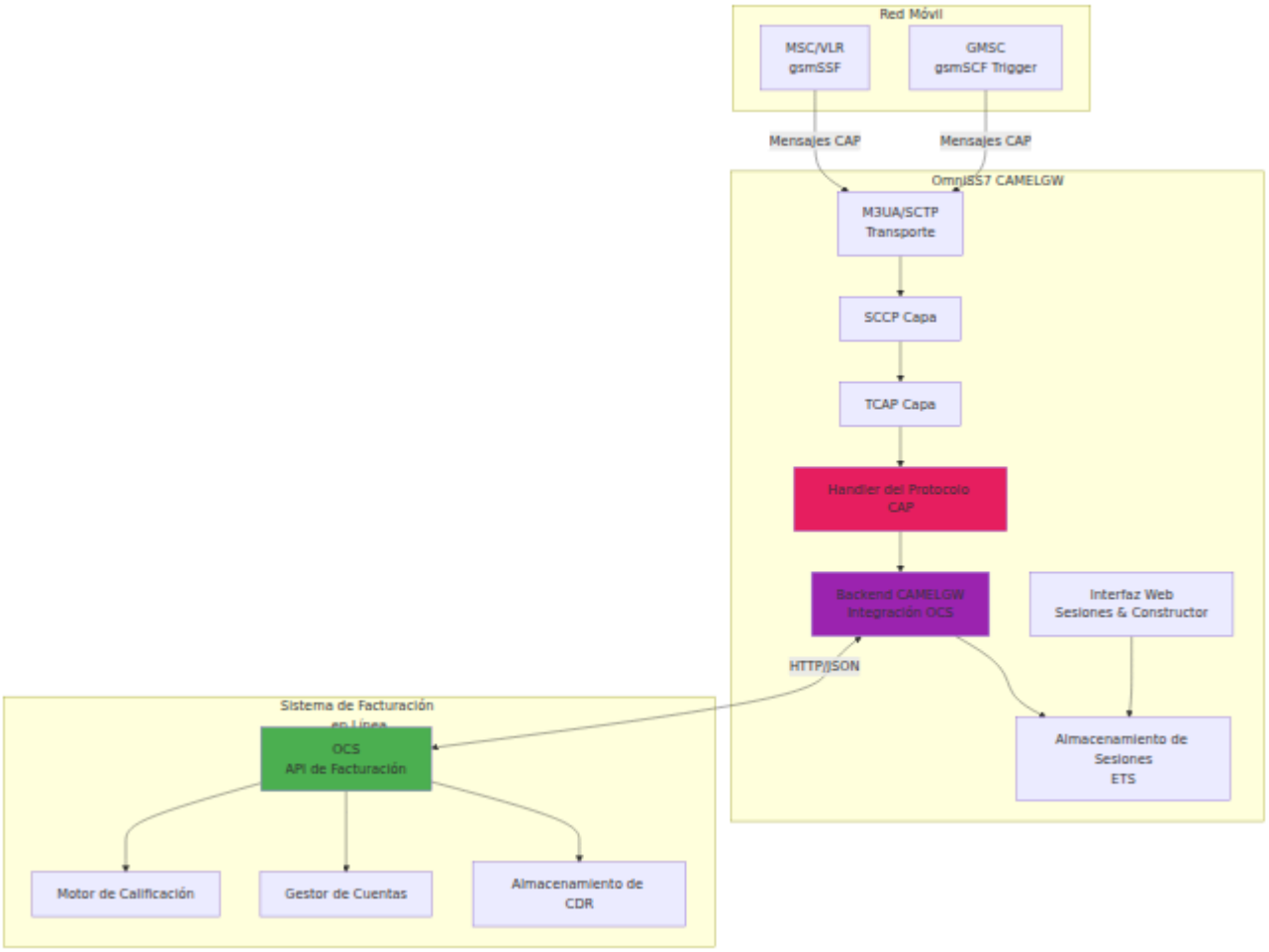
Versiones del Protocolo CAP

OmniSS7 CAMELGW soporta múltiples versiones de CAP:

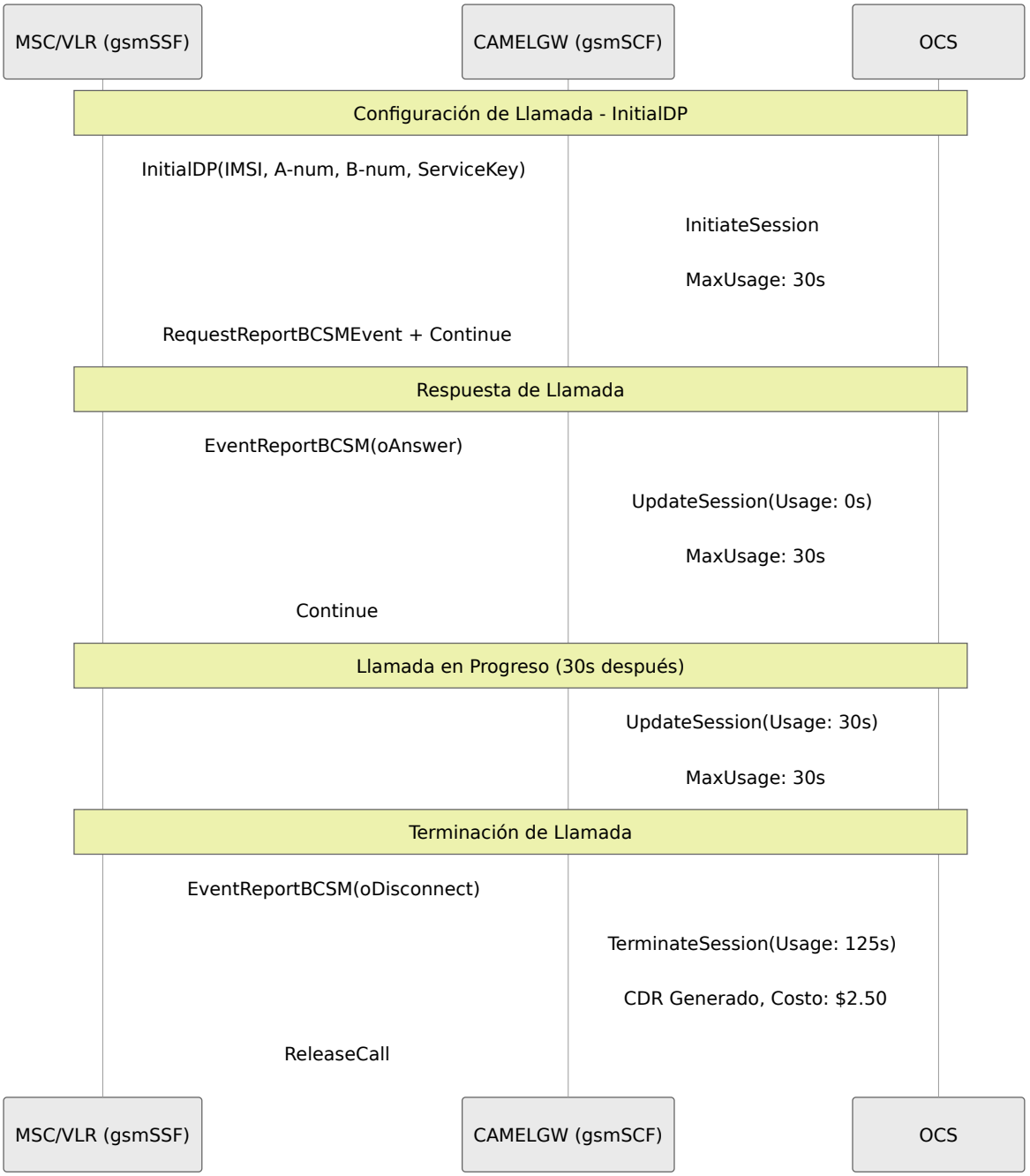
Versión	Fase	Características
CAP v1	Fase CAMEL 1	Control básico de llamadas, operaciones limitadas
CAP v2	Fase CAMEL 2	Operaciones mejoradas, soporte para SMS
CAP v3	Fase CAMEL 3	Soporte para GPRS, operaciones adicionales
CAP v4	Fase CAMEL 4	Características avanzadas, soporte multimedia

Predeterminado: CAP v2 (el más ampliamente desplegado)

Arquitectura



Ejemplo de Flujo de Llamadas



Configuración

Requisitos Previos

- OmniSS7 instalado y en funcionamiento
- Conectividad M3UA a MSC/GMSC (gsmSSF)

- Sistema de Facturación en Línea (OCS) con endpoint API (opcional, para facturación en tiempo real)

Habilitar Modo Gateway CAMEL

Edita `config/runtime.exs` y configura la sección del Gateway CAMEL:

```
config :omniss7,
  # Flags de modo - Habilitar características CAP/CAMEL
  cap_client_enabled: true,
  camelgw_mode_enabled: true,

  # Deshabilitar otros modos
  map_client_enabled: false,
  hlr_mode_enabled: false,
  smsc_mode_enabled: false,

  # Configuración de Versión CAP/CAMEL
  # Determina qué versión de CAP usar para solicitudes salientes y
diálogo
  # Opciones: :v1, :v2, :v3, :v4
  cap_version: :v2,

  # Integración OCS (para facturación en tiempo real)
  ocs_enabled: true,
  ocs_url: "http://your-ocs-server/api/charging",
  ocs_timeout: 5000,  # milisegundos
  ocs_auth_token: "your-api-token"  # Opcional, si OCS requiere
autenticación

  # Configuración de Conexión M3UA para CAMEL
  # Conectar como ASP (Proceso de Servidor de Aplicaciones) para
operaciones CAP
  cap_client_m3ua: %{
    mode: "ASP",
    callback: {CapClient, :handle_payload, []},
    process_name: :camelgw_client_asp,

    # Endpoint local (sistema CAMELGW)
    local_ip: {10, 179, 4, 13},
    local_port: 2905,

    # Endpoint remoto (MSC/GMSC - gsmSSF)
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,

    # Parámetros M3UA
    routing_context: 1,
    network_appearance: 0,
```

```
    asp_identifier: 13
  }
```

Configurar Páginas de Interfaz Web

La Interfaz Web incluye páginas especializadas para operaciones CAMEL:

```
config :control_panel,
  use_additional_pages: [
    {SS7.Web.EventsLive, "/events", "Eventos SS7"},
    {SS7.Web.TestClientLive, "/client", "Cliente SS7"},
    {SS7.Web.M3UAStatusLive, "/m3ua", "M3UA"},
    {SS7.Web.CAMELSessionsLive, "/camel_sessions", "Sesiones
CAP"},
    {SS7.Web.CAMELRequestLive, "/camel_request", "Solicitudes
CAP"}
  ],
  page_order: ["/events", "/client", "/m3ua", "/camel_sessions",
               "/camel_request", "/application", "/configuration"]
```

Operaciones CAP Soportadas

Operaciones Entrantes (de gsmSSF → gsmSCF)

Operación	Opcode	Descripción	H
InitialDP	0	Punto de Detección Inicial - notificación de configuración de llamada	`handle_initial_`
EventReportBCSM	6	Evento del Modelo de Estado de Llamada básico (respuesta, desconexión, etc.)	`handle_event_re`
ApplyChargingReport	71	Informe de facturación de gsmSSF	`handle_apply_ch`
AssistRequestInstructions	16	Solicitud de asistencia de gsmSRF	`handle_assist_r`

Operaciones Salientes (de gsmSCF → gsmSSF)

Operación	Opcode	Descripción	
Connect	20	Conectar llamada al número de destino	`CapRequestGen`
Continue	31	Continuar el procesamiento de la llamada sin modificación	`CapRequestGen`
ReleaseCall	22	Liberar/terminar la llamada	`CapRequestGen`
RequestReportBCSMEvent	23	Solicitar notificación de eventos de llamada	`CapRequestGen`
ApplyCharging	35	Aplicar facturación a la llamada	`CapRequestGen`

Características de la Interfaz Web

Página de Sesiones CAMEL

URL: `http://localhost/camel_sessions`

Monitoreo en tiempo real de sesiones de llamadas CAMEL activas:

Características:

- **Lista de sesiones en vivo** - Se actualiza automáticamente cada 2 segundos
- **Detalles de la sesión** - OTID, ID de llamada, Estado, Duración
- **Versión CAP** - Muestra la versión del protocolo (CAP v1/v2/v3/v4) detectada desde InitialDP
- **Información de llamada** - IMSI, Número A, Número B, Clave de Servicio
- **Seguimiento de estado** - Iniciada, Respondida, Terminada
- **Temporizador de duración** - Muestra la duración de la llamada en tiempo real

Columnas de la tabla:

- ID de llamada, Estado, Versión, IMSI, Número de Llamada, Número Llamado, Clave de Servicio, Duración, Hora de Inicio, OTID

Estados de la Sesión:

- □ **Iniciada** - InitialDP recibido, esperando respuesta
- □ **Respondida** - Llamada respondida, facturación en progreso
- □ **Terminada** - Llamada finalizada, CDR generado

Detección de Versión CAP: El sistema detecta automáticamente la versión del protocolo CAP desde la parte del diálogo de InitialDP y la muestra en la columna de Versión. Esto ayuda a identificar qué versión de CAP está utilizando cada MSC.

Constructor de Solicitudes CAMEL

URL: `http://localhost/camel_request`

Herramienta interactiva para construir y enviar solicitudes CAP:

Características:

- **Selector de tipo de solicitud** - InitialDP, Connect, ReleaseCall, etc.
- **Campos de formulario dinámicos** - Se adapta al tipo de solicitud seleccionado
- **Opciones SCCP/M3UA** - Configuración avanzada de direccionamiento

- **Historial de solicitudes** - Últimas 20 solicitudes con estado
- **Seguimiento de sesión** - Mantiene OTID para solicitudes de seguimiento
- **Retroalimentación en tiempo real** - Mensajes de éxito/error

Tipos de Solicitudes:

1. **InitialDP** - Iniciar nueva sesión de llamada
    - Clave de Servicio (entero)
    - Número de Llamada (A-party)
    - Número de Llamada (B-party)
2. **Connect** - Enrutar llamada al destino
    - Número de Destino
3. **ReleaseCall** - Terminar llamada
    - Código de Causa (16=Normal, 17=Ocupado, 31=No Especificado)
4. **RequestReportBCSMEvent** - Solicitar notificaciones de eventos
    - Eventos: oAnswer, oDisconnect, tAnswer, tDisconnect
5. **Continue** - Continuar llamada sin modificación
    - No se requieren parámetros
6. **ApplyCharging** - Aplicar límites de duración de llamada
    - Duración (segundos, 1-864000)
    - Liberar en Tiempo de Espera (booleano)
    - Ver Guía del Constructor de Solicitudes CAMEL para uso detallado

Opciones SCCP Avanzadas:

- Título Global de la Parte Llamada
- Título Global de la Parte Llamante
- SSN Llamado (predeterminado: 146 = gsmSSF)
- SSN Llamante (predeterminado: 146)

Opciones M3UA:

- OPC (Código de Punto de Origen, predeterminado: 5013)
- DPC (Código de Punto de Destino, predeterminado: 5011)

Integración con OCS

Ciclo de Vida de la Llamada con Facturación

1. Inicio de Llamada (InitialDP)

Cuando MSC envía InitialDP, CAMELGW:

1. **Detecta la versión CAP** - Examina la parte del diálogo para identificar CAP v1/v2/v3/v4
2. **Decodifica el mensaje CAP** - Extrae IMSI, números de llamada
3. **Llama a OCS** - API `InitiateSession`
4. **Recibe autorización** - MaxUsage (por ejemplo, 30 segundos)
5. **Almacena la sesión** - En SessionStore (tabla ETS) con versión CAP
6. **Responde a MSC** - RequestReportBCSMEvent + Continue (usando la misma versión CAP)

Ejemplo:

```
# Datos decodificados de InitialDP
%{
  imsi: "310150123456789",
  calling_party_number: "14155551234",
  called_party_number: "14155556789",
  service_key: 1,
  msc_address: "19216800123",
  cap_version: :v2  # Detectado del diálogo
}

# Respuesta de OCS
{:ok, %{max_usage: 30}}  # 30 segundos autorizados

# Entrada en SessionStore
%{
  call_id: "CAMEL-4B000173",
  initial_dp_data: %{...},
  cap_version: :v2,  # Almacenado para generación de respuestas
  start_time: 1730246400,
  state: :initiated
}
```

2. Respuesta de Llamada (EventReportBCSM - oAnswer)

Cuando la llamada es respondida:

1. **Recibe el evento oAnswer** - Desde MSC
2. **Actualiza OCS** - `UpdateSession` con uso=0
3. **Inicia bucle de débito** - OCS comienza a facturar
4. **Actualiza el estado de la sesión** - `:answered` en SessionStore
5. **Continúa la llamada** - Envía Continue a MSC

3. Actualizaciones Periódicas (Opcional)

Para llamadas largas, solicitar crédito adicional:

```
# Cada 30 segundos
OCS.Client.update_session(call_id, %{}, current_usage)
```

Si MaxUsage devuelve 0, el suscriptor no tiene crédito → Envía ReleaseCall

4. Terminación de Llamada (EventReportBCSM - oDisconnect)

Cuando la llamada finaliza:

1. **Recibe el evento oDisconnect** - Desde MSC
2. **Calcula la duración total** - Desde el tiempo de inicio de la sesión
3. **Termina la sesión OCS** - API `TerminateSession`
4. **CDR generado** - Por OCS con costo final
5. **Limpia la sesión** - Elimina de SessionStore
6. **Envía ReleaseCall** - Confirma la terminación a MSC

Análisis de CDR

Los CDR son generados por tu OCS e incluyen típicamente:

Campos CDR de CAMEL:

- `Account` - IMSI o número de llamada
- `Destination` - Número de la parte llamada
- `OriginID` - Identificador único de llamada (CAMEL-OTID)
- `Usage` - Duración total de la llamada (segundos)
- `Cost` - Costo calculado
- `IMSI` - IMSI del suscriptor
- `CallingPartyNumber` - Parte A
- `CalledPartyNumber` - Parte B
- `MSCAddress` - Código de punto del MSC que sirve
- `ServiceKey` - Clave de servicio CAMEL

Pruebas

Pruebas Manuales con Constructor de Solicitudes

1. **Navegar al Constructor de Solicitudes:**

   ```
   http://localhost/camel_request
   ```

2. **Enviar InitialDP:**

   - Seleccionar "InitialDP" del menú desplegable
   - Clave de Servicio: `100`
   - Número de Llamada: `14155551234`
   - Número Llamado: `14155556789`
   - Hacer clic en "Enviar Solicitud InitialDP"
   - Anotar el OTID generado

3. **Monitorear Sesión:**

   - Abrir nueva pestaña: `http://localhost/camel_sessions`
   - Ver sesión activa con estado "Iniciada"

4. **Simular Respuesta de Llamada:**

   - Volver al Constructor de Solicitudes
   - Seleccionar "EventReportBCSM"
   - Tipo de Evento: `oAnswer`
   - Hacer clic en "Enviar Solicitud EventReportBCSM"
   - El estado de la sesión cambia a "Respondida"

5. **Finalizar Llamada:**

   - Seleccionar "ReleaseCall"
   - Código de Causa: `16` (Normal)
   - Hacer clic en "Enviar Solicitud ReleaseCall"

- El estado de la sesión cambia a "Terminada"

Pruebas con MSC Real

Configurar Servicio CAMEL en MSC

En tu MSC/VLR, configura el servicio CAMEL:

```
# Ejemplo de configuración de MSC Huawei
ADD CAMELSERVICE:
  SERVICEID=1,
  SERVICEKEY=100,
  GSMSCFADDR="55512341234",  # Título Global CAMELGW
  DEFAULTCALLHANDLING=CONTINUE;

ADD CAMELSUBSCRIBER:
  IMSI="310150123456789",
  SERVICEID=1,
  TRIGGERTYPE=TERMCALL;
```

Monitorear Registros

Observa los registros de CAMELGW para mensajes CAP entrantes:

```
# Ver registros en tiempo real
tail -f /var/log/omniss7/omniss7.log

# Filtrar eventos CAP
grep "CAP:" /var/log/omniss7/omniss7.log

# Ver registro de eventos (formato JSON)
curl http://localhost/api/events | jq '.[] | select(.map_event |
startswith("CAP:"))'
```

Pruebas de Carga

Usa el Constructor de Solicitudes en un bucle para pruebas de carga:

```
# Enviar 100 solicitudes InitialDP
for i in {1..100}; do
  curl -X POST http://localhost/api/camel/initial_dp \
    -H "Content-Type: application/json" \
    -d '{
      "service_key": 100,
      "calling_number": "1415555'$i'",
      "called_number": "14155556789"
    }'
  sleep 0.1
done
```

Monitoreo y Operaciones

Métricas de Prometheus

CAMELGW expone métricas en `http://localhost:8080/metrics`:

Métricas específicas de CAP:

- `cap_requests_total{operation}` - Total de solicitudes CAP por tipo de operación (por ejemplo, initialDP, requestReportBCSMEvent)

Métricas adicionales de MAP/API:

- `map_requests_total{operation}` - Total de solicitudes MAP por tipo de operación
- `map_request_duration_milliseconds{operation}` - Histograma de duración de solicitudes
- `map_pending_requests` - Número de transacciones MAP pendientes

Métricas STP de M3UA (si el modo STP está habilitado):

- `m3ua_stp_messages_received_total{peer_name,point_code}` - Mensajes recibidos de pares

- `m3ua_stp_messages_sent_total{peer_name,point_code}` - Mensajes enviados a pares
- `m3ua_stp_routing_failures_total{reason}` - Fallos de enrutamiento por razón

Ejemplos de consultas:

```
# Solicitudes CAP
curl http://localhost:8080/metrics | grep cap_requests_total

# Total de InitialDP recibidos
curl http://localhost:8080/metrics | grep
'cap_requests_total{operation="initialDP"}'

# Solicitudes MAP pendientes
curl http://localhost:8080/metrics | grep map_pending_requests
```

Verificaciones de Salud

```
# Verificar conectividad M3UA
curl http://localhost/api/m3ua-status

# Verificar conectividad OCS
curl http://localhost/api/ocs-status

# Verificar sesiones activas
curl http://localhost/api/camel/sessions/count
```

Configuración de Registros

Ajusta el nivel de registro en `config/runtime.exs`:

```
config :logger,
  level: :info  # Opciones: :debug, :info, :warning, :error

# Habilitar registro de depuración CAP
config :logger, :console,
  metadata: [:cap_operation, :otid, :call_id]
```

Solución de Problemas

Problema: No se reciben mensajes CAP

Síntomas: El Constructor de Solicitudes funciona, pero MSC no envía InitialDP

Verificar:

1. Estado del enlace M3UA: `curl http://localhost/api/m3ua-status`
2. Configuración del servicio CAMEL en MSC (Clave de Servicio, dirección gsmSCF)
3. Enrutamiento SCCP (el Título Global debe enrutar a CAMELGW)
4. Reglas de firewall (permitir el puerto SCTP 2905)

Solución:

```
# Verificar conectividad M3UA
tcpdump -i eth0 sctp

# Comprobar si MSC puede alcanzar CAMELGW
ss -tuln | grep 2905
```

Problema: Errores en OCS

Síntomas: Errores de `INSUFFICIENT_CREDIT` o de tiempo de espera

Verificar:

1. OCS es accesible: `curl http://your-ocs-server/api/health`
2. La cuenta tiene saldo en OCS
3. Plan de calificación configurado en OCS
4. Conectividad de red con OCS
5. El token de autenticación es válido (si es necesario)

Solución:

- Verificar la configuración de la URL de OCS en `runtime.exs`
- Comprobar los registros de OCS en busca de errores
- Probar la API de OCS manualmente con curl
- Verificar que las reglas del firewall permitan la conectividad

Problema: Sesión no encontrada

Síntomas: EventReportBCSM falla con "Sesión no encontrada"

Causa: Desajuste de OTID o sesión expirada

Solución:

1. Verificar OTID en los registros
2. Comprobar el tiempo de espera de la sesión (predeterminado: sin expiración)
3. Asegurarse de que DTID coincida con OTID en los mensajes Continue/End

```
# Comprobar sesiones activas
iex> CAMELGW.SessionStore.list_sessions()
```

Problema: Errores de decodificación

Síntomas: `Failed to decode InitialDP` en los registros

Causa: Desajuste de versión CAP o mensaje mal formado

Solución:

1. Verificar que la configuración de la versión CAP coincida con MSC
2. Verificar que la codificación ASN.1 sea correcta
3. Capturar PCAP y analizar con Wireshark

```
# Capturar mensajes CAP
tcpdump -i eth0 -w cap_trace.pcap sctp port 2905

# Analizar con Wireshark (filtro: m3ua)
wireshark cap_trace.pcap
```

Configuración Avanzada

Múltiples Versiones CAP

Soporte para diferentes versiones CAP por clave de servicio:

```
config :omniss7,
  cap_version_map: %{
    100 => :v2,  # La Clave de Servicio 100 utiliza CAP v2
    200 => :v3,  # La Clave de Servicio 200 utiliza CAP v3
    300 => :v4   # La Clave de Servicio 300 utiliza CAP v4
  },
  cap_version: :v2  # Predeterminado
```

Resumen

El modo Gateway CAMEL permite que OmniSS7 funcione como una plataforma completa de Red Inteligente con:

 Soporte completo del protocolo CAP (v1/v2/v3/v4)
 Facturación en tiempo real a través de la integración OCS
 Operaciones de control de llamadas (Connect, Release, Continue)
 Gestión de sesiones con almacenamiento ETS

 Pruebas interactivas a través del Constructor de Solicitudes de la Interfaz Web
 Monitoreo en vivo de sesiones de llamadas activas
 Generación de CDR para facturación y análisis
 Rendimiento y fiabilidad listos para producción

Para información adicional:

- Documentación del Constructor de Solicitudes CAMEL
- Referencia Técnica - Operaciones CAP

Producto: Gateway CAMEL OmniSS7
Versión de Documentación: 1.0
Última Actualización: 2025-10-26

CAMEL Request Builder - Resumen de Implementación

Descripción General

Se ha creado un nuevo componente LiveView para construir y enviar solicitudes CAMEL/CAP con fines de prueba. Esto proporciona una interfaz de usuario interactiva para crear InitialDP y otras operaciones CAMEL.

Nuevos Componentes

1. Constructor de Solicitudes CAMEL LiveView

Características:

- Interfaz de usuario basada en formularios interactiva para construir solicitudes CAMEL
- Soporte para múltiples tipos de solicitudes:
  - **InitialDP** - Punto de Detección Inicial (notificación de configuración de llamada)
  - **Connect** - Conectar llamada al destino
  - **ReleaseCall** - Liberar/terminar llamada
  - **RequestReportBCSMEvent** - Solicitar notificaciones de eventos
  - **Continue** - Continuar el procesamiento de la llamada
  - **ApplyCharging** - Aplicar límites de carga/duración a las llamadas

Capacidades Clave:

- Desplegable de selección de tipo de solicitud
- Campos de formulario dinámicos según el tipo de solicitud seleccionado
- Opciones avanzadas de SCCP/M3UA (sección colapsable)
  - Títulos Globales de Parte Llamada/Llamante
  - Configuración de SSN (Número de Subsistema)
  - Configuración de OPC/DPC (Código de Punto)
- Historial de solicitudes en tiempo real (últimas 20 solicitudes)
- Seguimiento de sesión a través de OTID
- Retroalimentación de éxito/error
- Seguimiento del tamaño de la solicitud

Ruta: `/camel_request`

2. EventLog Mejorado con Soporte CAMEL

Nuevas Funciones:

- `paklog_camel/2` - Registro dedicado de mensajes CAMEL/CAP
- `lookup_cap_opcode_name/1` - Búsqueda de código de operación CAP
- `find_cap_opcode/1` - Extraer opcode CAP de JSON
- `extract_cap_tids/1` - Extraer OTID/DTID de mensajes CAP
- `format_cap_to_json/1` - Convertir PDUs CAP a formato JSON

Códigos de Operación CAP Soportados:

```
0  => "initialDP"
5  => "connect"
6  => "releaseCall"
7  => "requestReportBCSMEvent"
8  => "eventReportBCSM"
10 => "continue"
13 => "furnishChargingInformation"
35 => "applyCharging"
... (47 operaciones en total)
```

Características:

- Registro en JSON de todas las solicitudes/respuestas CAMEL
- Detección automática de acciones TCAP (Iniciar/Continuar/Finalizar/Abortar)
- Extracción de direccionamiento SCCP
- Manejo de errores para mensajes malformados
- Procesamiento de tareas en segundo plano (no bloqueante)
- Evento prefijado con "CAP:" para fácil filtrado

3. CapClient Actualizado

Cambios:

- Se añadieron llamadas `paklog_camel/2` para mensajes entrantes y salientes
- Registro dual: Tanto MAP (`paklog`) como CAP (`paklog_camel`) para compatibilidad
- Mensajes salientes registrados en `sccp_m3ua_maker/2`
- Mensajes entrantes registrados en `handle_payload/1`

Configuración

Las nuevas páginas LiveView se han añadido a la configuración de tiempo de ejecución:

```
# Archivo: config/runtime.exs

config :control_panel,
  use_additional_pages: [
    {SS7.Web.EventsLive, "/events", "Eventos SS7"},
    {SS7.Web.TestClientLive, "/client", "Cliente SS7"},
    {SS7.Web.M3UAStatusLive, "/m3ua", "M3UA"},
    {SS7.Web.HlrLinksLive, "/hlr_links", "Enlaces HLR"},
    {SS7.Web.CAMELSessionsLive, "/camel_sessions", "Sesiones
CAMEL"},
    {SS7.Web.CAMELRequestLive, "/camel_request", "Constructor de
Solicitudes CAMEL"}
  ],
  page_order: ["/events", "/client", "/m3ua", "/hlr_links",
               "/camel_sessions", "/camel_request",
               "/application", "/configuration"]
```

Uso

Accediendo al Constructor de Solicitudes

1. Navegar a: `https://your-server:8087/camel_request`
2. Seleccionar tipo de solicitud del desplegable
3. Completar los parámetros requeridos
4. Opcionalmente expandir "Opciones Avanzadas de SCCP/M3UA" para ajustes finos
5. Hacer clic en "Enviar Solicitud [RequestType]"

Flujo de Solicitud

InitialDP (Nueva Llamada)

1. Establecer Clave de Servicio (por ejemplo, 100)
2. Establecer Número Llamante (Parte A)
3. Establecer Número Llamado (Parte B)
4. Enviar solicitud → Genera nuevo OTID
5. OTID almacenado en la sesión para solicitudes de seguimiento

Solicitudes de Seguimiento (Connect, ReleaseCall, etc.)

1. Debe tener un OTID activo de InitialDP
2. La solicitud utiliza automáticamente el OTID almacenado
3. Advertencia mostrada si no hay OTID activo

Parámetros de Solicitud

InitialDP:

- Clave de Servicio (entero)
- Número Llamante (formato ISDN)
- Número Llamado (formato ISDN)

Connect:

- Número de Destino (donde enrutar la llamada)

ReleaseCall:

- Código de Causa (16 = Normal, 17 = Ocupado, 31 = No Especificado)

RequestReportBCSMEvent:

- Eventos BCSM (separados por comas: oAnswer, oDisconnect, etc.)

Continue:

- Sin parámetros (utiliza OTID activo)

ApplyCharging:

- Duración (segundos, 1-864000) - Duración máxima de la llamada antes de la acción
- Liberar en Tiempo de Espera (booleano) - Si se debe liberar la llamada cuando expira la duración

Opciones Avanzadas

Direccionamiento SCCP:

- GT de Parte Llamada (Título Global)
- GT de Parte Llamante
- SSN Llamado (por defecto 146 = gsmSSF)
- SSN Llamante (por defecto 146)

Códigos de Punto M3UA:

- OPC (Código de Punto de Origen, por defecto 5013)
- DPC (Código de Punto de Destino, por defecto 5011)

Registro en JSON

Todos los mensajes CAMEL ahora se registran en formato JSON en el registro de eventos con:

- **Dirección:** entrante/saliente
- **Acción TCAP:** Iniciar/Continuar/Finalizar/Abortar
- **Operación CAP:** por ejemplo, "CAP:initialDP", "CAP:connect"
- **Direccionamiento SCCP:** Información de Parte Llamada/Llamante
- **TIDs:** OTID/DTID para correlación
- **Mensaje Completo:** PDU CAP codificado en JSON

Ejemplo de Entrada de Registro


```
{
  "map_event": "CAP:initialDP",
  "direction": "outgoing",
  "tcap_action": "Begin",
  "otid": "A1B2C3D4",
  "sccp_called": {
    "SSN": 146,
    "GlobalTitle": {
      "Digits": "55512341234",
      "NumberingPlan": "isdn_tele",
      "NatureOfAddress_Indicator": "international"
    }
  },
  "event_message": "{ ... full CAP PDU ... }"
}
```


Historial de Solicitudes

La interfaz de usuario muestra las últimas 20 solicitudes con:

- Marca de tiempo
- Tipo de solicitud (con insignia codificada por colores)
- OTID (primeros 8 caracteres hexadecimales)
- Estado (enviado/error)
- Tamaño del mensaje en bytes

Seguimiento de Sesiones

Panel de Información de Sesión Actual:

- Muestra OTID activo
- Muestra el tamaño del último byte de solicitud
- Visible solo cuando la sesión está activa

Flujo de Trabajo de Pruebas

1. **Iniciar Nueva Llamada:**
    - Enviar InitialDP → Obtener OTID
    - El sistema crea sesión
2. **Controlar Llamada:**
    - Enviar RequestReportBCSMEvent → Solicitar notificaciones
    - Enviar ApplyCharging → Establecer límite de duración de la llamada (por ejemplo, 290 segundos)
    - Enviar Connect → Enrutar al destino
    - O ENVIAR ReleaseCall → Terminar
3. **Ver Resultados:**
    - Revisar historial de solicitudes
    - Monitorear página de Sesiones CAMEL
    - Revisar registros de eventos con prefijo "CAP:"

ApplyCharging - Control de Duración de Llamada

Descripción General

La operación ApplyCharging permite establecer una duración máxima de llamada y, opcionalmente, liberar la llamada cuando esa duración expira. Esto se utiliza típicamente para escenarios de carga prepaga o para hacer cumplir límites de tiempo en las llamadas.

Casos de Uso

- **Carga Prepaga**: Limitar la duración de la llamada según el saldo del suscriptor

- **Facturación Basada en Tiempo**: Hacer cumplir intervalos de carga periódicos
- **Gestión de Recursos**: Prevenir que las llamadas se ejecuten indefinidamente
- **Integración OCS**: Coordinar con Sistemas de Carga en Línea para control de crédito en tiempo real

Parámetros

Duración (maxCallPeriodDuration)

- **Tipo**: Entero (1-864000 segundos)
- **Descripción**: Número máximo de segundos que la llamada puede ejecutarse antes de que expire el temporizador
- **Ejemplos**:
  - `60` = 1 minuto
  - `290` = 4 minutos 50 segundos (valor de prueba común)
  - `3600` = 1 hora
  - `86400` = 24 horas

Liberar en Tiempo de Espera (releaseIfDurationExceeded)

- **Tipo**: Booleano (true/false)
- **Predeterminado**: true
- **Descripción**: Lo que sucede cuando expira la duración:
  - `true`: Liberar/desconectar automáticamente la llamada
  - `false`: Enviar notificación pero mantener la llamada activa (permite que gsmSCF tome acción)

Estructura del Mensaje

El mensaje ApplyCharging se codifica como un TCAP Continue con:

- **TCAP**: Mensaje de Continuar (utiliza la transacción existente)
- **Opcode**: 35 (applyCharging)
- **Parámetros**: ApplyChargingArg que contiene:

- `aChBillingChargingCharacteristics`: Información de carga basada en tiempo
  - `timeDurationCharging`: Duración máxima y bandera de liberación
- `partyToCharge`: Qué parte es cargada (predeterminado: sendingSideID)

Ejemplo de Uso

Escenario: Llamada prepaga con límite de 5 minutos

1. Enviar **InitialDP** para iniciar el monitoreo de la llamada

```
Clave de Servicio: 100
Llamante: 447700900123
Llamado: 447700900456
→ OTID: A1B2C3D4
```

2. Enviar **ApplyCharging** para establecer límite de 5 minutos

```
Duración: 300 (segundos)
Liberar en Tiempo de Espera: true
→ Utiliza OTID: A1B2C3D4
```

3. Enviar **Connect** para completar la llamada

```
Destino: 447700900456
→ Utiliza OTID: A1B2C3D4
```

4. Después de 5 minutos (300 segundos):
   - Llamada liberada automáticamente por la red
   - gsmSCF recibe notificación de desconexión

Mejores Prácticas

1. **Siempre enviar ApplyCharging ANTES de Connect**

    - Asegura que la carga esté activa cuando se conecta la llamada
    - Previene segmentos de llamada sin carga

2. **Usar con RequestReportBCSMEvent**
    - Solicitar eventos `oAnswer` y `oDisconnect`
    - Permite el seguimiento de la duración real de la llamada
    - Permite la reaplicación de carga si es necesario

3. **Establecer duraciones razonables**
    - Demasiado corto: Operaciones de carga frecuentes, mala experiencia del usuario
    - Demasiado largo: Riesgo de pérdida de ingresos en llamadas prepago
    - Típico: 60-300 segundos para prepago, más largo para postpago

4. **Manejar el tiempo de espera de manera adecuada**
    - Si `release=false`, estar preparado para manejar notificaciones de expiración del temporizador
    - Implementar lógica para extender la duración o liberar la llamada

Manejo de Errores

Problemas comunes:

- **No hay OTID activo**: Debe enviar InitialDP primero
- **Duración inválida**: Debe ser de 1-864000 segundos
- **Soporte de red**: Algunas implementaciones de SSF pueden no soportar ApplyCharging
- **Precisión del temporizador**: La resolución del temporizador de la red es típicamente de 1 segundo, pero puede variar

Monitoreo

Rastrear operaciones ApplyCharging a través de:

- **Historial de Solicitudes**: Muestra solicitudes ApplyCharging enviadas

- **Registro de Eventos**: Buscar "CAP:applyCharging"
- **Sesiones CAMEL**: Monitorear sesiones activas con carga aplicada
- **Rastreo TCAP**: Depurar problemas de codificación/decodificación

Detalles de Implementación

Gestión de Estado

- LiveView asigna el estado del formulario de seguimiento
- OTID almacenado en los sockets asigna
- Historial de solicitudes limitado a 20 entradas
- Auto-refresco desactivado (solo envío manual)

Generación de Solicitudes

- Utiliza el módulo existente `CapRequestGenerator`
- Construye las estructuras TCAP/CAP adecuadas
- Codifica con el códec `:TCAPMessages`
- Envuelve en SCCP a través de `CapClient.sccp_m3ua_maker/2`

Mecanismo de Envío

- Envía a través de M3UA a `:camelgw_client_asp`
- Utiliza el contexto de enrutamiento 1
- Encapsulación automática SCCP/M3UA

Manejo de Errores

- Validación de formularios con retroalimentación al usuario
- Manejo adecuado de OTID faltante
- Errores de análisis mostrados en la interfaz de usuario
- Fallos de codificación registrados

Mejoras Futuras

Adiciones potenciales:

1. Plantillas/presets de solicitud
2. Correlación y visualización de respuestas
3. Visualización del flujo de llamadas
4. Detalle de sesión en profundidad
5. Exportar historial de solicitudes
6. Pruebas de carga (solicitudes masivas)
7. Exportación PCAP de mensajes generados
8. Validación de parámetros CAP

Notas de Integración

- Compatible con el registro MAP existente (`paklog`)
- Comparte la base de datos de registro de eventos con eventos MAP
- Utiliza la misma infraestructura SCCP/M3UA
- Funciona con CAMELSessionsLive para monitoreo
- Se integra con el enrutamiento M3UA existente

Archivos Modificados

- `config/runtime.exs` - ACTUALIZADO

Dependencias

- Generador de Solicitudes CapRequestGenerator existente
- CapClient para envío M3UA
- M3UA.Server para transmisión de paquetes
- EventLog para registro de mensajes
- Marco Phoenix LiveView

- Panel de Control para infraestructura de UI

Guía de Características Comunes

← Volver a la Documentación Principal

Esta guía cubre características comunes a todos los modos de operación de OmniSS7.

Tabla de Contenidos

Descripción General de la Interfaz Web

La Interfaz Web es accesible a través de la dirección de su servidor web configurado.

Navegación Principal

- **Eventos** - Eventos de señalización SS7 en tiempo real y registros de mensajes
- **Aplicación** - Estado de la aplicación e información de tiempo de ejecución
- **Configuración** - Visor de configuración del sistema
- **Estado M3UA** - Conexiones de pares M3UA (modo STP)
- **Cola de SMS** - Mensajes SMS salientes (modo SMSc)

Accediendo a la Interfaz Web

1. Abra su navegador web
2. Navegue a la dirección del hostname configurado (por ejemplo, `http://localhost`)
3. Vea el panel de estado del sistema

Documentación de la API Swagger

Documentación interactiva de la API:

```
http://your-server/swagger
```

Configuración de la Interfaz Web

Configure en `config/runtime.exs`:

```
config :control_panel,
  # Orden de las páginas en el menú de navegación
  page_order: ["/events", "/application", "/configuration"],

  # Configuración del servidor web
  web: %{
    listen_ip: "0.0.0.0",    # IP para enlazar (0.0.0.0 para todas
las interfaces)
    port: 80,                 # Puerto HTTP (443 para HTTPS)
    hostname: "localhost",    # Nombre del servidor para la
generación de URL
    enable_tls: false,        # Establecer verdadero para
habilitar HTTPS
    tls_cert: "cert.pem",    # Ruta al archivo de certificado TLS
    tls_key: "key.pem"       # Ruta al archivo de clave privada
TLS
  }
```

Parámetros de Configuración:

Parámetro	Tipo	Predeterminado	Descripción
`page_order`	Lista	`["/events", "/application", "/configuration"]`	Orden de las páginas en el menú de navegación
`listen_ip`	Cadena	`"0.0.0.0"`	Dirección IP para enlazar el servidor web
`port`	Entero	`80`	Puerto HTTP (usar 443 para HTTPS)
`hostname`	Cadena	`"localhost"`	Nombre del servidor para la generación de URL
`enable_tls`	Booleano	`false`	Habilitar HTTPS con TLS
`tls_cert`	Cadena	`"cert.pem"`	Ruta al certificado TLS (cuando TLS está habilitado)
`tls_key`	Cadena	`"key.pem"`	Ruta a la clave privada TLS (cuando TLS está habilitado)

Configuración del Registrador

Configure el nivel de registro en `config/runtime.exs`:

```
config :logger,
  level: :debug  # Opciones: :debug, :info, :warning, :error
```

Niveles de Registro:

- `:debug` - Información detallada de depuración
- `:info` - Mensajes informativos generales
- `:warning` - Mensajes de advertencia sobre problemas potenciales
- `:error` - Mensajes de error solamente

Documentación de la API

URL Base de la API

```
http://your-server/api
```

Códigos de Respuesta

- **200** - Éxito
- **400** - Solicitud Incorrecta
- **504** - Tiempo de Espera de la Puerta de Enlace

Especificación OpenAPI

```
http://your-server/swagger.json
```

Monitoreo y Métricas

Endpoint de Métricas de Prometheus

```
http://your-server/metrics
```

Categorías Clave de Métricas

Métricas M3UA/SCTP:

- Cambios en el estado de la asociación SCTP
- Transiciones de estado ASP M3UA
- Unidades de datos de protocolo enviadas/recibidas

Métricas M2PA:

- Transiciones de estado de enlace (DOWN → ALIGNMENT → PROVING → READY)
- Mensajes y bytes enviados/recibidos por enlace
- Errores específicos de enlace (decodificar, codificar, SCTP)

Métricas STP:

- Mensajes recibidos/enviados por par
- Fallos de enrutamiento por razón
- Distribución del tráfico entre pares

Métricas del Cliente MAP:

- Solicitudes MAP por tipo de operación
- Histogramas de duración de solicitudes
- Medidor de transacciones pendientes

Métricas CAP:

- Solicitudes CAP por tipo de operación
- Operaciones de puerta de enlace CAMEL

Métricas SMSc:

- Profundidad de la cola
- Tasas de entrega
- Mensajes fallidos

Integración con Grafana

Las métricas de OmniSS7 son compatibles con Prometheus y Grafana.

Mejores Prácticas

Recomendaciones de Seguridad

1. **Aislamiento de Red**
    - Desplegar en VLAN dedicada
    - Reglas de firewall para restringir el acceso
    - Permitir SCTP solo desde direcciones conocidas
2. **Seguridad de la Interfaz Web**
    - Habilitar TLS para producción
    - Usar proxy inverso con autenticación
    - Restringir a IPs de gestión
3. **Seguridad de la API**
    - Implementar limitación de tasa
    - Usar claves de API o OAuth
    - Registrar todas las solicitudes para auditoría

Ajuste de Rendimiento

1. **Límites de TPS**
    - Configurar TPS adecuado
    - Monitorear carga del sistema
    - Ajustar buffers SCTP
2. **Optimización de Base de Datos**

- Agregar índices
- Archivar mensajes antiguos
- Monitorear el grupo de conexiones

3. **Ajuste de M3UA**
    - Ajustar intervalos de latido SCTP
    - Configurar valores de tiempo de espera
    - Usar múltiples enlaces para redundancia

Multihoming SCTP para Redundancia de Red

¿Qué es el Multihoming SCTP?

El Multihoming SCTP es una característica integrada del protocolo SCTP que permite que una única conexión M3UA se vincule a múltiples direcciones IP en la misma interfaz de red o a través de diferentes interfaces de red. Esto proporciona conmutación por error automática y redundancia en la capa de transporte.

Beneficios Clave:

- **Conmutación por Error Automática**: Si un camino de red falla, SCTP cambia automáticamente a un camino alternativo sin interrumpir la conexión
- **Conmutación por Error sin Configuración**: No se necesita lógica a nivel de aplicación - SCTP maneja la supervisión de caminos y la conmutación por error
- **Mayor Fiabilidad**: Sobrevive a fallos de red, fallos de conmutadores o fallos de NIC
- **Balanceo de Carga**: SCTP puede distribuir el tráfico a través de múltiples caminos (dependiente de la implementación)

Cómo Funciona

Cuando configura múltiples direcciones IP para una conexión M3UA, SCTP:

1. **Se vincula a todas las IPs**: El socket se vincula a todas las direcciones IP configuradas simultáneamente
2. **Supervisa los caminos**: SCTP envía continuamente paquetes de latido en todos los caminos para monitorear su salud
3. **Detecta fallos**: Si los latidos fallan en el camino principal, SCTP lo marca como inalcanzable
4. **Conmutación por error automática**: El tráfico cambia inmediatamente a un camino de respaldo sin intervención de la aplicación
5. **Recuperación de caminos**: Cuando el camino fallido se recupera, SCTP lo detecta y lo marca como disponible nuevamente

Configuración

El multihoming SCTP se configura proporcionando una **lista de direcciones IP** en lugar de una única tupla IP.

Una IP (Tradicional)

```
# Una IP - sin multihoming
local_ip: {10, 179, 4, 10}
```

Múltiples IPs (Multihoming Habilitado)

```
# Múltiples IPs - multihoming habilitado
# La primera IP es primaria, las IPs subsiguientes son caminos de
respaldo
local_ip: [{10, 179, 4, 10}, {10, 179, 4, 11}]
```

Ejemplos de Configuración

> **Nota Importante para el Rol del Servidor (Conexiones Entrantes):**

> Al configurar pares con `role: :server` (aceptando conexiones entrantes de pares multihomed), debe especificar `remote_ip` como una **única tupla** - la dirección IP que el par remoto utiliza para iniciar la conexión SCTP (SCTP INIT). NO use una lista.
>
> **¿Por qué?** El STP coincide las conexiones entrantes según la IP de origen del paquete SCTP INIT. SCTP descubrirá automáticamente las otras direcciones IP multihomed del par durante el apretón de manos de asociación. El formato de lista para `remote_ip` solo es válido para `role: :client` (conexiones salientes).
>
> **Ejemplos:**

```
# ✓ CORRECTO - Rol de servidor con single remote_ip
%&#123;
  role: :server,
  remote_ip: &#123;10, 0, 2, 100&#125;,  # Tupla única
  # ...
&#125;

# ✗ INCORRECTO - Rol de servidor con lista NO coincidirá
conexiones entrantes
%&#123;
  role: :server,
  remote_ip: [&#123;10, 0, 2, 100&#125;, &#123;10, 0, 2,
101&#125;],  # La lista no funcionará
  # ...
&#125;
```

Ejemplo 1: Par STP con Multihoming (Rol de Cliente - Saliente)

```
# Configuración de par en modo STP (conexión SALIENTE)
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 1,
      name: "Partner_STP_Redundant",
      role: :client,  # Saliente - nosotros iniciamos la conexión
      # Multihoming: vincular a dos IPs locales para redundancia
      local_ip: [{213, 57, 23, 200}, {213, 57, 23, 201}],
      local_port: 0,
      # El par remoto también soporta multihoming - la lista está
bien para el rol de cliente
      remote_ip: [{213, 57, 23, 100}, {213, 57, 23, 101}],
      remote_port: 2905,
      routing_context: 1,
      point_code: 100,
      network_indicator: :international
    }
  ]
```

Ejemplo 2: Cliente MAP con Multihoming

```
# Modo cliente MAP con multihoming
config :omniss7,
  map_client_enabled: true,
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    # Multihoming: dos IPs locales para conmutación por error
    local_ip: [{10, 0, 0, 100}, {10, 0, 0, 101}],
    local_port: 2905,
    # STP remoto con soporte de multihoming
    remote_ip: [{10, 0, 0, 1}, {10, 0, 0, 2}],
    remote_port: 2905,
    routing_context: 1
  }
```

Ejemplo 3: Escuchador STP con Multihoming

```
# Servidor STP independiente con multihoming
config :omniss7,
  sctp_handler: %{
    enabled: true,
    # Escuchar en múltiples IPs para conexiones entrantes
    local_ip: [{172, 16, 0, 10}, {172, 16, 0, 11}],
    local_port: 2905,
    point_code: 100
  }
```

Ejemplo 4: Rol de Servidor - Aceptando Entrantes de Par Multihomed

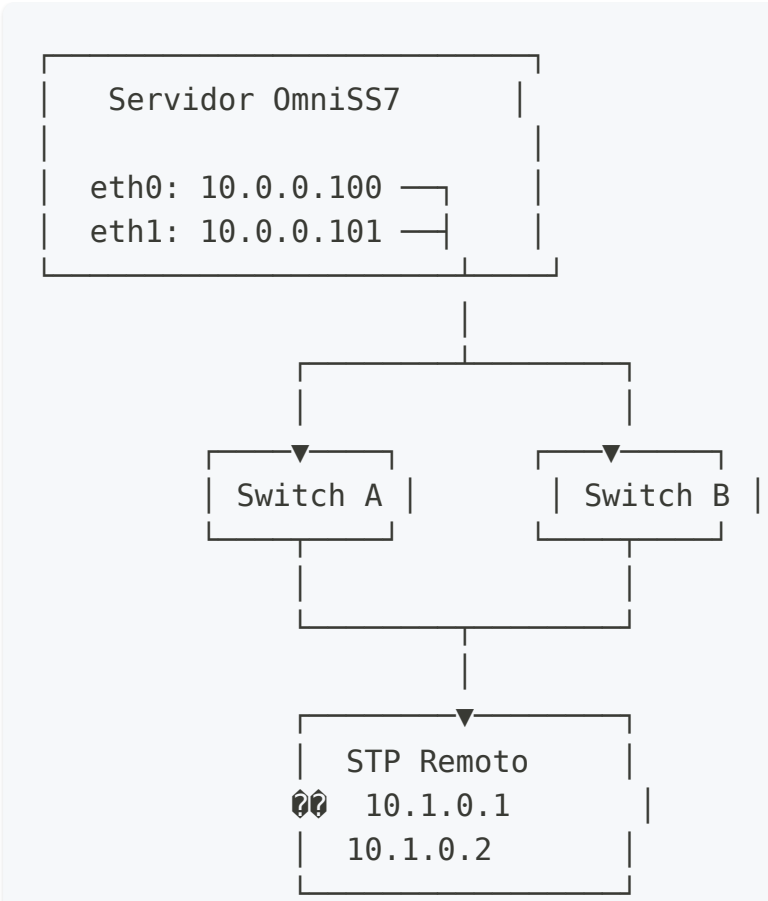
```
# Aceptando conexión entrante de un HLR multihomed
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 1,
      name: "HLR",
      role: :server,  # ENTRANTE - HLR se conecta a nosotros
      # Multihoming: nuestras IPs locales (la lista está bien)
      local_ip: [{10, 0, 1, 10}, {10, 0, 1, 11}],
      local_port: 2905,
      # IMPORTANTE: Solo tupla única - la IP que el HLR utiliza
para iniciar SCTP INIT
      # SCTP descubrirá automáticamente las otras IPs del HLR
durante el apretón de manos
      remote_ip: {10, 0, 2, 100},  # Solo IP primaria (¡NO una
lista!)
      remote_port: 0,  # Aceptar de cualquier puerto de origen
      routing_context: 1,
      point_code: 100,
      network_indicator: :international
    }
  ]
```

Ejemplo 5: Configuración Mixta (Compatible hacia Atrás)

```
# Mezcla de pares de una sola IP y multihomed
config :omniss7,
  m3ua_peers: [
    # Par legado - IP única
    %{
      peer_id: 1,
      name: "Legacy_STP",
      role: :client,
      local_ip: {10, 0, 0, 1},      # Tupla de IP única
      local_port: 0,
      remote_ip: {10, 0, 0, 10},
      remote_port: 2905,
      routing_context: 1,
      point_code: 100
    },
    # Nuevo par - multihoming
    %{
      peer_id: 2,
      name: "Redundant_STP",
      role: :client,
      local_ip: [{10, 0, 0, 2}, {10, 0, 0, 3}],  # Lista de IPs
      local_port: 0,
      remote_ip: [{10, 0, 0, 20}, {10, 0, 0, 21}],
      remote_port: 2905,
      routing_context: 2,
      point_code: 200
    }
  ]
```

Escenarios de Topología de Red

Escenario 1: Duales NICs (Despliegue Común)



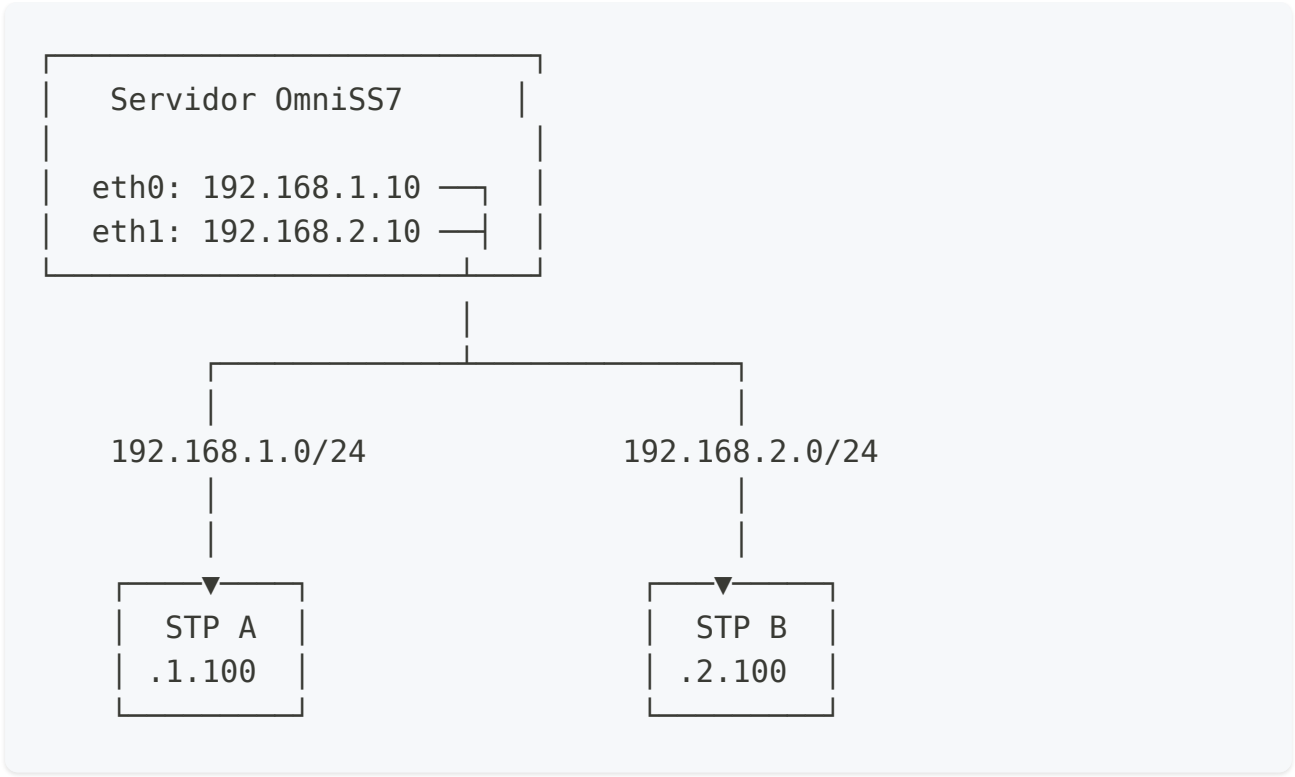
Configuración:

```
local_ip: [{10, 0, 0, 100}, {10, 0, 0, 101}]   # Ambos NICs
remote_ip: [{10, 1, 0, 1}, {10, 1, 0, 2}]      # Par remoto
```

Beneficios:

- Sobrevive a la falla de un NIC
- Sobrevive a la falla de un conmutador
- Conmutación por error automática en <1 segundo

Escenario 2: Múltiples Subredes



Configuración:

```
local_ip: [{192, 168, 1, 10}, {192, 168, 2, 10}]
remote_ip: [{192, 168, 1, 100}, {192, 168, 2, 100}]
```

Beneficios:

- Sobrevive a la falla de la subred
- Redundancia geográfica posible
- Rutas de enrutamiento independientes

Monitoreo y Registro

Cuando el multihoming está habilitado, verá mensajes de registro que indican la configuración:

Multihoming Exitoso

```
[info] Cliente SCTP multihoming: vinculado 2 IPs locales
[info] Escuchador STP multihoming habilitado: 2 IPs locales
vinculadas
```

Eventos de Conmutación de Caminos

```
[warning] [MULTIHOMING] El camino 10.0.0.100 es INALCANZABLE para
el par Partner_STP (assoc_id=1)
[info] [MULTIHOMING] El camino 10.0.0.101 es ahora PRIMARIO para
el par Partner_STP (assoc_id=1)
[info] [MULTIHOMING] El camino 10.0.0.100 ahora está DISPONIBLE
para el par Partner_STP (assoc_id=1)
```

Visualización en la Interfaz Web

La Interfaz Web muestra automáticamente información de multihoming:

Página de Estado M3UA:

- **IP Única**: Se muestra como `10.0.0.100`
- **Múltiples IPs**: Se muestra como `10.0.0.100 (+1)` o `10.0.0.100 (+2)`
- **Vista de detalles**: Muestra todas las IPs con etiquetas de primario/respaldo

Mejores Prácticas

1. Diseño de Red

- **Usar diferentes NICs** para máxima redundancia
- **Diferentes conmutadores** para sobrevivir a fallos de conmutador
- **Diferentes subredes** si es posible para diversidad de enrutamiento
- **Mismo centro de datos inicialmente** - probar antes de la separación geográfica

2. Planificación de Direcciones IP

- **La primera IP es primaria** - asegurarse de que esté en el camino más confiable
- **El orden importa** - listar IPs en orden de preferencia
- **Dirección consistente** - usar esquemas de direccionamiento similares para la solución de problemas

3. Pruebas de Conmutación por Error

```
# Deshabilitar la interfaz primaria para probar la conmutación por
error
sudo ip link set eth0 down

# Monitorear registros para la conmutación por error
tail -f /var/log/omniss7.log | grep MULTIHOMING

# Volver a habilitar la interfaz
sudo ip link set eth0 up
```

4. Ambos Lados Deben Soportar Multihoming

- **Óptimo**: Tanto local como remoto usan múltiples IPs
- **Aceptable**: Solo un lado usa multihoming
- **Nota**: La redundancia es mejor cuando ambos extremos lo soportan

5. Configuración del Firewall

```
# Permitir SCTP en todas las IPs de multihoming
iptables -A INPUT -p sctp --dport 2905 -s 10.0.0.0/24 -j ACCEPT
iptables -A INPUT -p sctp --dport 2905 -s 10.1.0.0/24 -j ACCEPT
```

Solución de Problemas

Problema: Multihoming No Funciona

Síntomas: Solo se usa la IP primaria, sin conmutación por error

Verificaciones:

1. Verifique el soporte de SCTP en Erlang: `erl -eval 'gen_sctp:open(9999, [binary, {ip, {127,0,0,1}}]).'`
2. Verifique el módulo SCTP del kernel: `lsmod | grep sctp`
3. Cargue SCTP si es necesario: `sudo modprobe sctp`
4. Verifique que ambas IPs estén configuradas en el sistema: `ip addr show`

Problema: El Camino No Cambia

Síntomas: El camino primario marcado como inactivo pero el tráfico no cambia

Verificaciones:

1. Verifique la configuración de latidos SCTP
2. Verifique que la tabla de enrutamiento tenga rutas para todos los caminos
3. Verifique que el firewall permita SCTP en todas las IPs
4. Revise los registros de monitoreo de caminos SCTP

Problema: Fluctuaciones Frecuentes de Caminos

Síntomas: Los caminos cambian constantemente entre ARRIBA y ABAJO

Verificaciones:

1. Inestabilidad de la red - verifique los enlaces físicos
2. Latido SCTP demasiado agresivo - puede necesitar ajuste
3. Firewall que elimina los latidos SCTP
4. Problemas de MTU en un camino

Consideraciones de Rendimiento

- **Sobrecarga mínima**: Los latidos SCTP son pequeños e infrecuentes
- **Sin cambios en la aplicación**: El multihoming es transparente a la capa de aplicación
- **Conmutación por error rápida**: Típicamente <1 segundo de detección y conmutación por error
- **Recuperación automática**: No se necesita intervención manual

Compatibilidad

- **Compatible hacia atrás**: El formato de tupla de IP única aún funciona
- **Despliegues mixtos**: Se pueden mezclar pares de IP única y multihomed

- **Todos los modos soportados**: Funciona en modos STP, HLR, SMSc y Cliente MAP
- **Requisito de Erlang**: Requiere Erlang con soporte SCTP compilado

Monitoreo y Alertas

Métricas Clave:

- Estado de conexión M3UA
- Tasa de éxito de solicitudes MAP
- Tiempos de respuesta de la API
- Profundidad de la cola de mensajes

Umbrales de Alerta:

- M3UA inactivo > 1 minuto
- Tasa de tiempo de espera de MAP > 10%
- Profundidad de la cola > 1000
- Tasa de error de la API > 5%

Referencia Completa de Configuración

Todos los Parámetros de Configuración

Esta sección proporciona una referencia completa de todos los parámetros de configuración disponibles en todos los modos de operación.

Configuración del Registrador (`:logger`)

```
config :logger,
  level: :debug  # :debug | :info | :warning | :error
```

Configuración de la Interfaz Web (`:control_panel`)

```
config :control_panel,
  page_order: ["/events", "/application", "/configuration"],
  web: %{
    listen_ip: "0.0.0.0",
    port: 80,
    hostname: "localhost",
    enable_tls: false,
    tls_cert: "cert.pem",
    tls_key: "key.pem"
  }
```

Parámetro	Tipo	Requerido	Predeterminado	Descripc
`page_order`	Lista de Cadenas	No	`["/events", "/application", "/configuration"]`	Orden de páginas d menú de navegaci
`web.listen_ip`	Cadena	Sí	`"0.0.0.0"`	Dirección para enla el servido web
`web.port`	Entero	Sí	`80`	Número d puerto HTTP/HT
`web.hostname`	Cadena	Sí	`"localhost"`	Nombre d servidor
`web.enable_tls`	Booleano	No	`false`	Habilitar HTTPS
`web.tls_cert`	Cadena	Si TLS está habilitado	`"cert.pem"`	Ruta del certificad TLS
`web.tls_key`	Cadena	Si TLS está habilitado	`"key.pem"`	Ruta de l clave privada T

Configuración del SocketHandler SCTP (`:omniss7`)

```
config :omniss7,
  sctp_handler: %{
    enabled: false,
    local_ip: {127, 0, 0, 1},
    local_port: 2905
  },
  enable_gt_routing: true,
  m3ua_peers: [...],
  m3ua_routes: [...],
  m3ua_gt_routes: [...]
```

Parámetro	Tipo	Requerido	Predeterminado	
`sctp_handler.enabled`	Booleano	Sí	`false`	
`sctp_handler.local_ip`	Tupla	Sí	`{127, 0, 0, 1}`	
`sctp_handler.local_port`	Entero	Sí	`2905`	
`enable_gt_routing`	Booleano	No	`false`	

Parámetros de Par M3UA:

Parámetro	Tipo	Requerido	Descripción
`peer_id`	Entero	Sí	Identificador único del par
`name`	Cadena	Sí	Nombre descriptivo del par
`role`	Átomo	Sí	`:client` o `:server`
`local_ip`	Tupla o Lista	Si `:client`	IP(s) local(es) a enlazar. Única: `{10, 0, 0, 1}` o Lista: `[{10, 0, 0, 1}, {10, 0, 0, 2}]`
`local_port`	Entero	Si `:client`	Puerto local (0 para dinámico)
`remote_ip`	Tupla o Lista	Sí	IP(s) del par remoto. Única: `{10, 0, 0, 10}` o Lista: `[{10, 0, 0, 10}, {10, 0, 0, 11}]`
`remote_port`	Entero	Si `:client`	Puerto del par remoto
`routing_context`	Entero	Sí	Contexto de enrutamiento M3UA
`point_code`	Entero	Sí	Código de punto SS7
`network_indicator`	Átomo	No	`:international` o `:national`

Parámetros de Ruta M3UA:

Parámetro	Tipo	Requerido	Descripción
`dest_pc`	Entero	Sí	Código de punto de destino
`peer_id`	Entero	Sí	Par a través del cual enrutar
`priority`	Entero	Sí	Prioridad de la ruta (menor = mayor prioridad)
`network_indicator`	Átomo	No	`:international` o `:national`

Parámetros de Ruta GT M3UA:

Parámetro	Tipo	Requerido	Descripción
`gt_prefix`	Cadena	Sí	Prefijo de Título Global a coincidir
`peer_id`	Entero	Sí	Par de destino
`priority`	Entero	Sí	Prioridad de la ruta
`description`	Cadena	No	Descripción de la ruta para registro
`source_ssn`	Entero	No	Coincidir solo si el SSN de origen coincide
`dest_ssn`	Entero	No	Reescribir el SSN de destino a este valor

Configuración del Cliente MAP (`:omniss7`)

```
config :omniss7,
  map_client_enabled: false,
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :map_client_asp,
    local_ip: {10, 0, 0, 100},
    local_port: 2905,
    remote_ip: {10, 0, 0, 1},
    remote_port: 2905,
    routing_context: 1
  }
```

Parámetro	Tipo	Requerido	Predetermi
`map_client_enabled`	Booleano	Sí	`false`
`map_client_m3ua.mode`	Cadena	Sí	`"ASP"`
`map_client_m3ua.callback`	Tupla	Sí	`{MapClient, :handle_pay []}`
`map_client_m3ua.process_name`	Átomo	Sí	`:map_client`
`map_client_m3ua.local_ip`	Tupla	Sí	-
`map_client_m3ua.local_port`	Entero	Sí	`2905`
`map_client_m3ua.remote_ip`	Tupla	Sí	-
`map_client_m3ua.remote_port`	Entero	Sí	`2905`
`map_client_m3ua.routing_context`	Entero	Sí	-

Configuración del Centro de SMS (`:omniss7`)

```
config :omniss7,
  auto_flush_enabled: false,
  auto_flush_interval: 10_000,
  auto_flush_dest_smsc: nil,
  auto_flush_tps: 10
```

Parámetro	Tipo	Requerido	Predeterminado	D
`auto_flush_enabled`	Booleano	No	`false`	Ha va au la
`auto_flush_interval`	Entero	No	`10000`	Int so co (m
`auto_flush_dest_smsc`	Cadena/nil	No	`nil`	Fil SM de to
`auto_flush_tps`	Entero	No	`10`	Ma tra po

Configuración de la API HTTP (`:omniss7`)

El backend de SMS ahora utiliza la API HTTP en lugar de conexiones directas a la base de datos.

```
config :omniss7,
  smsc_api_base_url: "https://10.5.198.200:8443",
  frontend_name: "omni-smsc01"  # Opcional: predeterminado a
hostname_SMSc
```

Parámetros de la API:

Parámetro	Tipo	Requerido	Predeterminado
`smsc_api_base_url`	Cadena	Sí	`"https://10.5.198.200:8443`
`frontend_name`	Cadena	No	`"{hostname}_SMSc"`

Endpoints de API Utilizados:

- `POST /api/frontends` - Registrar esta instancia de frontend con el backend
- `POST /api/messages_raw` - Insertar nuevos mensajes SMS
- `GET /api/messages` - Recuperar cola de mensajes (con encabezado `smsc`)
- `PATCH /api/messages/{id}` - Marcar mensaje como entregado
- `PUT /api/messages/{id}` - Actualizar estado del mensaje
- `POST /api/events` - Agregar seguimiento de eventos
- `GET /api/status` - Endpoint de verificación de salud

Registro del Frontend:

El sistema se registra automáticamente con la API del backend al inicio y vuelve a registrarse cada 5 minutos. El registro incluye:

- Nombre y tipo del frontend (SMSc)
- Nombre del host

- Tiempo de actividad en segundos
- Detalles de configuración (formato JSON)

Notas de Configuración:

- La verificación SSL está deshabilitada por defecto para certificados autofirmados
- Las solicitudes HTTP tienen un tiempo de espera de 5 segundos
- Todas las marcas de tiempo están en formato ISO 8601
- La API utiliza JSON para los cuerpos de solicitud/respuesta

---

Documentación Relacionada

- ← Volver a la Documentación Principal
- Guía STP
- Guía del Cliente MAP
- Guía del Centro de SMS
- Guía HLR

Referencia de Configuración

← Volver a la Documentación Principal

Este documento proporciona una referencia completa para todos los parámetros de configuración de OmniSS7.

Tabla de Contenidos

Descripción General

La configuración de OmniSS7 se gestiona a través de `config/runtime.exs`. El sistema soporta cuatro modos operativos:

- **Modo STP** - Punto de Transferencia de Señales para enrutamiento
- **Modo HLR** - Registro de Ubicación de Hogar para gestión de suscriptores
- **Modo SMSc** - Centro de SMS para entrega de mensajes

- **Modo CAMEL GW** - Puerta de enlace CAMEL para control de llamadas inteligente

Archivo de Configuración: `config/runtime.exs`

Flags de Modo Operativo

Controla qué características están habilitadas.

Parámetro	Tipo	Predeterminado	Descripción	I
`map_client_enabled`	Booleano	`false`	Habilitar cliente MAP y conectividad M3UA	T
`hlr_mode_enabled`	Booleano	`false`	Habilitar características específicas de HLR	H
`smsc_mode_enabled`	Booleano	`false`	Habilitar características específicas de SMSc	S
`cap_client_enabled`	Booleano	`false`	Habilitar cliente CAP para operaciones CAMEL	C C
`camelgw_mode_enabled`	Booleano	`false`	Habilitar características de la Puerta de enlace CAMEL	C C
`ussd_gateway_enabled`	Booleano	`false`	Habilitar Puerta de enlace USSD (puente HTTP/JSON)	U C

Ejemplo:

```
config :omniss7,
  map_client_enabled: true,
  hlr_mode_enabled: true,
  smsc_mode_enabled: false
```

Parámetros del Modo HLR

Configuración para el modo HLR (Registro de Ubicación de Hogar).

Configuración de la API HLR

Parámetro	Tipo	Predeterminado	Reque
`hlr_api_base_url`	Cadena	-	**Sí**
`hlr_api_verify_ssl`	Booleano	`false`	No
`hlr_service_center_gt_address`	Cadena	-	**Sí**
`smsc_service_center_gt_address`	Cadena	-	**Sí**

Ejemplo:

```
config :omniss7,
  hlr_api_base_url: "https://10.180.2.140:8443",
  hlr_api_verify_ssl: false,
  hlr_service_center_gt_address: "55512341111",
  smsc_service_center_gt_address: "55512341112"
```

Configuración de AlertServiceCenter

Cuando un suscriptor realiza UpdateLocation, el HLR envía mensajes alertServiceCenter a los GT de SMSc configurados para indicar que el suscriptor ahora es alcanzable.

Parámetro	Tipo	Predeterminado	Req
`hlr_smsc_alert_gts`	Lista de Cadenas	`[]`	No
`hlr_alert_location_expiry_seconds`	Entero	`172800`	No

Ejemplo:

```
config :omniss7,
  hlr_smsc_alert_gts: [
    "15559876543",
    "15559876544"
  ],
  hlr_alert_location_expiry_seconds: 172800  # 48 horas
```

Mapeo MSISDN ↔ IMSI

Configuración para la generación sintética de IMSI a partir de MSISDN. Para una explicación técnica detallada del algoritmo de mapeo, consulte Mapeo MSISDN ↔ IMSI en la Guía HLR.

Parámetro	Tipo	Predeterminado	Requerido	D
`hlr_imsi_plmn_prefix`	Cadena	`"50557"`	No	Pr (M pa ge si IM
`hlr_msisdn_country_code`	Cadena	`"61"`	No	Pr có pa in IM
`hlr_msisdn_nsn_offset`	Entero	`0`	No	De er do co NS (tí lo có
`hlr_msisdn_nsn_length`	Entero	`9`	No	Lo Nu Su Na ex M

Ejemplo (código de país de 2 dígitos):

```
config :omniss7,
  hlr_imsi_plmn_prefix: "50557",      # MCC 505 + MNC 57
  hlr_msisdn_country_code: "99",      # Ejemplo de código de país
de 2 dígitos
  hlr_msisdn_nsn_offset: 2,           # Omitir código de país de 2
dígitos
  hlr_msisdn_nsn_length: 9            # Extraer NSN de 9 dígitos
```

Ejemplo (código de país de 3 d��gitos):

```
config :omniss7,
  hlr_imsi_plmn_prefix: "50557",      # MCC 505 + MNC 57
  hlr_msisdn_country_code: "999",     # Ejemplo de código de país
de 3 dígitos
  hlr_msisdn_nsn_offset: 3,           # Omitir código de país de 3
dígitos
  hlr_msisdn_nsn_length: 8            # Extraer NSN de 8 dígitos
```

Importante: Establezca `nsn_offset` a la longitud de su código de país para extraer correctamente el NSN. Por ejemplo:

- Código de país "9" (1 dígito) → `nsn_offset: 1`
- Código de país "99" (2 dígitos) → `nsn_offset: 2`
- Código de país "999" (3 dígitos) → `nsn_offset: 3`

Configuración de InsertSubscriberData (ISD)

Configuración para los datos de aprovisionamiento de suscriptores enviados a los VLR durante UpdateLocation. Para una explicación detallada de la secuencia ISD y el flujo de mensajes, consulte Configuración de InsertSubscriberData en la Guía HLR.

Parámetro	Tipo	Predeterminado	Requerido
`isd_network_access_mode`	Átomo	`:packetAndCircuit`	No
`isd_send_ss_data`	Booleano	`true`	No
`isd_send_call_barring`	Booleano	`true`	No

Ejemplo:

```
config :omniss7,
  isd_network_access_mode: :packetAndCircuit,
  isd_send_ss_data: true,
  isd_send_call_barring: true
```

Configuración CAMEL

Configuración para el enrutamiento de llamadas inteligente basado en CAMEL. Para una explicación detallada de la integración CAMEL y las claves de servicio, consulte Integración CAMEL en la Guía HLR.

Parámetro	Tipo	Predeterminado	Re
`camel_service_key`	Entero	`11_110`	No
`camel_trigger_detection_point`	Átomo	`:termAttemptAuthorized`	No
`camel_gsmscf_gt_address`	Cadena	(usa GT llamado)	No

Ejemplo:

```
config :omniss7,
  camel_service_key: 11_110,
  camel_trigger_detection_point: :termAttemptAuthorized
```

Prefijos de VLR de Hogar

Configuración para distinguir entre suscriptores de hogar y de roaming. Para una explicación detallada de la detección de hogar/roaming y las operaciones PRN, consulte Manejo de Suscriptores de Roaming en la Guía HLR.

Parámetro	Tipo	Predeterminado	Requerido	Descripció
`home_vlr_prefixes`	Lista	`["5551231"]`	No	Prefijos GT de VLR considerado "red de hogar"

Ejemplo:

```
config :omniss7,
  home_vlr_prefixes: ["5551231", "5551234"]
```

Parámetros del Modo SMSc

Configuración para el modo Centro de SMS.

Configuración de la API SMSc

Parámetro	Tipo	Predeterminado	Reque
`smsc_api_base_url`	Cadena	-	**Sí**
`smsc_api_verify_ssl`	Booleano	`false`	No
`smsc_name`	Cadena	`"{hostname}_SMSc"`	No
`smsc_service_center_gt_address`	Cadena	-	**Sí**

Ejemplo:

```
config :omniss7,
  smsc_api_base_url: "https://10.179.3.219:8443",
  smsc_api_verify_ssl: false,
  smsc_name: "ipsmgw",
  smsc_service_center_gt_address: "55512341112"
```

Nota: El registro del frontend ocurre cada **5 minutos** (hardcoded) a través del módulo `SMS.FrontendRegistry`.

Configuración de Auto-Flujo

Parámetro	Tipo	Predeterminado	Requerido	
`auto_flush_enabled`	Booleano	`true`	No	Hab pro aut de S
`auto_flush_interval`	Entero	`10_000`	No	Inte pro cola
`auto_flush_dest_smsc`	Cadena	-	**Sí**	Non des
`auto_flush_tps`	Entero	`10`	No	Tasa de r (tra

Ejemplo:

```
config :omniss7,
  auto_flush_enabled: true,
  auto_flush_interval: 10_000,
  auto_flush_dest_smsc: "ipsmgw",
  auto_flush_tps: 10
```

Parámetros del Modo STP

Configuración para el modo M3UA Punto de Transferencia de Señales. Para una configuración de enrutamiento detallada y ejemplos, consulte la Guía de Configuración STP.

Servidor STP Autónomo

Parámetro	Tipo	Predeterminado	Requerido	
`sctp_handler.enabled`	Booleano	`false`	No	
`sctp_handler.local_ip`	Tupla o Lista	`{127, 0, 0, 1}`	No	
`sctp_handler.local_port`	Entero	`2905`	No	
`sctp_handler.point_code`	Entero	-	**Sí** (si está habilitado)	

Ejemplo (IP Única):

```
config :omniss7,
  sctp_handler: %{
    enabled: true,
    local_ip: {10, 179, 4, 10},
    local_port: 2905,
    point_code: 100
  }
```

Ejemplo (Multihoming SCTP):

```
config :omniss7,
  sctp_handler: %{
    enabled: true,
    # Múltiples IPs para redundancia
    local_ip: [{10, 179, 4, 10}, {10, 179, 4, 11}],
    local_port: 2905,
    point_code: 100
  }
```

> **Nota**: Para información detallada sobre la configuración y beneficios del multihoming SCTP, consulte Multihoming SCTP en la Guía Común.

Enrutamiento de Título Global

Parámetro	Tipo	Predeterminado	Requerido	Descri
`enable_gt_routing`	Booleano	`false`	No	Habilita enrutar GT ade del enrutar PC

Ejemplo:

```
config :omniss7,
  enable_gt_routing: true
```

Configuración de Pares M3UA/M2PA

Los pares se configuran a través de la lista `m3ua_peers` (soporta tanto protocolos M3UA como M2PA). Para ejemplos de configuración completos, consulte la Guía de Configuración STP y Soporte del Protocolo M2PA.

Parámetros Comunes de Pares:

Parámetro	Tipo	Predeterminado	Requerido	De
`peer_id`	Entero	-	**Sí**	Ident único
`name`	Cadena	-	**Sí**	Nom desc par
`protocol`	Átomo	`:m3ua`	No	Tipo proto `:m3u`
`role`	Átomo	`:client`	No	Rol d `:cli` `:ser` o `:s`
`local_ip`	Tupla o Lista	-	**Sí**	Direc local enlaz
`local_port`	Entero	-	**Sí**	Puert local 2905 3565
`remote_ip`	Tupla o Lista	-	**Sí**	Direc remo
`remote_port`	Entero	-	**Sí**	Puert remo
`routing_context`	Entero	-	No	Ident conte enru

Parámetro	Tipo	Predeterminado	Requerido	De
				M3U M3U
`point_code`	Entero	-	**Sí**	Códi local
`network_indicator`	Átomo	`:international`	No	Indic red: `:int` o `:n`
`initiate_connection`	Booleano	`true`	No	Si se inicia cone

Parámetros Específicos de M2PA:

Parámetro	Tipo	Predeterminado	Requerido	Descri
`adjacent_point_code`	Entero	-	**Sí** (M2PA)	Código punto d par adyace

Gestión de Sockets:

M2PA utiliza automáticamente SCTP.SocketHandler para la gestión compartida de sockets. Todos los pares M2PA utilizan el socket compartido, lo que permite que múltiples pares compartan eficientemente el mismo puerto SCTP. Para una configuración detallada, consulte Requisitos de Socket M2PA.

Ejemplo (Par M3UA):

```
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 1,
      name: "HLR_East",
      protocol: :m3ua,
      role: :sgp,
      local_ip: {10, 179, 4, 10},
      local_port: 2905,
      remote_ip: {10, 179, 4, 20},
      remote_port: 2905,
      point_code: 100,
      network_indicator: :international
    }
  ]
```

Ejemplo (Par M2PA):

```
config :omniss7,
  sctp_handler: %{
    enabled: true,
    local_ip: {10, 179, 4, 10},
    local_port: 3565,
    point_code: 100
  },
  m3ua_peers: [
    %{
      peer_id: 2,
      name: "M2PA_Link_STP_West",
      protocol: :m2pa,
      role: :client,
      local_ip: {10, 179, 4, 10},
      local_port: 3565,
      remote_ip: {10, 179, 4, 30},
      remote_port: 3565,
      point_code: 100,
      adjacent_point_code: 200
    }
  ]
```

Rutas de Código de Punto M3UA

Configuración de enrutamiento de Código de Punto. Las rutas definen qué par utilizar para alcanzar códigos de punto de destino específicos.

Parámetro	Tipo	Predeterminado	Requerido	Descripción
`m3ua_routes`	Lista de Mapas	`[]`	No	Lista de rutas de códigos de punto. Si no se especifica, las rutas se generan automáticamente a partir de los códigos de punto de los pares.

Formato de Ruta: Cada ruta en `m3ua_routes` debe ser un mapa con:

- `dest_pc`: Código de punto de destino (Entero)
- `peer_id`: ID del par a través del cual enrutar (Entero)
- `priority`: Prioridad de la ruta - valor más bajo = mayor prioridad (Entero)
- `network_indicator`: Indicador de red (Átomo): `:international` o `:national`

Ejemplo:

```
config :omniss7,
  m3ua_routes: [
    # Ruta a PC 100 a través del par 1 (mayor prioridad)
    %{dest_pc: 100, peer_id: 1, priority: 1, network_indicator:
:international},
    # Ruta a PC 200 a través del par 2
    %{dest_pc: 200, peer_id: 2, priority: 1, network_indicator:
:international},
    # Balanceo de carga: mismo dest_pc con diferentes prioridades
    %{dest_pc: 300, peer_id: 3, priority: 1, network_indicator:
:international},
    %{dest_pc: 300, peer_id: 4, priority: 2, network_indicator:
:international}
  ]
```

Rutas de Título Global M3UA

Enrutamiento basado en prefijos de Título Global con transformación avanzada de parámetros SCCP. Se utiliza la coincidencia de prefijo más largo primero, luego la prioridad.

Parámetro	Tipo	Predeterminado	Requerido	Descripció
`m3ua_gt_routes`	Lista de Mapas	`[]`	No	Lista de regla de enrutamie de Título Glob con transformacio opcionales

Formato de Ruta: Cada ruta en `m3ua_gt_routes` debe ser un mapa con:

Parámetros Básicos:

- `gt_prefix`: Prefijo de Título Global a coincidir (Cadena) - cadena vacía coincide con todos
- `peer_id`: ID del par a través del cual enrutar (Entero) - usar 0 para DESCARTAR tráfico

- `priority`: Prioridad de la ruta - valor más bajo = mayor prioridad (Entero)
- `description`: Descripción legible por humanos (Cadena)

Parámetros de Coincidencia Opcionales:

- `source_ssn`: Coincidir número de SubSistema de origen (Entero)
- `source_tt`: Coincidir tipo de Traducción de origen (Entero)
- `source_npi`: Coincidir Indicador de Plan de Numeración de origen (Entero)
- `source_nai`: Coincidir Indicador de Naturaleza de Dirección de origen (Entero)

Parámetros de Transformación Opcionales:

- `dest_ssn`: Transformar SSN en el mensaje reenviado (Entero)
- `dest_tt`: Transformar Tipo de Traducción en el mensaje reenviado (Entero)
- `dest_npi`: Transformar Indicador de Plan de Numeración en el mensaje reenviado (Entero)
- `dest_nai`: Transformar Indicador de Naturaleza de Dirección en el mensaje reenviado (Entero)

Valores Comunes:

- **Tipo de Traducción (TT):** 0=Desconocido, 1=Internacional, 2=Nacional, 3=Específico de la Red
- **Plan de Numeración (NPI):** 0=Desconocido, 1=ISDN(E.164), 6=Móvil(E.212)
- **Naturaleza de Dirección (NAI):** 0=Desconocido, 1=Suscriptor, 3=Nacional, 4=Internacional
- **Número de SubSistema (SSN):** 6=HLR, 7=VLR, 8=MSC, 9=EIR, etc.

Ejemplo:

```
config :omniss7,
  m3ua_gt_routes: [
    # Enrutamiento básico por prefijo
    %{gt_prefix: "1234", peer_id: 1, priority: 1, description:
"Números de EE. UU."},
    %{gt_prefix: "44", peer_id: 2, priority: 1, description:
"Números del Reino Unido"},

    # Transformación de Tipo de Traducción
    %{
      gt_prefix: "61",
      peer_id: 3,
      priority: 1,
      description: "Números australianos: transformación TT 0→1",
      source_tt: 0,  # Coincidir TT=0 (Desconocido)
      dest_tt: 1     # Transformar a TT=1 (Internacional)
    },

    # Transformación de NPI
    %{
      gt_prefix: "49",
      peer_id: 1,
      priority: 1,
      description: "Números alemanes: conversión de Móvil→ISDN
NPI",
      source_npi: 6,  # Coincidir NPI=6 (Móvil/E.212)
      dest_npi: 1     # Transformar a NPI=1 (ISDN/E.164)
    },

    # Transformación combinada con enrutamiento SSN
    %{
      gt_prefix: "86",
      source_ssn: 8,   # Coincidir SSN=8 (MSC)
      peer_id: 3,
      dest_ssn: 6,     # Reescribir a SSN=6 (HLR)
      priority: 1,
      description: "Tráfico chino: normalización completa",
      source_tt: 0,
      dest_tt: 2,
      source_npi: 6,
      dest_npi: 1,
      source_nai: 4,
      dest_nai: 3
```

```
    },

    # Ruta predeterminada/alternativa
    %{
      gt_prefix: "",
      peer_id: 1,
      priority: 99,
      description: "Ruta de respaldo predeterminada"
    }
  ]
```

Parámetros del Modo CAMEL Gateway

Configuración para el modo CAMEL Gateway (protocolo CAP).

Flags del Modo CAMEL

Habilitar características CAMEL/CAP (establecer `cap_client_enabled: true` y `camelgw_mode_enabled: true` en Flags de Modo Operativo).

Configuración del Protocolo CAP

Parámetro	Tipo	Predeterminado	Requerido	D
`cap_version`	Átomo	`:v2`	No	Ve pr `:v` o
`camel_gsmscf_gt_address`	Cadena	(usa GT llamado)	No	Tí gs pr pa re CA

Mapeo de Versiones CAP:

- `:v1` → OID de Contexto de Aplicación: 0.4.0.0.1.0.50.0
- `:v2` → OID de Contexto de Aplicación: 0.4.0.0.1.0.50.1 (predeterminado - más ampliamente soportado)
- `:v3` → OID de Contexto de Aplicación: 0.4.0.0.1.21.3.4
- `:v4` → OID de Contexto de Aplicación: 0.4.0.0.1.23.3.4

Nota: Las solicitudes entrantes se detectan automáticamente a partir de su OID de contexto de aplicación y las respuestas coinciden con la versión de la solicitud.

Ejemplo:

```
config :omniss7,
  cap_client_enabled: true,
  camelgw_mode_enabled: true,
  cap_version: :v2,
  camel_gsmscf_gt_address: "68988411553"
```

Integración CGrateS

Integración de carga en tiempo real con CGrateS para facturación prepago/postpago.

Parámetro	Tipo	Predeterminado	Requerido	D
`cgrates_enabled`	Booleano	`false`	No	Hab inte CGr
`cgrates_url`	Cadena	-	**Sí** (si está habilitado)	URL fina CGr
`cgrates_tenant`	Cadena	`"cgrates.org"`	No	Ider inqu CGr
`cgrates_request_type`	Cadena	`"*prepaid"`	No	Tipo `"*p` `"*p` `"*p`
`cgrates_timeout`	Entero	`5000`	No	Tier de s CGr mili

Ejemplo:

```
config :omniss7,
  cgrates_enabled: true,
  cgrates_url: "http://localhost:2080/jsonrpc",
  cgrates_tenant: "cgrates.org",
  cgrates_request_type: "*prepaid",
  cgrates_timeout: 5000
```

Conexión M3UA CAP

La Puerta de enlace CAMEL utiliza una conexión M3UA separada para operaciones CAP.

Parámetro	Tipo	Predeterminado	Requerido	Descripci
`cap_client_m3ua`	Mapa	-	**Sí**	Configuración conexión M3U CAP (misma estructura qu `map_client_` incluyendo parámetros opcionales `op` `dpc`)

Ejemplo:

```
config :omniss7,
  cap_client_m3ua: %{
    mode: "ASP",
    callback: {CapClient, :handle_payload, []},
    process_name: :camelgw_client_asp,
    local_ip: {10, 5, 198, 200},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 4,
    opc: 5013,    # Código de Punto de Origen
    dpc: 5011     # Código de Punto de Destino
  }
```

Parámetros de NAT de Título Global

La Traducción de Dirección de Título Global permite diferentes GT de respuesta basados en el prefijo de la parte que llama, el prefijo de la parte llamada, o ambos. Las reglas se emparejan por peso (menor = mayor prioridad), luego por especificidad de prefijo (prefijo combinado más largo = más específico). Para una explicación detallada y ejemplos, consulte la Guía de NAT de Título Global.

Parámetro	Tipo	Predeterminado	Requerido	Descr
`gt_nat_enabled`	Booleano	`false`	No	Habilitar/d la función
`gt_nat_rules`	Lista de Mapas	`[]`	**Sí** (si está habilitado)	Lista de re NAT GT co coincidenc prefijo

Formato de Regla: Cada regla en `gt_nat_rules` debe ser un mapa con:

- `calling_prefix`: Prefijo de cadena para coincidir con GT de llamada (opcional)
- `called_prefix`: Prefijo de cadena para coincidir con GT de llamada (opcional)
- `weight`: Prioridad entera (menor = mayor prioridad) - predeterminado: 100
- `response_gt`: Título Global a utilizar en las respuestas (requerido)

Prioridad de Coincidencia:

1. Las reglas se emparejan por `weight` (valor menor = mayor prioridad)
2. Si los pesos son iguales, la longitud del prefijo combinado más largo gana
3. Tanto `calling_prefix` como `called_prefix` pueden usarse juntos para coincidencias precisas

Ejemplo:

```
config :omniss7,
  gt_nat_enabled: true,
  gt_nat_rules: [
    # Alta prioridad: Coincidir tanto llamando desde "8772" COMO
llamado a "555"
    %{calling_prefix: "8772", called_prefix: "555", weight: 1,
response_gt: "111111"},

    # Prioridad media: Coincidir solo llamando desde "8772"
    %{calling_prefix: "8772", weight: 10, response_gt:
"68988411553"},

    # Prioridad media: Coincidir solo llamado a "555"
    %{called_prefix: "555", weight: 10, response_gt:
"68988411554"},

    # Coincidir solo llamando desde "8773"
    %{calling_prefix: "8773", weight: 10, response_gt:
"68988411554"},

    # Regla de respaldo comodín (coincide con todo, mayor peso)
    %{weight: 100, response_gt: "68988411555"}
  ]
```

Véase También: Guía de NAT GT para un uso y ejemplos detallados.

Parámetros de Conexión M3UA

Configuración de conexión M3UA para el modo cliente MAP. Para un uso y ejemplos detallados, consulte la Guía del Cliente MAP.

Parámetro	Tipo	Predeterminado
`map_client_m3ua.mode`	Cadena	-
`map_client_m3ua.callback`	Tupla	-
`map_client_m3ua.process_name`	Átomo	-
`map_client_m3ua.local_ip`	Tupla o Lista	-
`map_client_m3ua.local_port`	Entero	`2905`
`map_client_m3ua.remote_ip`	Tupla o Lista	-
`map_client_m3ua.remote_port`	Entero	`2905`
`map_client_m3ua.routing_context`	Entero	-
`map_client_m3ua.opc`	Entero	`5013`

Parámetro	Tipo	Predeterminado
`map_client_m3ua.dpc`	Entero	`5011`
`map_client_m3ua.receive_watchdog`	Booleano	`true`
`map_client_m3ua.receive_watchdog_idle`	Entero	`15`

Ejemplo (IP Única):

```
config :omniss7,
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1,
    opc: 5013,    # Código de Punto de Origen (2-114-5)
    dpc: 5011     # Código de Punto de Destino (2-114-3)
  }
```

> **Formato de Código de Punto**: Los códigos de punto en formato `X-Y-Z` se convierten a enteros como: `(X * 2048) + (Y * 8) + Z`. Por ejemplo, `2-114-5` = `(2 * 2048) + (114 * 8) + 5` = `5013`.

Ejemplo (Multihoming SCTP):

```
config :omniss7,
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    # Múltiples IPs locales para redundancia
    local_ip: [{10, 179, 4, 11}, {10, 179, 4, 12}],
    local_port: 2905,
    # Múltiples IPs remotas para redundancia STP
    remote_ip: [{10, 179, 4, 10}, {10, 179, 4, 20}],
    remote_port: 2905,
    routing_context: 1
  }
```

> **Nota**: Para información detallada sobre la configuración y beneficios del multihoming SCTP, consulte Multihoming SCTP en la Guía Común.

Receive Watchdog

El **watchdog de recepción** monitorea las conexiones SCTP en busca de sockets zombis: asociaciones que permanecen en un estado ESTABLECIDO a nivel del sistema operativo pero donde el extremo remoto ha dejado de enviar datos silenciosamente. Sin el watchdog, una conexión muerta puede no ser detectada hasta que falle el siguiente intento de envío.

Cualquier carga útil SCTP recibida restablece el temporizador de inactividad, incluyendo respuestas M3UA BEAT, mensajes NOTIFY y datos de aplicación. Según RFC 4666 §3.8, M3UA BEAT es opcional: SCTP ya realiza un heartbeat obligatorio a nivel de transporte. En entornos donde el SG remoto no envía tráfico de capa de aplicación periódico (y no envía M3UA BEATs), deshabilitar el watchdog evita ciclos de reconexión innecesarios.

Parámetro	Tipo	Predeterminado	Descripción
`receive_watchdog`	Booleano	`true`	Habilitar o deshabilitar el watchdog de recepción. Cuand `false`, las conexiones inactivas nunca son cerradas por watchdog; el heartbeat de cap de transporte SC sigue funcionand normalmente.
`receive_watchdog_idle`	Entero	`15`	Segundos de inactividad antes de que el watchd cierre la conexió active una reconexión. No tiene efecto cuando `receive_watchdo` `false`.

Ejemplo — Deshabilitando el watchdog para un cliente ASP:

```
config :omniss7,
  map_client_m3ua: %{
    mode: "ASP",
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1,
    receive_watchdog: false        # Deshabilitar — el SG remoto
no envía tráfico periódico
  }
```

Ejemplo — Personalizando el umbral de inactividad:

```
config :omniss7,
  map_client_m3ua: %{
    mode: "ASP",
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1,
    receive_watchdog: true,
    receive_watchdog_idle: 30      # Cerrar después de 30 s de
silencio (predeterminado: 15 s)
  }
```

Parámetros de Infraestructura

Configuración para componentes de infraestructura del sistema, incluyendo licencias, interfaz web, servidor API y registro.

Configuración de Licencia

Parámetro	Tipo	Predeterminado
`license_client.license_server_api_urls`	Lista de Cadenas	-
`license_client.licensee`	Cadena	-

Ejemplo:

```
config :license_client,
  license_server_api_urls: ["https://localhost:10443/api"],
  licensee: "Omnitouch Network Services Pty. Ltd."
```

Interfaz Web del Panel de Control

Parámetro	Tipo	Pred
`control_panel.parent_application_readable_name`	Cadena	`"OmniSS7`
`control_panel.use_additional_pages`	Lista de Tuplas	`[]`
`control_panel.page_order`	Lista de Cadenas	`[]`
`ControlPanelWeb.Endpoint.url.host`	Cadena	`"0.0.0.0"`
`ControlPanelWeb.Endpoint.https.port`	Entero	`8087`
`ControlPanelWeb.Endpoint.https.keyfile`	Cadena	`"priv/cer`
`ControlPanelWeb.Endpoint.https.certfile`	Cadena	`"priv/cer`

Parámetro	Tipo	Pred

Ejemplo:

```
config :control_panel, ControlPanelWeb.Endpoint,
  url: [host: "0.0.0.0", path: "/"],
  https: [
    port: 8087,
    keyfile: "priv/cert/omnitouch.pem",
    certfile: "priv/cert/omnitouch.crt"
  ],
  parent_application_readable_name: "OmniSS7 Stack STP"

config :control_panel,
  use_additional_pages: [
    {SS7.Web.EventsLive, "/events", "Eventos SS7"},
    {SS7.Web.M3UAStatusLive, "/m3ua", "Pares"}
  ],
  page_order: ["/events", "/m3ua", "/application",
"/configuration"]
```

Servidor API REST

Parámetro	Tipo	Predeterminado	Re
`start_http_server`	Booleano	`true`	No
`api_ex.api.port`	Entero	`8445`	No
`api_ex.api.listen_ip`	Cadena	`"0.0.0.0"`	No
`api_ex.api.product_name`	Cadena	`"OmniSS7"`	No
`api_ex.api.title`	Cadena	`"API - OmniSS7"`	No
`api_ex.api.hostname`	Cadena	`"localhost"`	No
`api_ex.api.enable_tls`	Booleano	`true`	No
`api_ex.api.tls_cert_path`	Cadena	`"priv/cert/omnitouch.crt"`	No
`api_ex.api.tls_key_path`	Cadena	`"priv/cert/omnitouch.pem"`	No

Ejemplo:

```
config :omniss7,
  start_http_server: true

config :api_ex,
  api: %{
    port: 8445,
    listen_ip: "0.0.0.0",
    product_name: "OmniSS7",
    title: "API - OmniSS7",
    hostname: "localhost",
    enable_tls: true,
    tls_cert_path: "priv/cert/omnitouch.crt",
    tls_key_path: "priv/cert/omnitouch.pem"
  }
```

Puntos finales de la API:

- API REST: `https://[server-ip]:8445/api/*`
- Swagger UI: `http://[server-ip]:8080/swagger`
- Métricas de Prometheus: `http://[server-ip]:8080/metrics`

Configuración de Registro

Parámetro	Tipo	Predeterminado
`logger.level`	Átomo	`:debug`
`logger.backends`	Lista	`[:console]`
`logger.default_formatter.format`	Cadena	-
`logger.default_formatter.metadata`	Lista	`[]`
`logger.default_formatter.truncate`	Átomo/Entero	`:infinity`

Ejemplo:

```
config :logger,
  level: :debug,
  backends: [:console, SS7.Web.LoggerBackend]

config :logger, :default_formatter,
  format: "[$date] [$time] [$level] $message\n",
  metadata: [:error_code, :file],
  truncate: :infinity
```

Parámetros de Base de Datos

Configuración para la persistencia de la base de datos Mnesia.

Parámetro	Tipo	Predeterminado	Requerido	Desc
`mnesia_storage_type`	Átomo	`:disc_copies`	No	Tipo de almacer de Mne `:disc_` `:ram_c`

Ejemplo:

```
config :omniss7,
  mnesia_storage_type: :disc_copies  # Producción
  # mnesia_storage_type: :ram_copies  # Solo para pruebas
```

Tipos de Almacenamiento:

- `:disc_copies` - Almacenamiento persistente en disco (sobrevive a reinicios) - **Recomendado para producción**
- `:ram_copies` - Solo en memoria (se pierde al reiniciar) - Solo para pruebas

Tablas de Mnesia:

- `m3ua_peer` - Conexiones de pares M3UA
- `m3ua_route` - Rutas de Código de Punto
- `m3ua_gt_route` - Rutas de Título Global

Ubicación: directorio `Mnesia.{node_name}/`

Valores Hardcoded

Los siguientes valores están **hardcoded en el código fuente** y no pueden ser cambiados a través de la configuración.

Tiempos de Espera

Valor	Impacto	Solución Alternativa
Tiempo de espera de solicitud MAP: **10 segundos**	Todas las operaciones MAP expiran después de 10s	Modificar el código fuente
Tiempo de espera ISD: **10 segundos**	Cada mensaje ISD expira después de 10s	Modificar el código fuente

Servidor HTTP

Valor	Impacto	Solución Alternativa
IP HTTP: **0.0.0.0**	El servidor de métricas/Swagger escucha en todas las interfaces	Modificar el código fuente
Puerto HTTP: **8080**	El punto final de métricas/Swagger se ejecuta en el puerto 8080	Modificar el código fuente

Intervalos de Registro

Valor	Impacto	Solución Alternativa
Registro del frontend: **5 minutos**	SMSc se registra con el backend cada 5 min	Modificar el código fuente

Actualización Automática de la Interfaz Web

Página	Intervalo
Gestión de Enrutamiento	5 segundos
Suscriptores Activos	2 segundos

Ejemplos de Configuración

Configuración Mínima de HLR

```
config :omniss7,
  map_client_enabled: true,
  hlr_mode_enabled: true,
  smsc_mode_enabled: false,

  hlr_api_base_url: "https://10.180.2.140:8443",
  hlr_service_center_gt_address: "55512341111",
  smsc_service_center_gt_address: "55512341112",

  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1
  }
```

Configuración Mínima de SMSc

```
config :omniss7,
  map_client_enabled: true,
  hlr_mode_enabled: false,
  smsc_mode_enabled: true,

  smsc_api_base_url: "https://10.179.3.219:8443",
  smsc_name: "ipsmgw",
  smsc_service_center_gt_address: "55512341112",

  auto_flush_enabled: true,
  auto_flush_interval: 10_000,
  auto_flush_dest_smsc: "ipsmgw",
  auto_flush_tps: 10,

  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :stp_client_asp,
    local_ip: {10, 179, 4, 12},
    local_port: 2905,
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1
  }
```

STP con Servidor Autónomo

```
config :omniss7,
  map_client_enabled: true,
  hlr_mode_enabled: false,
  smsc_mode_enabled: false,

  enable_gt_routing: true,
  mnesia_storage_type: :disc_copies,

  sctp_handler: %{
    enabled: true,
    local_ip: {10, 179, 4, 10},
    local_port: 2905,
    point_code: 100
  },

  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :stp_client_asp,
    local_ip: {10, 179, 4, 10},
    local_port: 2906,
    remote_ip: {10, 179, 4, 11},
    remote_port: 2905,
    routing_context: 1
  }
```

Resumen

Total de Parámetros de Configuración: 75+

Por Categoría:

- Modo Operativo: 5 parámetros
- Modo HLR: 17 parámetros
- Modo SMSc: 8 parámetros

- Modo STP: 5+ parámetros (más listas m3ua_peers, m3ua_routes, m3ua_gt_routes)
- Modo CAMEL GW: 14 parámetros
- NAT de Título Global: 2 parámetros
- Conexión M3UA: 8 parámetros
- Infraestructura (Licencia, Web, API, Registro): 23 parámetros
- Base de Datos: 1 parámetro

Parámetros Requeridos por Modo:

Modo HLR:

- `hlr_api_base_url`
- `hlr_service_center_gt_address`
- `smsc_service_center_gt_address`
- Todos los parámetros `map_client_m3ua.*` (8)

Modo SMSc:

- `smsc_api_base_url`
- `smsc_service_center_gt_address`
- `auto_flush_dest_smsc` (si se habilita el auto-flush)
- Todos los parámetros `map_client_m3ua.*` (8)

Modo STP:

- `sctp_handler.point_code` (si se habilita el manejador SCTP)
- `sctp_handler.local_ip`
- `sctp_handler.local_port`

Modo CAMEL GW:

- `cgrates_url` (si se habilita CGrateS)
- Todos los parámetros `cap_client_m3ua.*` (8)

Infraestructura:

- `license_client.license_server_api_urls`

- `license_client.licensee`

Documentación Relacionada

- **Guía HLR** - Configuración específica de HLR
- **Guía SMSc** - Configuración específica de SMSc
- **Guía STP** - Configuración de enrutamiento STP
- **Guía API** - Referencia de API REST
- **Guía de Puerta de Enlace USSD** - Configuración de la Puerta de Enlace USSD y protocolo de callback HTTP
- **Guía de Interfaz Web** - Documentación de la interfaz web

Guía de NAT de Título Global

Descripción general

La Traducción de Direcciones de Título Global (GT NAT) es una función que permite a OmniSS7 responder con diferentes direcciones de Título Global basadas en el prefijo GT de la parte que llama, el prefijo GT de la parte llamada, o una combinación de ambos. Esto es esencial cuando se opera con múltiples Títulos Globales y se necesita asegurar que las respuestas utilicen el GT correcto según qué red o par está llamando y/o qué GT han llamado.

Novedades (GT NAT Mejorado)

La función GT NAT ha sido mejorada con potentes nuevas capacidades:

Nuevas características

1. **Coincidencia de Prefijo de Parte Llamada**: Las reglas ahora pueden coincidir en `called_prefix` además de `calling_prefix`
2. **Coincidencia Combinada**: Las reglas pueden coincidir en ambos prefijos de llamada Y de llamada simultáneamente
3. **Priorización Basada en Peso**: Las reglas ahora utilizan un campo `weight` (menor = mayor prioridad) en lugar de solo la longitud del prefijo
4. **Coincidencia Flexible**: Ahora puedes crear reglas con:
    - Solo prefijo de llamada
    - Solo prefijo de parte llamada
    - Ambos prefijos de llamada y parte llamada
    - Ninguno (regla comodín/por defecto)

Nuevo Formato de Regla

Campos requeridos:

- `weight`: Prioridad entera (menor = mayor prioridad)
- `response_gt`: El GT con el que responder

Campos opcionales (se recomienda al menos uno para coincidencia específica):

- `calling_prefix`: Coincidir en el prefijo GT de la parte que llama
- `called_prefix`: Coincidir en el prefijo GT de la parte llamada

Ejemplo:

```
gt_nat_rules: [
  # Regla específica con ambos prefijos - mayor prioridad
  %{calling_prefix: "8772", called_prefix: "555", weight: 1,
response_gt: "111111"},

  # Reglas específicas - prioridad media
  %{calling_prefix: "8772", weight: 10, response_gt: "222222"},
  %{called_prefix: "555", weight: 10, response_gt: "333333"},

  # Comodín por defecto - menor prioridad
  %{weight: 100, response_gt: "999999"}
]
```

Casos de uso

Operación Multi-Rede

Cuando tienes múltiples redes pares y cada una espera respuestas de un GT específico:

- **Red A** llama a tu GT `111111` y espera respuestas de `111111`
- **Red B** llama a tu GT `222222` y espera respuestas de `222222`

Sin GT NAT, necesitarías instancias separadas o enrutamiento complejo. Con GT NAT, una sola instancia de OmniSS7 puede manejar esto de manera inteligente.

Escenarios de Roaming

Cuando operas como un HLR o SMSc con acuerdos de roaming:

- Suscriptores de la **red de origen** utilizan GT `555000`
- **Socio de roaming 1** utiliza GT `555001`
- **Socio de roaming 2** utiliza GT `555002`

GT NAT asegura que cada socio reciba respuestas del GT correcto al que están configurados para enrutar.

Pruebas y Migración

Durante migraciones de red o pruebas:

- Migrar gradualmente el tráfico del antiguo GT al nuevo GT
- Mantener ambos GTs durante el período de transición
- Enrutar respuestas según qué GT utilizó el llamador

Cómo funciona

Flujo de Traducción de Direcciones

1. **Solicitud Entrante**: OmniSS7 recibe un mensaje SCCP con:
    - GT de Parte Llamada: `55512341112` (tu GT)
    - GT de Parte Llamante: `877234567` (su GT)
2. **Búsqueda de GT NAT**: El sistema verifica el GT de llamada `877234567` contra las reglas de prefijo configuradas
3. **Coincidencia de Prefijo**: Encuentra el prefijo coincidente más largo (por ejemplo, `8772` coincide con `877234567`)

4. **Selección de GT de Respuesta**: Utiliza `response_gt` de la regla coincidente (por ejemplo, `55512341112`)

5. **Respuesta Enviada**: La respuesta SCCP utiliza:

    - GT de Parte Llamada: `877234567` (invertido - su GT)
    - GT de Parte Llamante: `55512341112` (GT NAT'd)

Tipos de Respuesta Afectados

GT NAT se aplica a múltiples capas de la pila SS7:

Capa SCCP (Todas las Respuestas)

- Direcciones GT de Parte Llamada/Llamante en todos los mensajes de respuesta
- Reconocimientos de ISD (InsertSubscriberData)
- Respuestas de UpdateLocation
- Respuestas de error

Capa MAP (Específica de Operación)

- **Respuestas SRI-for-SM**: `networkNode-Number` (dirección GT de SMSc)
- **UpdateLocation**: `hlr-Number` en respuestas
- **InsertSubscriberData**: GT de HLR en mensajes ISD

Configuración

Configuración Básica

Agrega a `config/runtime.exs`:

```
config :omniss7,
  # Habilitar GT NAT
  gt_nat_enabled: true,

  # Definir reglas de GT NAT
  gt_nat_rules: [
    # Regla 1: Llamadas desde el prefijo "8772" obtienen respuesta
de "55512341112"
    %{calling_prefix: "8772", response_gt: "55512341112"},

    # Regla 2: Llamadas desde el prefijo "8773" obtienen respuesta
de "55512341111"
    %{calling_prefix: "8773", response_gt: "55512341111"},

    # Regla por defecto (prefijo vacío coincide con todo)
    %{calling_prefix: "", response_gt: "55512311555"}
  ]
```

Parámetros de Configuración

Para referencia completa de configuración, consulta Parámetros de NAT de Título Global en la Referencia de Configuración.

Parámetro	Tipo	Requerido	Descripción
`gt_nat_enabled`	Booleano	Sí	Habilitar/deshabilitar la función GT NAT
`gt_nat_rules`	Lista de Mapas	Sí (si está habilitado)	Lista de reglas de coincidencia de prefijos

Formato de Regla

Cada regla es un mapa con las siguientes claves:

```
%{
  calling_prefix: "8772",      # (Opcional) Prefijo para coincidir
con el GT de llamada
  called_prefix: "555",        # (Opcional) Prefijo para coincidir
con el GT de llamada
  weight: 10,                  # (Requerido) Valor de prioridad
(menor = mayor prioridad)
  response_gt: "55512341112"   # (Requerido) GT a utilizar en las
respuestas
}
```

Campos de Regla:

- **`calling_prefix`** (Opcional): Prefijo de cadena para coincidir con el GT de llamada entrante
  - La coincidencia se realiza mediante `String.starts_with?/2`
  - La cadena vacía `""` o `nil` actúa como comodín (coincide con cualquier GT de llamada)
  - Se puede omitir para coincidir con cualquier GT de llamada
- **`called_prefix`** (Opcional): Prefijo de cadena para coincidir con el GT de llamada entrante
  - La coincidencia se realiza mediante `String.starts_with?/2`
  - La cadena vacía `""` o `nil` actúa como comodín (coincide con cualquier GT de llamada)
  - Se puede omitir para coincidir con cualquier GT de llamada
- **`weight`** (Requerido): Valor de prioridad entero
  - Menor peso = mayor prioridad (procesado primero)
  - Debe ser >= 0
  - Se utiliza como criterio de ordenación principal para las reglas de coincidencia
- **`response_gt`** (Requerido): La dirección de Título Global a utilizar en las respuestas

- Debe ser una cadena de número E.164 válida
- Debe coincidir con uno de tus GTs configurados

Al menos uno de `calling_prefix` o `called_prefix` debe ser especificado para el enrutamiento específico. Ambos pueden ser omitidos para una regla comodín/por defecto.

Lógica de Coincidencia de Reglas

Las reglas se evalúan por **peso primero (ascendente), luego por especificidad combinada de prefijo**:

Algoritmo de Coincidencia:

1. Filtrar reglas donde todos los prefijos especificados coinciden
   - Si `calling_prefix` está configurado, debe coincidir con el GT de llamada
   - Si `called_prefix` está configurado, debe coincidir con el GT de llamada
   - Si ambos están configurados, ambos deben coincidir
   - Si ninguno está configurado, la regla actúa como un comodín
2. Ordenar las reglas coincidentes por:
   - **Primario**: Peso (ascendente - valores más bajos primero)
   - **Secundario**: Longitud combinada del prefijo (descendente - más largo = más específico)
3. Devolver la primera regla coincidente

Ejemplos:

```
# Ejemplo de reglas
gt_nat_rules: [
  # Peso 1: Mayor prioridad - coincide con ambos prefijos
  %{calling_prefix: "8772", called_prefix: "555", weight: 1,
response_gt: "111111"},

  # Peso 10: Prioridad media - reglas específicas
  %{calling_prefix: "8772", weight: 10, response_gt: "222222"},  #
Solo llamada
  %{called_prefix: "555", weight: 10, response_gt: "333333"},    #
Solo llamada

  # Peso 100: Menor prioridad - comodín por defecto
  %{weight: 100, response_gt: "444444"}  # Coincide con todo
]

# Ejemplos de coincidencia:
# Llamada: "877234567", Llamada: "555123" -> "111111" (peso 1,
ambos coinciden)
# Llamada: "877234567", Llamada: "999999" -> "222222" (peso 10,
solo llamada)
# Llamada: "999999999", Llamada: "555123" -> "333333" (peso 10,
solo llamada)
# Llamada: "999999999", Llamada: "888888" -> "444444" (peso 100,
comodín)
```

Ejemplos

Ejemplo 1: Dos Socios de Red

Escenario: Operas un SMSc con dos socios de red. Cada uno espera respuestas de un GT diferente.

```
config :omniss7,
  gt_nat_enabled: true,

  # GT de SMSc por defecto (utilizado cuando GT NAT está
deshabilitado o no coincide ninguna regla)
  smsc_service_center_gt_address: "5551000",

  # Reglas de GT NAT para socios
  gt_nat_rules: [
    # Socio A (prefijo 4412) espera respuestas del GT 5551001
    %{calling_prefix: "4412", weight: 10, response_gt: "5551001"},

    # Socio B (prefijo 4413) espera respuestas del GT 5551002
    %{calling_prefix: "4413", weight: 10, response_gt: "5551002"},

    # Por defecto: usar GT estándar de SMSc (comodín por defecto)
    %{weight: 100, response_gt: "5551000"}
  ]
```

Flujo de Tráfico:

```
SRI-for-SM entrante de 44121234567:
  GT Llamada: 5551001 (tu GT que utiliza el Socio A)
  GT Llamante: 44121234567 (GT del Socio A)

Búsqueda de GT NAT:
  "44121234567" coincide con el prefijo "4412"
  GT de respuesta seleccionado: "5551001"

Respuesta SRI-for-SM a 44121234567:
  GT Llamada: 44121234567 (invertido)
  GT Llamante: 5551001 (NAT'd)
  networkNode-Number: 5551001 (en respuesta MAP)
```

Ejemplo 2: HLR con GTs Regionales

Escenario: HLR nacional con diferentes GTs por región.

```
config :omniss7,
  gt_nat_enabled: true,
  hlr_service_center_gt_address: "555000",  # GT de HLR por
defecto

  gt_nat_rules: [
    # VLRs de la región norte (prefijo 5551)
    %{calling_prefix: "5551", weight: 10, response_gt: "555100"},

    # VLRs de la región sur (prefijo 5552)
    %{calling_prefix: "5552", weight: 10, response_gt: "555200"},

    # VLRs de la región oeste (prefijo 5553)
    %{calling_prefix: "5553", weight: 10, response_gt: "555300"},

    # Por defecto para otras regiones (comodín)
    %{weight: 100, response_gt: "555000"}
  ]
```

Ejemplo 3: Escenario de Migración

Escenario: Migrando gradualmente del antiguo GT al nuevo GT.

```
config :omniss7,
  gt_nat_enabled: true,
  hlr_service_center_gt_address: "123456789",  # Antiguo GT (por
defecto)

  gt_nat_rules: [
    # Redes migradas (ya actualizaron sus configuraciones)
    %{calling_prefix: "555", weight: 10, response_gt:
"987654321"},  # Nuevo GT
    %{calling_prefix: "666", weight: 10, response_gt:
"987654321"},  # Nuevo GT

    # Todos los demás todavía utilizan el antiguo GT (comodín)
    %{weight: 100, response_gt: "123456789"}  # Antiguo GT
  ]
```

Ejemplo 4: Coincidencia de Prefijo de Parte Llamada (NUEVO)

Escenario: Tienes múltiples GTs para diferentes servicios y quieres responder con el GT correcto según qué GT fue llamado.

```
config :omniss7,
  gt_nat_enabled: true,

  gt_nat_rules: [
    # Cuando llaman a tu GT de SMS (5551xxx), responde con ese GT
    %{called_prefix: "5551", weight: 10, response_gt: "555100"},

    # Cuando llaman a tu GT de Voz (5552xxx), responde con ese GT
    %{called_prefix: "5552", weight: 10, response_gt: "555200"},

    # Cuando llaman a tu GT de Datos (5553xxx), responde con ese
GT
    %{called_prefix: "5553", weight: 10, response_gt: "555300"},

    # Comodín por defecto
    %{weight: 100, response_gt: "555000"}
  ]
```

Flujo de Tráfico:

```
Solicitud entrante al GT Llamado: 555100 (tu GT de SMS)
GT Llamante: 441234567 (cualquier llamador)

Búsqueda de GT NAT:
  GT Llamado "555100" coincide con el prefijo "5551"
  GT de respuesta seleccionado: "555100"

La respuesta utiliza GT Llamante: 555100 (coincide con lo que
llamaron)
```

Ejemplo 5: Coincidencia Combinada de Prefijo de Llamada + Parte Llamada (AVANZADO)

Escenario: Diferentes socios llaman a diferentes GTs y quieres un control detallado.

```
config :omniss7,
  gt_nat_enabled: true,

  gt_nat_rules: [
    # Socio A llamando a tu GT de SMS - mayor prioridad (peso 1)
    %{calling_prefix: "4412", called_prefix: "5551", weight: 1,
response_gt: "555101"},

    # Socio B llamando a tu GT de SMS - mayor prioridad (peso 1)
    %{calling_prefix: "4413", called_prefix: "5551", weight: 1,
response_gt: "555102"},

    # Cualquiera llamando a tu GT de SMS - prioridad media (peso
10)
    %{called_prefix: "5551", weight: 10, response_gt: "555100"},

    # Socio A llamando a cualquier GT - prioridad media (peso 10)
    %{calling_prefix: "4412", weight: 10, response_gt: "555200"},

    # Comodín por defecto - baja prioridad (peso 100)
    %{weight: 100, response_gt: "555000"}
  ]
```

Ejemplos de Coincidencia:

```
# Socio A llama al GT de SMS
Llamada: "441234567", Llamada: "555100"
→ Coincide con la regla de peso 1 (ambos prefijos) → "555101"

# Socio A llama al GT de Voz
Llamada: "441234567", Llamada: "555200"
→ Coincide con la regla de peso 10 (solo llamada) → "555200"

# Llamador desconocido llama al GT de SMS
Llamada: "999999999", Llamada: "555100"
→ Coincide con la regla de peso 10 (solo llamada) → "555100"

# Llamador desconocido llama al GT de Voz
Llamada: "999999999", Llamada: "555200"
→ Coincide con el comodín de peso 100 → "555000"
```

Modos Operativos

GT NAT funciona en todos los modos operativos de OmniSS7:

Modo HLR

GT NAT afecta:

- Respuestas de UpdateLocation (GT de HLR en respuesta)
- Mensajes de InsertSubscriberData (GT de HLR como parte llamante)
- Respuestas de SendAuthenticationInfo
- Respuestas de Cancel Location

Para más información sobre las operaciones de HLR, consulta la Guía de Configuración de HLR.

Configuración:

```
config :omniss7,
  hlr_mode_enabled: true,
  hlr_service_center_gt_address: "5551234567",  # GT de HLR por
defecto

  gt_nat_enabled: true,
  gt_nat_rules: [
    %{calling_prefix: "331", weight: 10, response_gt:
"5551234568"},  # Francia
    %{calling_prefix: "44", weight: 10, response_gt:
"5551234569"},   # Reino Unido
    %{weight: 100, response_gt: "5551234567"}  # Comodín por
defecto
  ]
```

Modo SMSc

GT NAT afecta:

- Respuestas SRI-for-SM (`networkNode-Number` campo) - consulta Detalles de SRI-for-SM
- Reconocimientos de MT-ForwardSM

Para más información sobre las operaciones de SMSc, consulta la Guía de Configuración de SMSc.

Configuración:

```
config :omniss7,
  smsc_mode_enabled: true,
  smsc_service_center_gt_address: "5559999",  # GT de SMSc por
defecto

  gt_nat_enabled: true,
  gt_nat_rules: [
    %{calling_prefix: "1", weight: 10, response_gt: "5559991"},
# América del Norte
    %{calling_prefix: "44", weight: 10, response_gt: "5559992"},
# Reino Unido
    %{calling_prefix: "86", weight: 10, response_gt: "5559993"},
# China
    %{weight: 100, response_gt: "5559999"}  # Comodín por defecto
  ]
```

Modo Puerta de Enlace CAMEL

GT NAT afecta:

- Todas las respuestas a nivel SCCP (GT de gsmSCF como parte llamante)
- Respuestas de operaciones CAMEL/CAP (InitialDP, EventReportBCSM, etc.)
- Reconocimientos de RequestReportBCSMEvent
- Respuestas de ApplyCharging
- Respuestas de Continue

Configuración:

```
config :omniss7,
  camelgw_mode_enabled: true,
  camel_gsmscf_gt_address: "55512341112",  # GT de gsmSCF por
defecto

  gt_nat_enabled: true,
  gt_nat_rules: [
    %{calling_prefix: "555", weight: 10, response_gt:
"55512341111"},  # Red A
    %{calling_prefix: "666", weight: 10, response_gt:
"55512311555"},  # Red B
    %{weight: 100, response_gt: "55512341112"}  # Comodín por
defecto
  ]
```

Caso de Uso: Cuando operas como un gsmSCF (Función de Control de Servicio) para múltiples redes, cada gsmSSF de la red puede esperar respuestas de un GT de gsmSCF específico. GT NAT asegura que se utilice el GT correcto según qué gsmSSF está llamando.

Registro y Depuración

Habilitar Registro de GT NAT

GT NAT incluye registro automático de todas las traducciones:

```
# En los registros, verás:
[info] GT NAT [respuesta SRI-for-SM]: GT Llamante 877234567 -> GT
de Respuesta 55512341112
[info] GT NAT [UpdateLocation ISD]: GT Llamante 331234567 -> GT de
Respuesta 55512341111
[info] GT NAT [respuesta MAP BEGIN]: GT Llamante 441234567 -> GT
de Respuesta 55512311555
```

El campo de contexto muestra dónde se aplicó el NAT:

- `"respuesta SRI-for-SM"` - En el controlador SRI-for-SM

- `"UpdateLocation ISD"` - En mensajes de InsertSubscriberData
- `"UpdateLocation END"` - En respuesta de UpdateLocation END
- `"respuesta MAP BEGIN"` - Respuestas MAP BEGIN genéricas
- `"ISD ACK"` - Reconocimiento de ISD
- `"respuesta de error HLR"` - Respuesta de error de HLR
- `"respuesta CAMEL"` - Respuestas de operaciones CAMEL/CAP (gsmSCF)

Validación

El sistema valida la configuración de GT NAT al inicio:

```
# Verificar configuración de GT NAT
iex> GtNat.validate_config()
{:ok, [
  %{calling_prefix: "8772", weight: 10, response_gt:
"55512341112"},
  %{calling_prefix: "8773", weight: 10, response_gt:
"55512341111"}
]}

# Verificar si está habilitado
iex> GtNat.enabled?()
true

# Obtener todas las reglas
iex> GtNat.get_rules()
[
  %{calling_prefix: "8772", weight: 10, response_gt:
"55512341112"},
  %{calling_prefix: "8773", weight: 10, response_gt:
"55512341111"}
]
```

Probar GT NAT

Prueba la lógica de GT NAT programáticamente:

```
# Probar traducción con solo GT de llamada (called_gt es nil)
iex> GtNat.translate_response_gt("877234567", nil, "default_gt")
"55512341112"

# Probar traducción con ambos GTs
iex> GtNat.translate_response_gt("877234567", "555123",
"default_gt")
"55512341112"

# Probar con registro (GT llamada nil)
iex> GtNat.translate_response_gt_with_logging("877234567", nil,
"default_gt", "test")
# Registros: GT NAT [test]: GT Llamante 877234567 -> GT de
Respuesta 55512341112
"55512341112"

# Probar con registro (ambos GTs)
iex> GtNat.translate_response_gt_with_logging("877234567",
"555123", "default_gt", "test")
# Registros: GT NAT [test]: GT Llamante 877234567, GT Llamada
555123 -> GT de Respuesta 55512341112
"55512341112"

# Probar sin coincidencia (devuelve por defecto)
iex> GtNat.translate_response_gt("999999999", "888888",
"default_gt")
"default_gt"
```

Solución de Problemas

Problema: GT NAT No Funciona

Verificación 1: ¿Está habilitado?

```
iex> Application.get_env(:omniss7, :gt_nat_enabled)
true  # Debería ser true
```

Verificación 2: ¿Están configuradas las reglas?

```
iex> Application.get_env(:omniss7, :gt_nat_rules)
[%{calling_prefix: "8772", response_gt: "55512341112"}, ...]  #
Debería devolver lista
```

Verificación 3: Verificar registros Busca "GT NAT" en los registros para ver si se están realizando traducciones.

Problema: GT Incorrecto en Respuestas

Síntoma: Las respuestas utilizan una dirección GT inesperada

Causa: La coincidencia de prefijo de regla podría ser demasiado amplia o la regla por defecto está capturando tráfico

Solución: Revisa los pesos y prefijos de las reglas:

```
# MALO: Comodín con bajo peso (captura todo primero)
gt_nat_rules: [
  %{weight: 1, response_gt: "111111"},              # ¡Esto
coincide con todo primero!
  %{calling_prefix: "8772", weight: 10, response_gt: "222222"}  #
Nunca alcanzado
]

# BUENO: Reglas específicas con peso más bajo, comodín con peso
más alto
gt_nat_rules: [
  %{calling_prefix: "8772", weight: 10, response_gt: "222222"},  #
Específico, bajo peso
  %{weight: 100, response_gt: "111111"}  # Comodín, alto peso (por
defecto)
]
```

Problema: GT NAT No Aplicado a Tipo de Mensaje Específico

Síntoma: Algunas respuestas utilizan GT NAT'd, otras no

Cobertura Actual:

- ✅ GT Llamante SCCP (todas las respuestas)
- ✅ Respuestas SRI-for-SM (networkNode-Number)
- ✅ Mensajes ISD de UpdateLocation (GT de HLR)
- ✅ Respuestas de UpdateLocation END
- ✅ Reconocimientos de ISD
- ✅ Respuestas MAP BEGIN

Si un tipo de mensaje específico no está utilizando GT NAT, puede que aún no esté implementado. Revisa el código fuente o contacta con soporte.

Consideraciones de Rendimiento

Rendimiento de Búsqueda

GT NAT utiliza coincidencia de prefijo simple con complejidad O(n) donde n es el número de reglas.

Consejos de rendimiento:

- Mantén el conteo de reglas por debajo de 100 para un mejor rendimiento
- Utiliza prefijos específicos para reducir el conteo de reglas
- La regla por defecto (prefijo vacío) debe ser la última

Prueba de rendimiento (sistema típico):

- 10 reglas: < 1µs por búsqueda
- 50 reglas: < 5µs por búsqueda
- 100 reglas: < 10µs por búsqueda

Uso de Memoria

Cada regla requiere ~100 bytes de memoria:

- 10 reglas ≈ 1 KB

- 100 reglas ≈ 10 KB

Mejores Prácticas

1. Siempre Incluir una Regla de Comodín por Defecto

```
gt_nat_rules: [
  %{calling_prefix: "8772", weight: 10, response_gt: "111111"},
  %{calling_prefix: "8773", weight: 10, response_gt: "222222"},
  %{weight: 100, response_gt: "default_gt"}  # Siempre tener un
comodín con alto peso
]
```

2. Utilizar Prefijos y Pesos Significativos

```
# BUENO: Prefijos claros y específicos con pesos apropiados
%{calling_prefix: "331", weight: 10, response_gt: "..."}  #
Francia
%{calling_prefix: "44", weight: 10, response_gt: "..."}   # Reino
Unido

# MALO: Prefijos demasiado amplios o pesos confusos
%{calling_prefix: "3", weight: 5, response_gt: "..."}     #
Demasiados países
%{calling_prefix: "331", weight: 100, response_gt: "..."}  # El
peso debería ser más bajo para reglas específicas
```

3. Documentar Tus Reglas

```
gt_nat_rules: [
  # Socio XYZ - red del Reino Unido (rango de GT: 4412xxxxxxx)
  # Peso 10: Prioridad estándar del socio
  %{calling_prefix: "4412", weight: 10, response_gt: "5551001"},

  # Socio ABC - red de Francia (rango de GT: 33123xxxxxx)
  # Peso 10: Prioridad estándar del socio
  %{calling_prefix: "33123", weight: 10, response_gt: "5551002"}
]
```

4. Probar Antes de Desplegar

```
# Probar en iex antes de desplegar
iex> GtNat.translate_response_gt("44121234567", nil, "default")
"5551001"  # Resultado esperado

# Probar con GT llamado
iex> GtNat.translate_response_gt("44121234567", "555123",
"default")
"5551001"  # Resultado esperado
```

5. Monitorear Registros

Habilita el registro de nivel INFO para ver todas las traducciones de GT NAT en producción.

Integración con Otras Funciones

Modo STP

GT NAT funciona independientemente del enrutamiento STP. STP enruta en función de códigos de punto y GTs de destino, mientras que GT NAT maneja la dirección de respuesta.

Para más información sobre el enrutamiento STP, consulta la Guía de Configuración de STP.

Integración CAMEL

GT NAT está **totalmente integrado** con operaciones CAMEL/CAP:

Capa SCCP:

- GT de Parte Llamante en todas las respuestas CAMEL
- Aplicado automáticamente según el GT de gsmSSF entrante

Configuración:

- `camel_gsmscf_gt_address` - GT de gsmSCF por defecto (opcional)
- Si no está configurado, utiliza el GT de Parte Llamada de la solicitud entrante
- Las reglas de GT NAT anulan el valor por defecto según el prefijo de parte llamante

Ejemplo:

```
# Cuando gsmSSF 555123456 llama a tu gsmSCF
# Entrante: Llamada=55512341112, Llamante=555123456
# Búsqueda de GT NAT: "555" -> response_gt="55512341111"
# Respuesta: Llamada=555123456, Llamante=55512341111
```

Balanceo de Carga

GT NAT puede combinarse con balanceo de carga M3UA para una gestión avanzada del tráfico.

Guía de Migración

Habilitar GT NAT en un Sistema Existente

1. **Preparar Configuración**

```
# Agregar a runtime.exs (mantener deshabilitado inicialmente)
config :omniss7,
  gt_nat_enabled: false,  # Comenzar deshabilitado
  gt_nat_rules: [
    # Tus reglas aquí con pesos
    %{calling_prefix: "877", weight: 10, response_gt:
"111111"},
    %{weight: 100, response_gt: "999999"}  # Comodín por
defecto
  ]
```

2. **Probar Configuración**

```
# Validar que la configuración compile
mix compile

# Probar en iex
iex -S mix
iex> GtNat.validate_config()
```

3. **Habilitar en Staging**

```
gt_nat_enabled: true  # Cambiar a true
```

4. **Monitorear Registros**

```
tail -f log/omniss7.log | grep "GT NAT"
```

5. **Desplegar en Producción**

    - Desplegar durante la ventana de mantenimiento
    - Monitorear las primeras 24 horas de cerca
    - Tener un plan de reversión listo (establecer `gt_nat_enabled: false`)

Soporte

Para problemas o preguntas:

- Verifica los registros para mensajes "GT NAT"
- Valida la configuración con `GtNat.validate_config()`
- Revisa la sección de solución de problemas de esta guía
- Contacta con el soporte de OmniSS7 con extractos de registros

Guía de Configuración de HLR

← Volver a la Documentación Principal

Esta guía proporciona la configuración para usar OmniSS7 como un **Registro de Ubicación del Hogar (HLR/HSS)** con **OmniHSS** como la base de datos de suscriptores de backend.

Integración de OmniHSS

El modo HLR de OmniSS7 funciona como un frontend de señalización SS7 que se conecta con **OmniHSS**, un servidor de suscriptores del hogar (HSS) de características completas. Esta arquitectura separa las preocupaciones:

- **OmniSS7 (Frontend HLR)**: Maneja toda la señalización del protocolo SS7/MAP, el enrutamiento SCCP y la comunicación de red
- **OmniHSS (Backend HSS)**: Gestiona los datos de suscriptores, la autenticación, la provisión y características avanzadas

¿Por qué OmniHSS?

OmniHSS proporciona gestión de suscriptores de grado de operador con características que incluyen:

- **Soporte Multi-IMSI**: Cada suscriptor puede tener múltiples IMSIs asociadas con un solo MSISDN para roaming internacional, cambio de red y provisión de eSIM
- **Autenticación Flexible**: Soporte para algoritmos de autenticación tanto Milenage (3G/4G/5G) como COMP128 (2G)
- **Seguimiento de Sesiones de Circuito y Paquete**: Seguimiento independiente de registros de red CS (circuito conmutado) y PS (paquete conmutado)

- **Provisionamiento Avanzado**: Perfiles de servicio personalizables, servicios suplementarios y datos de suscripción CAMEL
- **Diseño API-Primero**: API HTTP RESTful para integración con sistemas de facturación, CRM y provisión
- **Actualizaciones en Tiempo Real**: Seguimiento de ubicación, gestión de sesiones y generación de vectores de autenticación

Todos los datos de suscriptores, credenciales de autenticación y configuraciones de servicio se almacenan y gestionan en OmniHSS. OmniSS7 consulta a OmniHSS a través de llamadas API HTTPS para responder a operaciones MAP como UpdateLocation, SendAuthenticationInfo y SendRoutingInfo.

Importante: El modo HLR de OmniSS7 es un **frontend de señalización solamente**. Toda la lógica de gestión de suscriptores, algoritmos de autenticación, reglas de provisión y operaciones de base de datos son manejadas por OmniHSS. Esta guía cubre la configuración del protocolo SS7/MAP en OmniSS7. Para información sobre provisión de suscriptores, configuración de autenticación, perfiles de servicio y operaciones administrativas, **consulte la documentación de OmniHSS**.

Soporte Multi-IMSI

OmniHSS admite de forma nativa configuraciones Multi-IMSI, permitiendo que un solo suscriptor (identificado por MSISDN) tenga múltiples IMSIs. Esto permite:

- **Perfiles de Roaming Internacional**: Diferentes IMSIs para diferentes regiones para reducir costos de roaming
- **eSIM Multi-Perfil**: Múltiples perfiles de red en un solo dispositivo compatible con eSIM
- **Cambio de Red**: Cambio sin problemas entre redes sin cambiar MSISDN
- **Coordinación de Doble SIM**: Coordinación entre múltiples SIM físicas o virtuales
- **Pruebas y Desarrollo**: Múltiples IMSIs de prueba apuntando al mismo suscriptor

Cómo funciona:

- Cada IMSI tiene sus propias credenciales de autenticación (Ki, OPc, algoritmo)
- Cada IMSI puede tener registros de sesión de circuito y paquete independientes
- Los servicios y perfiles de suscriptores pueden ser compartidos o personalizados por IMSI
- OmniSS7 consulta a OmniHSS por IMSI, y OmniHSS devuelve los datos de suscriptor apropiados
- Los sistemas de facturación pueden rastrear el uso por IMSI mientras asocian todas las IMSIs a una sola cuenta

Ejemplo de escenario Multi-IMSI:

```
Suscriptor MSISDN: +1-555-123-4567
├─ IMSI 1: 310260123456789 (Red Nacional de EE. UU. -
autenticación Milenage)
├─ IMSI 2: 208011234567890 (Perfil de Roaming en Francia -
autenticación Milenage)
└─ IMSI 3: 440201234567891 (Perfil de Roaming en el Reino Unido -
autenticación COMP128)
```

Las tres IMSIs pueden ser utilizadas de forma independiente para el registro en la red, pero todas pertenecen a la misma cuenta de suscriptor. OmniHSS gestiona el mapeo de IMSI a suscriptor y asegura la autenticación y provisión adecuadas para cada IMSI.

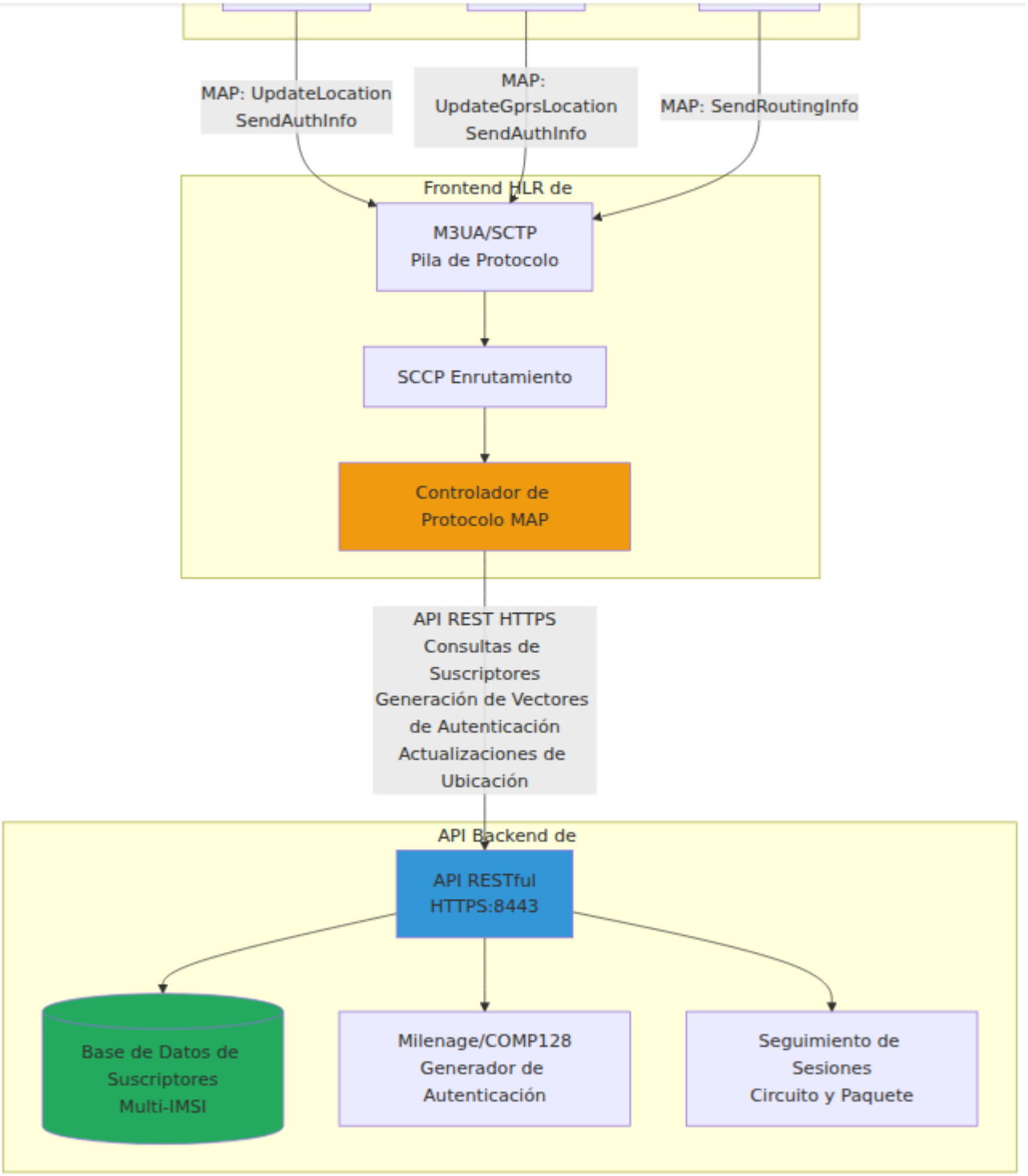
Tabla de Contenidos

¿Qué es el Modo HLR?

Modo HLR permite que OmniSS7 funcione como un Registro de Ubicación del Hogar para:

- **Gestión de Suscriptores**: Almacenar y gestionar datos de suscriptores
- **Autenticación**: Generar vectores de autenticación para acceso a la red
- **Seguimiento de Ubicación**: Procesar actualizaciones de ubicación de los VLRs
- **Información de Enrutamiento**: Proporcionar información de enrutamiento para llamadas y SMS

Arquitectura HLR



Habilitando el Modo HLR

OmniSS7 puede operar en diferentes modos. Para usarlo como un HLR, necesitas habilitar el modo HLR en la configuración.

Cambio a Modo HLR

El `config/runtime.exs` de OmniSS7 contiene tres modos operativos preconfigurados. Para habilitar el modo HLR:

1. **Abrir** `config/runtime.exs`
2. **Encontrar** las tres secciones de configuración (líneas 53-174):
   - Configuración 1: Modo STP (líneas 53-85)
   - Configuración 2: Modo HLR (líneas 87-123)
   - Configuración 3: Modo SMSc (líneas 125-174)
3. **Comentar** la configuración actualmente activa (agregar `#` a cada línea)
4. **Descomentar** la configuración HLR (eliminar `#` de las líneas 87-123)
5. **Personalizar** los parámetros de configuración según sea necesario
6. **Reiniciar** la aplicación: `iex -S mix`

Configuración del Modo HLR

La configuración completa del HLR se ve así:

```
config :omniss7,
  # Banderas de modo - Habilitar características HLR solamente
  map_client_enabled: true,
  hlr_mode_enabled: true,
  smsc_mode_enabled: false,

  # Configuración de la API Backend de OmniHSS
  hlr_api_base_url: "https://10.180.2.140:8443",

  # Dirección GT del Centro de Servicio HLR para operaciones SMS
  hlr_service_center_gt_address: "1234567890",

  # Configuración de Mapeo MSISDN ↔ IMSI
  # Ver: sección de Mapeo MSISDN ↔ IMSI para detalles
  hlr_imsi_plmn_prefix: "50557",
  hlr_msisdn_country_code: "61",
  hlr_msisdn_nsn_offset: 0,
  hlr_msisdn_nsn_length: 9,

  # Configuración de InsertSubscriberData
  # Modo de Acceso a la Red: :packetAndCircuit, :packetOnly, o
:circuitOnly
  isd_network_access_mode: :packetAndCircuit,

  # Enviar ISD #2 (datos de Servicios Suplementarios)
  isd_send_ss_data: true,

  # Enviar ISD #3 (datos de Barring de Llamadas)
  isd_send_call_barring: true,

  # Configuración CAMEL (para respuestas SendRoutingInfo)
  # Clave de Servicio para la iniciación de servicios CAMEL
  camel_service_key: 11_110,

  # Punto de Detección de Disparador CAMEL
  # Opciones: :termAttemptAuthorized, :tBusy, :tNoAnswer, :tAnswer
  camel_trigger_detection_point: :termAttemptAuthorized,

  # Prefijos de VLR de Hogar
  # Lista de prefijos de dirección VLR que se consideran "red" de
hogar
  # Si el VLR del suscriptor comienza con uno de estos prefijos,
usar respuesta SRI estándar
```

```
  # De lo contrario, el suscriptor está en roaming y necesitamos
enviar PRN para obtener MSRN
  home_vlr_prefixes: ["123456"],

  # Configuración de Conexión M3UA
  # Conectar como ASP para recibir operaciones MAP
(UpdateLocation, SendAuthInfo, etc.)
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :hlr_client_asp,
    # Punto final local (sistema HLR)
    local_ip: {10, 179, 4, 11},
    local_port: 2905,
    # Punto final remoto STP
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1
  }
```

Parámetros de Configuración a Personalizar

Para una referencia completa de todos los parámetros de configuración, consulte la Referencia de Configuración.

Parámetro	Tipo	Predeterminado
`hlr_api_base_url`	Cadena	*Requerido*
`hlr_service_center_gt_address`	Cadena	*Requerido*
`smsc_service_center_gt_address`	Cadena	*Requerido*
`hlr_smsc_alert_gts`	Lista	`[]`
`hlr_alert_location_expiry_seconds`	Entero	`172800`
`hlr_imsi_plmn_prefix`	Cadena	`"50557"`
`hlr_msisdn_country_code`	Cadena	`"61"`

Parámetro	Tipo	Predeterminado
`hlr_msisdn_nsn_offset`	Entero	`0`
`hlr_msisdn_nsn_length`	Entero	`9`
`isd_network_access_mode`	Átomo	`:packetAndCircuit`
`isd_send_ss_data`	Booleano	`true`
`isd_send_call_barring`	Booleano	`true`
`camel_service_key`	Entero	`11_110`
`camel_trigger_detection_point`	Átomo	`:termAttemptAuthorized`

Parámetro	Tipo	Predeterminado
`home_vlr_prefixes`	Lista	`["5551231"]`
`local_ip`	Tupla	*Requerido*
`local_port`	Entero	`2905`
`remote_ip`	Tupla	*Requerido*
`remote_port`	Entero	`2905`
`routing_context`	Entero	`1`

Qué Sucede Cuando se Habilita el Modo HLR

Cuando `hlr_mode_enabled: true`, la interfaz web mostrará:

- **Eventos SS7** - Registro de eventos
- **Cliente SS7** - Pruebas de operaciones MAP
- **M3UA** - Estado de conexión
- **Enlaces HLR** - Estado de la API HLR + gestión de suscriptores ← *Específico de HLR*
- **Recursos** - Monitoreo del sistema
- **Configuración** - Visor de configuración

Las pestañas **Enrutamiento**, **Prueba de Enrutamiento** y **Enlaces SMSc** estarán ocultas.

Notas Importantes

- **Configuración Requerida**: El parámetro `hlr_service_center_gt_address` es **obligatorio**. La aplicación no podrá iniciarse si no está configurado.
- **Backend de OmniHSS**: La API de OmniHSS debe ser accesible en la `hlr_api_base_url` configurada
- **Tiempo de Espera de Solicitud API**: Todas las solicitudes API de OmniHSS tienen un **tiempo de espera codificado de 5 segundos**
- **Tiempo de Espera de Solicitud MAP**: Todas las solicitudes MAP (SRI, UpdateLocation, SendAuthInfo, etc.) tienen un **tiempo de espera codificado de 10 segundos**
- **Tiempo de Espera ISD**: Cada mensaje InsertSubscriberData (ISD) en una secuencia UpdateLocation tiene un **tiempo de espera codificado de 10 segundos**
- Se requiere conexión M3UA al STP para recibir operaciones MAP
- Después de cambiar de modo, debes reiniciar la aplicación para que los cambios surtan efecto
- **Interfaz Web**: Consulte la Guía de Interfaz Web para información sobre el uso de la interfaz web
- **Acceso API**: Consulte la Guía API para documentación de la API REST y acceso a Swagger UI

Base de Datos de Suscriptores

OmniHSS gestiona todos los datos de suscriptores incluyendo identidades, credenciales de autenticación, perfiles de servicio y información de ubicación. OmniSS7 recupera estos datos a través de llamadas API RESTful.

Modelo de Suscriptor de OmniHSS

OmniHSS almacena información completa del suscriptor:

- **Múltiples IMSIs por suscriptor**: Soporte para configuraciones Multi-IMSI (eSIM, perfiles de roaming, cambio de red)

- **Credenciales de autenticación**: Selección de Ki, OPc y algoritmo (Milenage o COMP128)
- **Perfiles de servicio**: Categoría de suscriptor, servicios permitidos, parámetros de QoS
- **Seguimiento de ubicación**: Seguimiento independiente de VLR/MSC actuales (sesión de circuito) y SGSN/GGSN (sesión de paquete)
- **Datos de suscripción CAMEL**: Claves de servicio, puntos de activación y direcciones gsmSCF
- **Servicios suplementarios**: Desvío de llamadas, barring, espera, configuraciones CLIP/CLIR
- **Estado administrativo**: Habilitado/deshabilitado, restricciones de servicio, fechas de expiración

---

Vectores de Autenticación

Generar Vectores de Autenticación

OmniHSS genera vectores de autenticación utilizando los algoritmos Milenage o COMP128 basados en el método de autenticación configurado de cada suscriptor. Cuando OmniSS7 recibe solicitudes MAP de **sendAuthenticationInfo**:

1. OmniSS7 extrae el IMSI de la solicitud MAP
2. OmniSS7 llama a la API de OmniHSS para generar vectores de autenticación
3. OmniHSS recupera las credenciales Ki y OPc del suscriptor
4. OmniHSS genera el número solicitado de vectores (RAND, XRES, CK, IK, AUTN)
5. OmniSS7 codifica los vectores en formato MAP y los devuelve al VLR/SGSN solicitante

Integración de la API de OmniHSS

OmniSS7 se comunica con OmniHSS a través de la API REST HTTPS para recuperar información de suscriptores, actualizar datos de ubicación y generar vectores de autenticación:

```
config :omniss7,
  hlr_api_base_url: "https://omnihss-server:8443"
```

Cuando OmniSS7 recibe operaciones MAP de la red SS7, consulta a OmniHSS para:

- **Recuperar datos de suscriptor** por IMSI o MSISDN
- **Generar vectores de autenticación** utilizando credenciales Ki/OPc almacenadas
- **Actualizar ubicación de sesión de circuito** cuando los suscriptores realizan UpdateLocation
- **Verificar estado de suscriptor** y derechos de servicio

Actualizaciones de Ubicación

Procesamiento de Actualización de Ubicación

Al recibir solicitudes MAP de **updateLocation**, OmniSS7 coordina con OmniHSS para registrar al suscriptor en un nuevo VLR:

1. **Extraer información de ubicación** de la solicitud UpdateLocation (IMSI, nuevo GT de VLR, nuevo GT de MSC)
2. **Consultar a OmniHSS** para verificar que el suscriptor existe y está habilitado
3. **Actualizar sesión de circuito** en OmniHSS con nueva ubicación VLR/MSC
4. **Enviar mensajes InsertSubscriberData (ISD)** para provisionar al suscriptor en el nuevo VLR

5. **Devolver respuesta UpdateLocation** al VLR (incluye GT HLR de `hlr_service_center_gt_address`)
6. **Enviar alertServiceCenter** a los GTs SMSc configurados (si `hlr_smsc_alert_gts` está poblado)

Nota: El parámetro de configuración `hlr_service_center_gt_address` especifica el Título Global del HLR que se devuelve en las respuestas UpdateLocation. Esto permite que el VLR/MSC identifique y enrute mensajes de vuelta a este HLR.

Integración del Centro de Servicio de Alerta

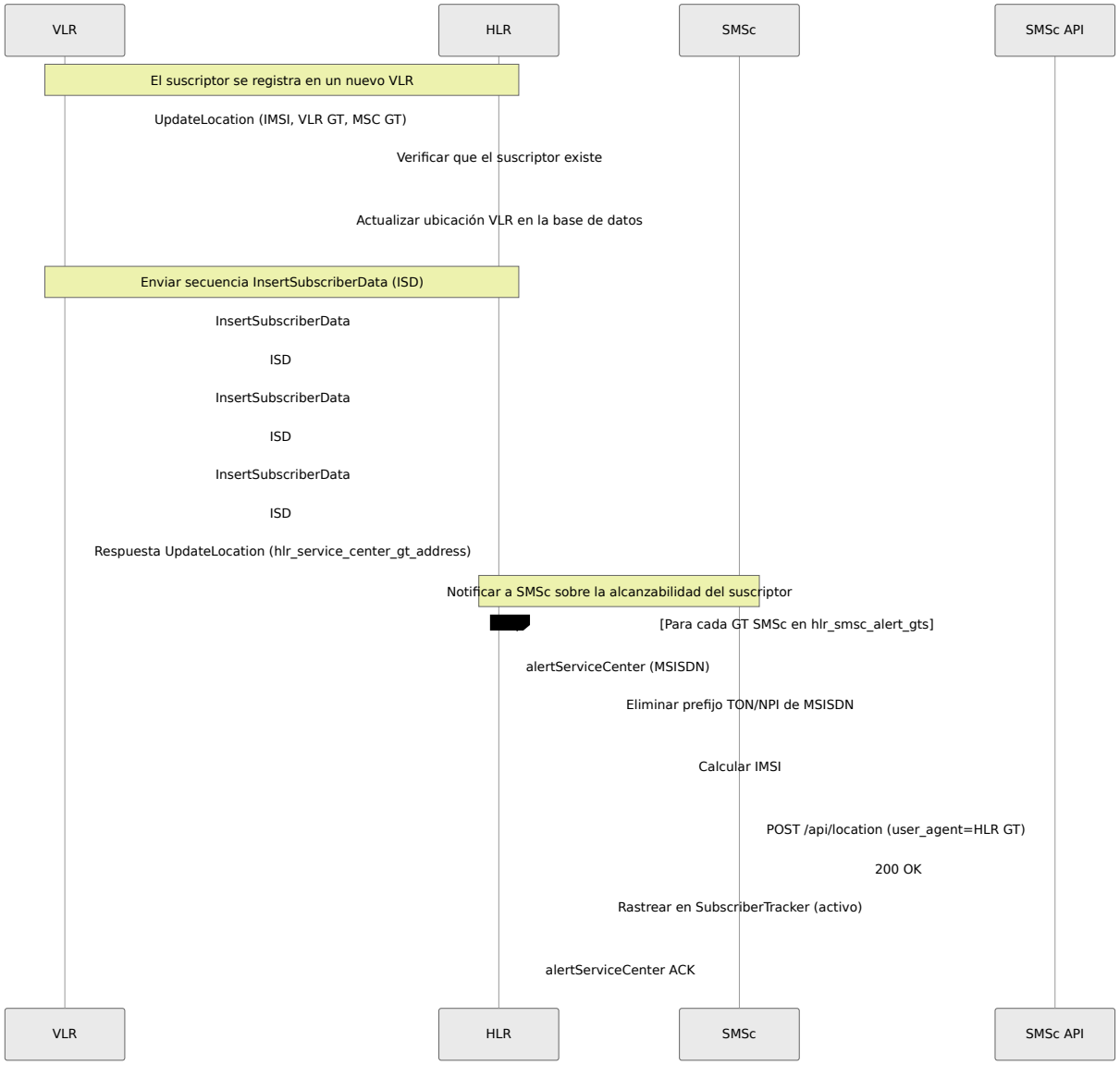
Después de una actualización de ubicación exitosa, el HLR puede notificar automáticamente a los sistemas SMSc que un suscriptor ahora es alcanzable enviando mensajes **alertServiceCenter** (código de operación MAP 64). Para información sobre cómo el SMSc maneja estas alertas, consulte Manejo del Centro de Servicio de Alerta en la Guía SMSc.

Configuración

Configure la lista de Títulos Globales SMSc a notificar:

```
config :omniss7,
  # Lista de GTs SMSc para enviar alertServiceCenter después de
UpdateLocation
  hlr_smsc_alert_gts: [
    "15559876543",
    "15559876544"
  ],

  # Tiempo de expiración de ubicación cuando SMSc recibe
alertServiceCenter (predeterminado: 48 horas)
  hlr_alert_location_expiry_seconds: 172800
```

Diagrama de Flujo



Comportamiento

Cuando un suscriptor realiza UpdateLocation:

1. HLR envía alertServiceCenter a **cada** GT SMSc en la lista `hlr_smsc_alert_gts`
2. El mensaje incluye el MSISDN del suscriptor
3. HLR utiliza `hlr_service_center_gt_address` como el GT de la parte que llama
4. Direccionamiento SCCP: SSN de llamada=6 (HLR), SSN llamado=8 (SMSc)

El SMSc recibe la alerta y:

- **Elimina el prefijo TON/NPI** del MSISDN (por ejemplo, "19123123213" → "123123213")
- Marca al suscriptor como alcanzable en su base de datos de ubicación (a través de POST a /api/location)
- **Establece el campo** `user_agent` al GT HLR al llamar a la API (para rastrear qué HLR envió la alerta)
- Establece el tiempo de expiración de ubicación basado en `hlr_alert_location_expiry_seconds`
- Rastrea al suscriptor en el Rastreador de Suscriptores del SMSc para monitoreo

Pruebas

Utilice la página **Suscriptores Activos** en la interfaz web para enviar manualmente mensajes alertServiceCenter para pruebas:

1. Navegue a la pestaña "Suscriptores Activos"
2. Encuentre la sección "Probar Centro de Servicio de Alerta"
3. Ingrese MSISDN, GT SMSc y GT HLR (los valores predeterminados están pre-poblados desde la configuración)
   - GT SMSc predeterminado es la primera entrada en `hlr_smsc_alert_gts`
   - GT HLR predeterminado es `hlr_service_center_gt_address`
4. Haga clic en "Enviar alertServiceCenter"

Esto es útil para probar el manejo de alertas del SMSc sin requerir un flujo completo de UpdateLocation. El formulario utiliza validación `phx-blur` para evitar mostrar errores mientras se escribe.

Configuración de InsertSubscriberData (ISD)

Después de una actualización de ubicación exitosa, el HLR envía datos de provisión de suscriptor al VLR utilizando mensajes **InsertSubscriberData** (ISD). La configuración de ISD permite personalizar qué datos se envían y cómo.

Para referencia de parámetros de configuración, consulte Configuración de ISD en la Referencia de Configuración.

Secuencia ISD

El HLR puede enviar hasta 3 mensajes ISD secuenciales:

1. **ISD #1** (Siempre enviado) - Datos básicos del suscriptor:
    - IMSI
    - MSISDN
    - Categoría de suscriptor
    - Estado del suscriptor (serviceGranted)
    - Lista de servicios de portadora
    - Lista de teleservicios
    - Modo de acceso a la red
2. **ISD #2** (Opcional) - Datos de Servicios Suplementarios (SS):
    - Configuraciones de desvío de llamadas (incondicional, ocupado, sin respuesta, no alcanzable)
    - Espera de llamadas
    - Retención de llamadas
    - Servicio de múltiples partes
    - Estado y características del servicio suplementario
3. **ISD #3** (Opcional) - Datos de Barring de Llamadas:
    - Barring de todas las llamadas salientes (BAOC)
    - Barring de llamadas internacionales salientes (BOIC)
    - Datos de restricción de acceso

Opciones de Configuración

```
# Configuración de InsertSubscriberData
# Modo de Acceso a la Red: :packetAndCircuit, :packetOnly, o
:circuitOnly
isd_network_access_mode: :packetAndCircuit,

# Enviar ISD #2 (datos de Servicios Suplementarios)
isd_send_ss_data: true,

# Enviar ISD #3 (datos de Barring de Llamadas)
isd_send_call_barring: true,
```

Modo de Acceso a la Red

El parámetro `isd_network_access_mode` controla qué tipo de acceso a la red se permite al suscriptor:

Valor	Descripción	Caso de Uso
`:packetAndCircuit`	Tanto conmutado por paquete (GPRS/LTE) como conmutado por circuito (voz)	Predeterminado - Suscriptores de servicio completo
`:packetOnly`	Solo conmutado por paquete (datos/LTE)	Tarjetas SIM solo de datos, dispositivos IoT
`:circuitOnly`	Solo conmutado por circuito (voz/SMS)	Dispositivos heredados, planes solo de voz

Controlando Mensajes ISD

Puedes controlar qué mensajes ISD se envían según tus requisitos de red:

Enviar todos los ISDs (Predeterminado - Conjunto completo de características):

```
isd_send_ss_data: true,
isd_send_call_barring: true,
```

Enviar solo datos básicos del suscriptor (Provisionamiento mínimo):

```
isd_send_ss_data: false,
isd_send_call_barring: false,
```

Enviar básico + servicios suplementarios (Sin barring de llamadas):

```
isd_send_ss_data: true,
isd_send_call_barring: false,
```

Ejemplo de Flujo ISD

Cuando se recibe UpdateLocation:

```
VLR → HLR: UpdateLocation (INICIO)
HLR → VLR: InsertSubscriberData #1 (CONTINUAR) - Datos básicos
VLR → HLR: ISD #1 ACK (CONTINUAR)
HLR → VLR: InsertSubscriberData #2 (CONTINUAR) - Datos SS [si está
habilitado]
VLR → HLR: ISD #2 ACK (CONTINUAR)
HLR → VLR: InsertSubscriberData #3 (CONTINUAR) - Barring de
llamadas [si está habilitado]
VLR → HLR: ISD #3 ACK (CONTINUAR)
HLR → VLR: Respuesta UpdateLocation (FIN)
```

Si `isd_send_ss_data` o `isd_send_call_barring` están configurados como `false`, esos mensajes ISD se omiten, y el UpdateLocation FIN se envía antes.

Mejores Prácticas

- **Configuración Predeterminada**: Utilice `:packetAndCircuit` y habilite todos los ISDs para máxima compatibilidad
- **IoT/M2M**: Utilice `:packetOnly` y desactive datos SS/barring de llamadas para dispositivos solo de datos

- **Interoperabilidad**: Algunos VLRs más antiguos pueden no soportar todos los servicios suplementarios - desactive `isd_send_ss_data` si encuentra problemas
- **Rendimiento**: Desactivar ISDs no utilizados reduce la sobrecarga de mensajes y acelera las actualizaciones de ubicación

---

Integración CAMEL

Configuración CAMEL para SendRoutingInfo

Al responder a solicitudes de **SendRoutingInfo** (SRI) de un GMSC (Gateway MSC), el HLR puede instruir al GMSC para invocar servicios CAMEL para enrutamiento inteligente de llamadas y control de servicios.

Para referencia de parámetros de configuración, consulte Configuración CAMEL en la Referencia de Configuración.

¿Qué es CAMEL?

CAMEL (Aplicaciones Personalizadas para Lógica Mejorada de Redes Móviles) es un protocolo que permite servicios de red inteligente en redes GSM/UMTS. Permite a los operadores de red implementar servicios de valor agregado como:

- Facturación prepago
- Filtrado y barring de llamadas
- Redes Privadas Virtuales (VPN)
- Servicios de tarifa premium
- Desvío de llamadas con lógica personalizada
- Servicios basados en ubicación

Opciones de Configuración

```
# Configuración CAMEL (para respuestas SendRoutingInfo)
# Clave de Servicio para la iniciación de servicios CAMEL
camel_service_key: 11_110,

# Punto de Detección de Disparador CAMEL
# Opciones: :termAttemptAuthorized, :tBusy, :tNoAnswer, :tAnswer
camel_trigger_detection_point: :termAttemptAuthorized,
```

Clave de Servicio

La `camel_service_key` identifica qué servicio CAMEL debe ser invocado en el gsmSCF (Función de Control de Servicio). Este es un identificador numérico configurado en su red:

Clave de Servicio	Caso de Uso Típico
`11_110`	Control de llamadas terminadas prepago (predeterminado)
`100`	Servicio prepago de origen
`200`	Desvío de llamadas con lógica personalizada
`300`	Red Privada Virtual (VPN)
Personalizado	Servicios específicos del operador

Ejemplo de Configuración:

```
# Para control de llamadas terminadas prepago
camel_service_key: 11_110,

# Para servicio VPN
camel_service_key: 300,
```

Punto de Detección de Disparador

El `camel_trigger_detection_point` especifica cuándo debe activarse el servicio CAMEL durante el establecimiento de la llamada:

Punto de Detección	Descripción	Cuándo se Activa
`:termAttemptAuthorized`	Intento de llamada autorizado (predeterminado)	Antes de que la llamada sea enrutada al suscriptor
`:tBusy`	Ocupado terminando	Cuando el suscriptor está ocupado
`:tNoAnswer`	Sin respuesta terminando	Cuando el suscriptor no responde
`:tAnswer`	Respuesta terminando	Cuando el suscriptor responde la llamada

Ejemplos de Configuración:

Control prepago estándar (activar antes del enrutamiento):

```
camel_trigger_detection_point: :termAttemptAuthorized,
```

Manejo personalizado de ocupado (activar cuando está ocupado):

```
camel_trigger_detection_point: :tBusy,
```

Facturación basada en respuesta (activar al responder):

```
camel_trigger_detection_point: :tAnswer,
```

Respuesta SRI con CAMEL

Cuando está configurado, las respuestas SendRoutingInfo incluyen información de suscripción CAMEL:

```
GMSC → HLR: SendRoutingInfo (INICIO)
HLR → GMSC: Respuesta SRI (FIN) con:
  - IMSI
  - Número VLR
  - Estado del suscriptor
  - Información de enrutamiento CAMEL:
    * Clave de Servicio: 11_110
    * Dirección gsmSCF: <dirección configurada>
    * Punto de Detección de Disparador: termAttemptAuthorized
    * Manejo de Llamadas por Defecto: continueCall

GMSC contacta a gsmSCF en el punto de disparo para ejecutar el
servicio CAMEL
```

Mejores Prácticas

- **Redes de Producción**: Utilice claves de servicio estandarizadas acordadas con su proveedor de gsmSCF
- **Pruebas**: Utilice `:termAttemptAuthorized` para pruebas más completas
- **Servicios Prepagos**: La clave de servicio `11_110` es un estándar común de la industria para llamadas terminadas prepago
- **Manejo de Respaldo**: `defaultCallHandling: :continueCall` asegura que las llamadas continúen si el gsmSCF no está disponible

---

Manejo de Suscriptores en Roaming

Detección de VLR de Hogar vs VLR en Roaming

Cuando el HLR recibe una solicitud de **SendRoutingInfo** (SRI), necesita determinar si el suscriptor está en un VLR "de hogar" (dentro de su red) o en

un VLR en roaming (visitando otra red). El comportamiento difiere según esta determinación:

Para referencia de parámetros de configuración, consulte Prefijos de VLR de Hogar en la Referencia de Configuración.

- **VLR de Hogar**: Devolver respuesta SRI estándar con información de enrutamiento CAMEL
- **VLR en Roaming**: Enviar una solicitud de Proveer Número en Roaming (PRN) para obtener un MSRN, luego devolverlo en la respuesta SRI

Configuración

```
# Prefijos de VLR de Hogar
# Lista de prefijos de dirección VLR que se consideran "red" de
hogar
# Si la dirección VLR del suscriptor comienza con uno de estos
prefijos, usar respuesta SRI estándar
# De lo contrario, el suscriptor está en roaming y necesitamos
enviar PRN para obtener MSRN
home_vlr_prefixes: ["555123"],
```

Ejemplo de Configuración:

```
# Un solo operador de red de hogar
home_vlr_prefixes: ["555123"],

# Múltiples operadores de red de hogar (por ejemplo, diferentes
regiones o subsidiarias)
home_vlr_prefixes: ["555123", "555124", "555125"],
```

Cómo Funciona

1. Flujo de Suscriptor de Hogar (Estándar)

Cuando la dirección VLR del suscriptor comienza con un prefijo de hogar configurado:

```
GMSC → HLR: SendRoutingInfo (MSISDN: "1234567890")
HLR consulta la API backend para datos del suscriptor
HLR verifica la dirección VLR: "5551234567"
HLR determina: VLR comienza con "555123" → Red de hogar
HLR → GMSC: Respuesta SRI con información de enrutamiento CAMEL:
  - IMSI
  - Número VLR: "5551234567"
  - Dirección gsmSCF (MSC): "5551234501"
  - Clave de servicio CAMEL: 11_110
  - Punto de detección de disparador: termAttemptAuthorized
```

2. Flujo de Suscriptor en Roaming (PRN Requerido)

Cuando la dirección VLR del suscriptor NO coincide con ningún prefijo de hogar:

```
GMSC → HLR: SendRoutingInfo (MSISDN: "1234567890")
HLR consulta la API backend para datos del suscriptor
HLR verifica la dirección VLR: "49170123456"
HLR determina: VLR no comienza con "555123" → Roaming
HLR → MSC: ProvideRoamingNumber (PRN):
  - MSISDN: "1234567890"
  - IMSI: "999999876543210"
  - Número MSC: "49170123456"
  - Dirección GMSC: "5551234501"
MSC → HLR: Respuesta PRN con MSRN: "49170999888777"
HLR → GMSC: Respuesta SRI con información de enrutamiento:
  - IMSI
  - Número VLR: "49170123456"
  - Número en Roaming (MSRN): "49170999888777"
```

Diferencias en la Estructura de Respuesta

Respuesta SRI de Suscriptor de Hogar

```
%{
  imsi: "999999876543210",
  extendedRoutingInfo: {
    :camelRoutingInfo, %{
      gmscCamelSubscriptionInfo: %{
        "t-CSI": %{
          serviceKey: 11_110,
          "gsmSCF-Address": "5551234501",
          defaultCallHandling: :continueCall,
          "t-BcsmTriggerDetectionPoint": :termAttemptAuthorized
        }
      }
    }
  },
  subscriberInfo: %{
    locationInformation: %{"vlr-number": "5551234567"},
    subscriberState: {:notProvidedFromVLR, :NULL}
  }
}
```

Respuesta SRI de Suscriptor en Roaming

```
%{
  imsi: "999999876543210",
  extendedRoutingInfo: {
    :routingInfo, %{
      roamingNumber: "49170999888777"  # MSRN de PRN
    }
  },
  subscriberInfo: %{
    locationInformation: %{"vlr-number": "49170123456"},
    subscriberState: {:notProvidedFromVLR, :NULL}
  }
}
```

Operación de Proveer Número en Roaming (PRN)

Estructura de Solicitud PRN

La solicitud PRN enviada al MSC/VLR contiene:

Campo	Fuente	Descripción
MSISDN	Solicitud SRI	Número de teléfono del suscriptor
IMSI	API HLR	IMSI del suscriptor
Número MSC	API HLR	MSC que atiende al suscriptor en roaming (`serving_msc`)
Dirección GMSC	Solicitud SRI	GMSC que realiza la solicitud SRI original
Número de Referencia de Llamada	Estático	Identificador de referencia de llamada
Fases CAMEL Soportadas	Estático	Fases CAMEL soportadas por GMSC

Manejo de Respuesta PRN

El HLR espera una respuesta PRN que contenga:

- **MSRN** (Número de Estación Móvil en Roaming): Un número temporal asignado por la red visitada para enrutar la llamada

Manejo de Errores:

- Si PRN se agota → Devuelve error 27 (Suscriptor Ausente) en la respuesta SRI
- Si PRN falla → Devuelve error 27 (Suscriptor Ausente) en la respuesta SRI
- Si no se puede extraer MSRN → Devuelve error 27 (Suscriptor Ausente) en la respuesta SRI

Ejemplos de Configuración

Operador de Red de Hogar Único

```
# Todas las direcciones VLR que comienzan con "555123" se
consideran de hogar
home_vlr_prefixes: ["555123"],
```

- VLR `5551234567` → Hogar (respuesta CAMEL)
- VLR `5551235001` → Hogar (respuesta CAMEL)
- VLR `49170123456` → Roaming (PRN + respuesta MSRN)

Operador Multi-Región

```
# Múltiples redes de hogar en diferentes regiones
home_vlr_prefixes: ["555123", "555124", "555125"],
```

- VLR `5551234567` → Hogar (región 1)
- VLR `5552341234` → Hogar (región 2)
- VLR `5553411111` → Hogar (región 3)
- VLR `44201234567` → Roaming (internacional)

Configuración de Pruebas

Para probar la funcionalidad PRN, establezca una lista vacía para tratar todos los VLR como roaming:

```
# Todos los VLRs se tratan como roaming (para probar flujo PRN)
home_vlr_prefixes: [],
```

Mejores Prácticas

- **Selección de Prefijos**: Utilice el prefijo único más corto que identifique los VLRs de su red (por ejemplo, código de país + código de red)
- **Múltiples Prefijos**: Incluya todos los prefijos de VLR en su red, incluidos diferentes regiones y subsidiarias

- **Acuerdos de Roaming**: Asegúrese de que PRN sea soportado adecuadamente por las redes de socios de roaming
- **Pruebas**: Pruebe a fondo tanto escenarios de hogar como de roaming antes de la implementación en producción
- **Monitoreo**: Monitoree las tasas de tiempo de espera de PRN para identificar problemas de conectividad con socios de roaming

Solución de Problemas

Síntoma: Todos los suscriptores tratados como roaming

- **Causa**: `home_vlr_prefixes` no configurado o los prefijos no coinciden con las direcciones VLR
- **Solución**: Verifique las direcciones VLR en su base de datos y actualice los prefijos en consecuencia

Síntoma: Solicitudes PRN que se agotan

- **Causa**: Problemas de conectividad de red con el MSC/VLR del socio de roaming
- **Solución**: Verifique el enrutamiento M3UA/SCCP a las direcciones MSC remotas

Síntoma: MSRN inválido en la respuesta SRI

- **Causa**: El formato de respuesta PRN del socio de roaming no coincide con la estructura esperada
- **Solución**: Revise los registros de respuesta PRN y ajuste `extract_msrn_from_prn/1` si es necesario

---

Operaciones HLR

Operaciones MAP Soportadas

- `updateLocation` (Código de operación 2) - Registrar ubicación VLR

- `sendAuthenticationInfo` (Código de operación 56) - Generar vectores de autenticación
- `sendRoutingInfo` (Código de operación 22) - Proporcionar MSRN para llamadas con soporte CAMEL
- `sendRoutingInfoForSM` (Código de operación 45) - Proporcionar GT de MSC para SMS
- `cancelLocation` (Código de operación 3) - Desregistrar del antiguo VLR
- `insertSubscriberData` (Código de operación 7) - Enviar perfil de suscriptor

Mapeo de Campos de Respuesta

Esta sección detalla de dónde proviene cada campo en las respuestas HLR.

Respuesta SendRoutingInfo (SRI)

Propósito: Proporciona información de enrutamiento para llamadas entrantes a un suscriptor.

El HLR proporciona dos tipos de respuesta diferentes según si el suscriptor está en un VLR de hogar o en roaming:

Respuesta de Suscriptor de Hogar (Enrutamiento CAMEL)

Usado cuando la dirección VLR del suscriptor comienza con un valor configurado en `home_vlr_prefixes`.

Estructura de Respuesta:

Campo	Fuente	Descripción
IMSI	API OmniHSS	IMSI del suscriptor de la base de datos de OmniHSS
Número VLR	API OmniHSS	VLR actual que atiende al suscriptor (`circuit_session.assigned_vlr`)
Estado del Suscriptor	Estático	Siempre `notProvidedFromVLR`
extendedRoutingInfo	-	Tipo: `camelRoutingInfo`
Dirección gsmSCF	API OmniHSS	MSC que atiende al suscriptor (`circuit_session.assigned_msc`)
Clave de Servicio	runtime.exs	Identificador de servicio CAMEL (`camel_service_key`)
Punto de Detección de Disparador	runtime.exs	Cuándo activar CAMEL (`camel_trigger_detection_point`)
Manejo de Capacidades CAMEL	Estático	Nivel de soporte de fase CAMEL
Manejo de Llamadas por Defecto	Estático	Respaldo si gsmSCF no está disponible

Respuesta de Suscriptor en Roaming (Enrutamiento MSRN)

Usado cuando la dirección VLR del suscriptor NO coincide con ningún valor configurado en `home_vlr_prefixes`.

Estructura de Respuesta:

Campo	Fuente	Descripción	
IMSI	API OmniHSS	IMSI del suscriptor de la base de datos de OmniHSS	'
Número VLR	API OmniHSS	VLR actual que atiende al suscriptor (`circuit_session.assigned_vlr`)	'
Estado del Suscriptor	Estático	Siempre `notProvidedFromVLR`	:
extendedRoutingInfo	-	Tipo: `routingInfo`	-
Número en Roaming (MSRN)	Respuesta PRN	MSRN obtenido de la solicitud ProvideRoamingNumber	'

Lógica de Decisión de Enrutamiento:

```
1. OmniSS7 recibe solicitud SendRoutingInfo
2. OmniSS7 consulta datos del suscriptor desde la API de OmniHSS
3. OmniSS7 verifica la dirección VLR contra home_vlr_prefixes:

   Si VLR comienza con prefijo de hogar:
     → Devolver información de enrutamiento CAMEL (flujo de
suscriptor de hogar)

   Si VLR NO coincide con ningún prefijo de hogar:
     → Enviar ProvideRoamingNumber (PRN) al MSC
     → Extraer MSRN de la respuesta PRN
     → Devolver información de enrutamiento con MSRN (flujo de
suscriptor en roaming)
```

Flujo de Datos:

- OmniSS7 consulta a OmniHSS para obtener información del suscriptor
- OmniHSS devuelve IMSI, ubicación actual VLR/MSC y estado del suscriptor
- OmniSS7 utiliza estos datos para construir la respuesta MAP

Requisitos de Configuración:

```
# En runtime.exs
home_vlr_prefixes: ["555123"],  # Lista de prefijos de VLR de
hogar
```

Respuestas de Error:

- Si `serving_vlr` y `serving_msc` son `null`: Devuelve error 27 (Suscriptor Ausente)
- Si el suscriptor no se encuentra: Devuelve error 1 (Suscriptor Desconocido)
- Si la solicitud PRN se agota (caso de roaming): Devuelve error 27 (Suscriptor Ausente)
- Si la respuesta PRN es inválida (caso de roaming): Devuelve error 27 (Suscriptor Ausente)

Respuesta UpdateLocation con InsertSubscriberData

Propósito: Registra al suscriptor en un nuevo VLR y provisiona datos del suscriptor.

Respuesta UpdateLocation FIN

Campo	Fuente	Descripción	Ejemplo
Número HLR	runtime.exs	Título Global de este HLR (`hlr_service_center_gt_address`)	`"5551234568`
Tipo de Mensaje TCAP	Estático	Respuesta final después de todos los ISDs	`FIN`

InsertSubscriberData #1 (Datos Básicos del Suscriptor)

Campo	Fuente	Descripción	Ejemplo
IMSI	Solicitud	De la solicitud UpdateLocation	`"999999876543`
MSISDN	API OmniHSS	Número de teléfono del suscriptor de OmniHSS	`"555123456"`
Categoría	Estático	Categoría del suscriptor	`"\n"` (0x0A)
Estado del Suscriptor	Estático	Estado del servicio	`:serviceGrant`
Lista de Servicios de Portadora	Estático	Servicios de portadora soportados	`[<<31>>]`
Lista de Teleservicios	Estático	Teleservicios soportados	`[<<17>>, "` `"\""]`
Modo de Acceso a la Red	runtime.exs	Acceso por paquete/circuito (`isd_network_access_mode`)	`:packetAndCi`

InsertSubscriberData #2 (Servicios Suplementarios) - Opcional

Campo	Fuente	Descripción	Controlado Por
SS Provisionados	Estático	Datos de servicios suplementarios	`isd_send_ss_data: true`
Desvío de Llamadas	Estático	Configuraciones de desvío (incondicional, ocupado, sin respuesta, no alcanzable)	Config habilitado
Espera de Llamadas	Estático	Estado del servicio de espera de llamadas	Config habilitado
Servicio de Múltiples Partes	Estático	Soporte para llamadas de conferencia	Config habilitado

ISD #2 incluye:

- Desvío de llamadas incondicional (código SS 21)
- Desvío de llamadas en ocupado (código SS 41)
- Desvío de llamadas en sin respuesta (código SS 42)
- Desvío de llamadas en no alcanzable (código SS 62)
- Espera de llamadas (código SS 43)
- Servicio de múltiples partes (código SS 51)
- Servicios CLIP/CLIR

InsertSubscriberData #3 (Barring de Llamadas) - Opcional

Campo	Fuente	Descripción	Controlado Por
Información de Barring de Llamadas	Estático	Configuraciones de barring de llamadas	`isd_send_call_barring: true`
BAOC	Estático	Barring de Todas las Llamadas Salientes (código SS 146)	Config habilitado
BOIC	Estático	Barring de Llamadas Internacionales Salientes (código SS 147)	Config habilitado
Datos de Restricción de Acceso	Estático	Restricciones de acceso a la red	Config habilitado

Control de Secuencia ISD:

- ISD #1: **Siempre enviado** - Contiene datos esenciales del suscriptor
- ISD #2: Enviado solo si `isd_send_ss_data: true` en runtime.exs
- ISD #3: Enviado solo si `isd_send_call_barring: true` en runtime.exs

Respuesta SendRoutingInfoForSM (SRI-for-SM)

Propósito: Proporciona información de enrutamiento MSC/SMSC para la entrega de SMS. Cuando un SMSc necesita entregar un SMS a un suscriptor, envía una solicitud SRI-for-SM al HLR para determinar dónde enrutar el mensaje.

Estructura de Respuesta:

Campo	Fuente	Descripción	Cómo Generado
IMSI	Calculado	IMSI sintético derivado de MSISDN	`PLMN_PREFIX + zero_padded_MSISDN`
Número de Nodo de Red	runtime.exs	Dirección GT SMSc para enrutamiento de SMS	`smsc_service_center_gt_address`

Parámetros de Configuración (de `runtime.exs`):

```
# Dirección del Centro de Servicio GT (devuelta en respuestas SRI-
for-SM)
# Esto le dice al SMSc solicitante dónde enviar mensajes MT-
ForwardSM
smsc_service_center_gt_address: "5551234567",  # Requerido

# Configuración de Mapeo MSISDN ↔ IMSI
# Prefijo PLMN: MCC (001 = Red de Prueba) + MNC (01 = Operador de
Prueba)
hlr_imsi_plmn_prefix: "001001",                 # ¡Único
parámetro de configuración necesario!
```

Mapeo MSISDN ↔ IMSI

Parámetros de Configuración:

Estos parámetros controlan cómo OmniSS7 genera IMSIs sintéticos a partir de MSISDNs para respuestas SRI-for-SM:

- **`hlr_imsi_plmn_prefix`**: El prefijo MCC+MNC que se utilizará al construir IMSIs sintéticos (por ejemplo, "50557" para MCC=505, MNC=57)
- **`hlr_msisdn_country_code`**: Código de país que se debe anteponer al hacer el mapeo inverso IMSI→MSISDN (por ejemplo, "61" para Australia, "1" para EE.UU./Canadá)

- `hlr_msisdn_nsn_offset`: Posición de carácter donde comienza el Número de Suscriptor Nacional (NSN) dentro del MSISDN (típicamente 0 si el MSISDN no incluye código de país, o longitud del código de país si lo incluye)
- `hlr_msisdn_nsn_length`: Número de dígitos a extraer del MSISDN como el NSN

Para detalles adicionales de configuración, consulte Mapeo MSISDN ↔ IMSI en la Referencia de Configuración.

¿Por qué se Necesita el Mapeo MSISDN a IMSI?

El protocolo MAP para **SendRoutingInfoForSM** (SRI-for-SM) requiere que el HLR devuelva un **IMSI** (Identidad Internacional de Suscriptor Móvil) en su respuesta. Sin embargo, el SMSc solicitante solo conoce el **MSISDN** (número de teléfono del suscriptor).

En una red tradicional:

- El SMSc envía SRI-for-SM con el MSISDN de destino (por ejemplo, "5551234567")
- El HLR debe buscar al suscriptor en su base de datos para encontrar su IMSI
- El HLR devuelve el IMSI en la respuesta SRI-for-SM
- El SMSc luego utiliza este IMSI al enviar MT-ForwardSM al MSC/VLR

Enfoque de OmniSS7 - IMSIs Sintéticos:

En lugar de mantener una base de datos completa de suscriptores con mapeos MSISDN a IMSI, OmniSS7 utiliza un esquema de codificación simple para **calcular** IMSIs sintéticos directamente a partir del MSISDN. Este enfoque proporciona dos beneficios clave:

1. **Privacidad**: Los verdaderos IMSIs de suscriptores almacenados en la base de datos HLR nunca se exponen en las respuestas SRI-for-SM enviadas a través de la red SS7
2. **Simplicidad**: No es necesario consultar la base de datos HLR para búsquedas de IMSI durante las operaciones SRI-for-SM - el IMSI se calcula sobre la marcha a partir del MSISDN

Cómo Funciona:

Los MSISDNs se codifican directamente en la porción de suscriptor del IMSI (los dígitos después de MCC+MNC):

```
IMSI = PLMN_PREFIX + zero_padded_MSISDN
```

Donde:

- **PLMN_PREFIX**: MCC + MNC (por ejemplo, "001001" para Red de Prueba)
- **MSISDN**: Todos los dígitos numéricos del número de teléfono
- **Relleno con Ceros**: Rellenado a la izquierda con ceros para llenar el IMSI a exactamente 15 dígitos

Ejemplo Paso a Paso:

```
# Configuración
plmn_prefix = "001001"  # MCC 001 + MNC 01

# Entrada: MSISDN de la solicitud SRI-for-SM (decodificado TBCD)
msisdn = "555123456"    # 9 dígitos

# Paso 1: Calcular espacio disponible para el número de suscriptor
subscriber_digits = 15 - String.length("001001")  # = 9 dígitos

# Paso 2: Rellenar MSISDN con ceros a la izquierda para llenar la
porción de suscriptor
padded_msisdn = String.pad_leading("555123456", 9, "0")  # =
"555123456" (sin relleno necesario)

# Paso 3: Concatenar prefijo PLMN + MSISDN rellenado
imsi = "001001" <> "555123456"  # = "001001555123456" (exactamente
15 dígitos)
```

Ejemplos Completos:

MSISDN de Entrada	Prefijo PLMN	Dígitos de Suscriptor Disponibles	MSISDN Rellenado	IMSI Fina
`"555123456"`	`"001001"` (6)	9	`"555123456"`	`"001001555123`
`"99"`	`"001001"` (6)	9	`"000000099"`	`"001001000000`
`"999999999"`	`"001001"` (6)	9	`"999999999"`	`"001001999999`
`"91123456789"`	`"001001"` (6)	9	`"555123456"`	`"001001555123`

Manejo de Casos Límite:

- **MSISDNs Cortos**: Rellenados a la izquierda con ceros (por ejemplo, `"99"` → `"000000099"`)
- **MSISDNs Largos**: Se mantienen los dígitos más a la derecha, se recortan los dígitos más a la izquierda (por ejemplo, `"91123456789"` → `"555123456"`)
- **Longitud del IMSI**: Siempre exactamente 15 dígitos

Manejo de Mapeo Inverso (IMSI → MSISDN):

El SMSc puede revertir este mapeo para convertir IMSIs de nuevo a MSISDNs:

```
# Entrada: IMSI de la respuesta SRI-for-SM
imsi = "001001555123456"

# Paso 1: Eliminar prefijo PLMN
plmn_prefix = "001001"
subscriber_portion = String.slice(imsi, 6, 9)  # = "555123456"

# Paso 2: Eliminar ceros a la izquierda para obtener el MSISDN
real
msisdn = String.replace_leading(subscriber_portion, "0", "")  # =
"555123456"
```

Ejemplos de Mapeo Inverso:

IMSI de Entrada	Prefijo PLMN	Porción de Suscriptor	Eliminar Ceros a la Izquierda	MSISDI Final
`"001001555123456"`	`"001001"`	`"555123456"`	`"555123456"`	`"5551234`
`"001001000000099"`	`"001001"`	`"000000099"`	`"99"`	`"99"`
`"001001999999999"`	`"001001"`	`"999999999"`	`"999999999"`	`"9999999`

Propiedades de Este Mapeo:

- **Determinista**: El mismo MSISDN siempre produce el mismo IMSI
- **Reversible**: Se puede convertir de IMSI a MSISDN
- **Configuración Mínima**: Solo requiere `hlr_imsi_plmn_prefix`
- **Protección de Privacidad**: Los verdaderos IMSIs nunca se exponen
- **Sin Búsqueda en Base de Datos**: Cálculo rápido, no se necesitan llamadas API
- **Siempre 15 Dígitos**: El IMSI es siempre exactamente 15 dígitos

Manejo de Entrada MSISDN:

Cuando el HLR recibe una solicitud SRI-for-SM, el MSISDN pasa por decodificación TBCD:

1. **Decodificación TBCD**: Convertir TBCD binario a cadena (puede incluir prefijo TON/NPI como "91")
2. **Extraer Dígitos**: Mantener solo dígitos numéricos, eliminar cualquier carácter no numérico
3. **Normalizar**: Si es más largo que el espacio disponible, tomar los dígitos más a la derecha; si es más corto, rellenar con ceros a la izquierda
4. **Codificar**: Concatenar prefijo PLMN + MSISDN normalizado

Consideraciones de Seguridad:

Los IMSIs sintéticos devueltos en las respuestas SRI-for-SM son puramente para fines de enrutamiento. No son los verdaderos IMSIs almacenados en la base de datos de suscriptores HLR. Esto proporciona una capa adicional de protección de privacidad, ya que los verdaderos IMSIs de suscriptores solo se exponen cuando es absolutamente necesario (por ejemplo, durante operaciones UpdateLocation o SendAuthenticationInfo que requieren verdaderos vectores de autenticación).

Flujo de Respuesta:

```
1. SMSc → HLR: Solicitud SRI-for-SM
   - MSISDN (TBCD): "91123456789" (incluye TON/NPI)

2. Procesamiento HLR:
   - Decodificación TBCD: "91123456789"
   - Extraer dígitos: "91123456789" (11 dígitos)
   - Ajustar a 9 dígitos: "555123456" (9 más a la derecha)
   - Agregar PLMN: "001001" + "555123456" = "001001555123456"
   - Obtener GT SMSc de la configuración: "5551234567"

3. HLR → SMSc: Respuesta SRI-for-SM
   - IMSI: "001001555123456" (sintético, siempre 15 dígitos)
   - Número de Nodo de Red: "5551234567" (dónde enviar MT-
ForwardSM)

4. SMSc envía MT-ForwardSM a "5551234567" con IMSI
"001001555123456"
```

Configuración:

Los siguientes parámetros se utilizan en `runtime.exs`:

```
# Prefijo PLMN: MCC (001 = Red de Prueba) + MNC (01 = Operador de
Prueba)
hlr_imsi_plmn_prefix: "001001",

# Extracción de NSN (si los MSISDN incluyen código de país)
hlr_msisdn_country_code: "1",      # Utilizado para mapeo inverso
(IMSI→MSISDN)
hlr_msisdn_nsn_offset: 1,          # Omitir 1 dígito del código de
país
hlr_msisdn_nsn_length: 10          # Extraer 10 dígitos de NSN
```

Configuración de Extracción de NSN:

Si sus MSISDN incluyen el código de país (por ejemplo, `"68988000088"` en lugar de solo `"88000088"`), debe configurar la extracción de NSN:

- **`hlr_msisdn_nsn_offset`**: Posición donde comienza el NSN (típicamente la longitud de su código de país)
- **`hlr_msisdn_nsn_length`**: Número de dígitos en el NSN

Ejemplos:

Ejemplo	Código de País	Ejemplo MSISDN	nsn_offset	nsn_length	NS Extr
CC de 1 dígito	`"9"`	`"95551234567"`	`1`	`10`	`"55512`
CC de 2 dígitos	`"99"`	`"99412345678"`	`2`	`9`	`"41234`
CC de 3 dígitos	`"999"`	`"99988000088"`	`3`	`8`	`"88000`

Cómo Funciona:

1. **MSISDN → IMSI**: Extraer NSN del MSISDN, rellenar con ceros a la izquierda, concatenar con prefijo PLMN

```
MSISDN: "99988000088"
NSN: String.slice("99988000088", 3, 8) = "88000088"
MSISDN Rellenado: "088000088" (9 dígitos)
IMSI: "547050" + "088000088" = "547050088000088"
```

2. **IMSI → MSISDN**: Eliminar prefijo PLMN, eliminar ceros a la izquierda, anteponer código de país

```
IMSI: "547050088000088"
Porción de Suscriptor: "088000088"
Eliminar ceros: "88000088"
MSISDN: "+999" + "88000088" = "+99988000088"
```

Requisitos de API: Ninguno - SRI-for-SM utiliza valores calculados y configuración solamente. No se

requieren llamadas a la API de backend.

Resumen de Fuentes de Campos

Tipo de Fuente	Descripción	Ejemplos
API OmniHSS	Datos dinámicos de la base de datos de suscriptores OmniHSS	IMSI, MSISDN, VLR/MSC que atiende desde circuit_session
runtime.exs	Parámetros de configuración de OmniSS7	`smsc_service_center_gt_address`, `camel_service_key`, `isd_network_access_mode`
Estático	Valores codificados en el generador de respuestas	Estado del suscriptor, servicios de portadora, códigos SS
Solicitud	Campos extraídos de la solicitud MAP entrante	IMSI de UpdateLocation, MSISDN de SRI
Calculado	Valores derivados utilizando lógica	IMSI sintético en SRI-for-SM (hlr_imsi_prefix + NSN)

Dependencias de Configuración

Requerido en runtime.exs:

- `hlr_service_center_gt_address` - Usado en respuestas UpdateLocation
- `smsc_service_center_gt_address` - Usado en respuestas SRI-for-SM (dónde se deben enrutar MT-ForwardSM)

Opcional en runtime.exs (con valores predeterminados):

- `camel_service_key` - Predeterminado: `11_110`
- `camel_trigger_detection_point` - Predeterminado: `:termAttemptAuthorized`
- `isd_network_access_mode` - Predeterminado: `:packetAndCircuit`
- `isd_send_ss_data` - Predeterminado: `true`
- `isd_send_call_barring` - Predeterminado: `true`
- `hlr_imsi_plmn_prefix` - Predeterminado: `"001001"` (prefijo PLMN para mapeo MSISDN↔IMSI)

Requerido de OmniHSS:

OmniHSS debe proporcionar puntos finales de API REST para:

- Búsqueda de suscriptores por IMSI y MSISDN
- Actualizaciones de ubicación de sesión de circuito (asignación de VLR/MSC)
- Generación de vectores de autenticación
- Consultas sobre estado de suscriptores y perfiles de servicio

---

Documentación Relacionada

Documentación de OmniSS7:

- ← Volver a la Documentación Principal
- Guía de Características Comunes
- Guía del Cliente MAP
- Referencia Técnica
- Referencia de Configuración

Documentación de OmniHSS: Para gestión de suscriptores, provisión, configuración de autenticación y operaciones administrativas, consulte la **documentación del producto OmniHSS**. OmniHSS contiene toda la lógica de base de datos de suscriptores, algoritmos de autenticación, reglas de provisión de servicios y capacidades de gestión Multi-IMSI.

OmniSS7 por Omnitouch Network Services

Guía de Configuración del Cliente MAP

← Volver a la Documentación Principal

Esta guía proporciona una configuración detallada para usar OmniSS7 como un **Cliente MAP** para enviar solicitudes del protocolo MAP a elementos de red.

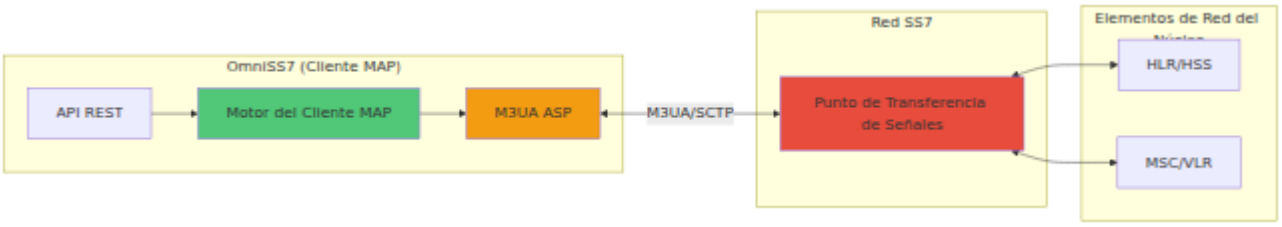
Tabla de Contenidos

¿Qué es el Modo Cliente MAP?

El **Modo Cliente MAP** permite que OmniSS7 se conecte como un **Proceso de Servidor de Aplicaciones (ASP)** a un par M3UA (STP o SGP) y envíe/reciba mensajes **MAP (Parte de Aplicación Móvil)** para servicios como:

- **Consultas HLR**: SRI (Enviar Información de Enrutamiento), SRI-para-SM, Información de Autenticación
- **Actualizaciones de Ubicación**: Actualizar Ubicación, Cancelar Ubicación
- **Gestión de Suscriptores**: Proveer Número de Roaming (PRN), Insertar Datos del Suscriptor

Arquitectura de Red



Habilitar el Modo Cliente MAP

Edita `config/runtime.exs` y configura los ajustes del cliente MAP. Para una referencia de configuración completa, consulta Parámetros de Conexión M3UA en la Referencia de Configuración.

Configuración Básica

```
config :omniss7,
  # Habilitar el modo Cliente MAP
  map_client_enabled: true,

  # Conexión M3UA para el Cliente MAP (se conecta como ASP a
STP/SGP remoto)
  map_client_m3ua: %{
    mode: "ASP",                      # Modo M3UA: "ASP" (cliente)
o "SGP" (servidor)
    callback: {MapClient, :handle_payload, []},  # Callback para
mensajes entrantes
    process_name: :map_client_asp,    # Nombre del proceso
registrado
    local_ip: {10, 0, 0, 100},       # Dirección IP local
    local_port: 2905,                 # Puerto SCTP local
    remote_ip: {10, 0, 0, 1},        # IP STP/SGP remota
    remote_port: 2905,                # Puerto STP/SGP remoto
    routing_context: 1                # Contexto de enrutamiento
M3UA
  }
```

Ejemplo de Configuración en Producción

```
config :omniss7,
  # Habilitar el Cliente MAP para producción
  map_client_enabled: true,

  # Conexión M3UA en producción
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :map_client_asp,
    local_ip: {10, 0, 0, 100},
    local_port: 2905,
    remote_ip: {10, 0, 0, 1},        # IP STP en producción
    remote_port: 2905,
    routing_context: 1
  }

config :control_panel,
  web: %{
    listen_ip: "0.0.0.0",
    port: 443,
    hostname: "ss7-gateway.example.com",
    enable_tls: true,
    tls_cert: "/etc/ssl/certs/gateway.crt",
    tls_key: "/etc/ssl/private/gateway.key"
  }
```

Operaciones MAP Disponibles

1. Enviar Información de Enrutamiento para SM (SRI-para-SM)

Consulta al HLR para determinar el MSC que sirve para la entrega de SMS. Para información detallada sobre cómo el HLR procesa las solicitudes SRI-para-SM, consulta SRI-para-SM en la Guía HLR.

Endpoint de la API: `POST /api/sri-for-sm`

Solicitud:

```
{
  "msisdn": "447712345678",
  "serviceCenter": "447999123456"
}
```

Respuesta:

```
{
  "result": {
    "imsi": "234509876543210",
    "locationInfoWithLMSI": {
      "networkNode-Number": "447999555111"
    }
  }
}
```

Ejemplo de cURL:

```
curl -X POST http://localhost/api/sri-for-sm \
  -H "Content-Type: application/json" \
  -d '{
    "msisdn": "447712345678",
    "serviceCenter": "447999123456"
  }'
```

---

2. Enviar Información de Enrutamiento (SRI)

Consulta al HLR para obtener información de enrutamiento de llamadas de voz.

Endpoint de la API: `POST /api/sri`

Solicitud:

```
{
  "msisdn": "447712345678",
  "gmsc": "447999123456"
}
```

Respuesta:

```
{
  "result": {
    "imsi": "234509876543210",
    "extendedRoutingInfo": {
      "routingInfo": {
        "roamingNumber": "447999555222"
      }
    }
  }
}
```

---

3. Proveer Número de Roaming (PRN)

Solicita un número de roaming temporal (MSRN) del MSC que sirve.

Endpoint de la API: `POST /api/prn`

Solicitud:

```
{
  "msisdn": "447712345678",
  "gmsc": "447999123456",
  "msc_number": "447999555111",
  "imsi": "234509876543210"
}
```

---

4. Enviar Información de Autenticación

Solicita vectores de autenticación del HLR para la autenticación del suscriptor.

Endpoint de la API: `POST /api/send-auth-info`

Solicitud:

```
{
  "imsi": "234509876543210",
  "vectors": 5
}
```

Respuesta:

```
{
  "result": {
    "authenticationSetList": [
      {
        "rand": "0123456789ABCDEF0123456789ABCDEF",
        "xres": "ABCDEF0123456789",
        "ck": "0123456789ABCDEF0123456789ABCDEF",
        "ik": "FEDCBA9876543210FEDCBA9876543210",
        "autn": "0123456789ABCDEF0123456789ABCDEF"
      }
    ]
  }
}
```

---

5. Actualizar Ubicación

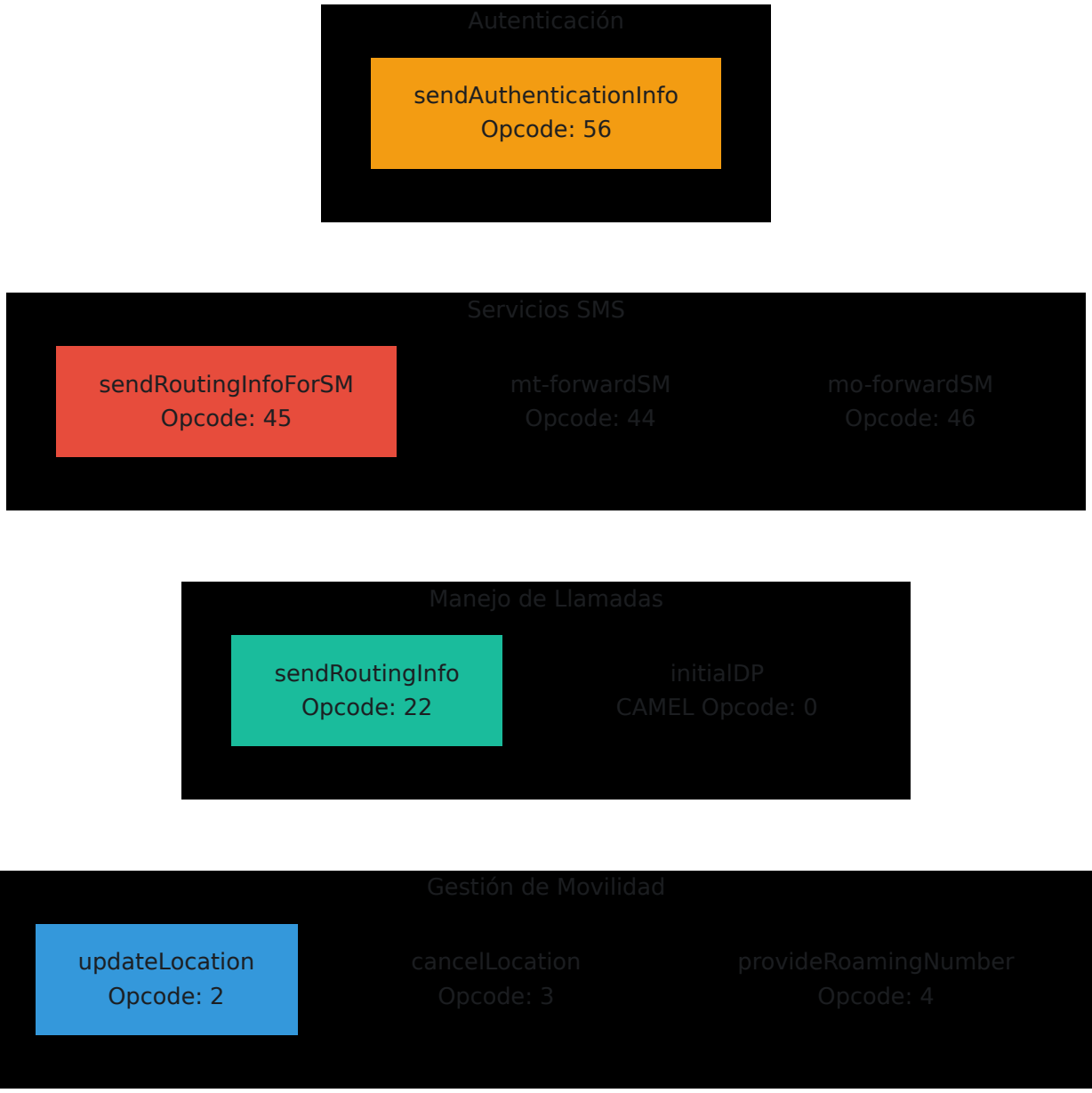
Registra la ubicación actual de un suscriptor con el HLR. Para información detallada sobre el procesamiento de UpdateLocation y las secuencias de InsertSubscriberData, consulta Actualizaciones de Ubicación en la Guía HLR.

Endpoint de la API: `POST /api/updateLocation`

Solicitud:

```
{
  "imsi": "234509876543210",
  "vlr": "447999555111"
}
```

Resumen de Operaciones MAP



Enviando Solicitudes a través de la API

Usando Swagger UI

La interfaz Swagger UI proporciona una interfaz interactiva para enviar solicitudes SS7.

Acceder a Swagger UI:

1. Navega a `http://your-server/swagger`
2. Explora los endpoints de API disponibles
3. Haz clic en cualquier endpoint para expandir sus detalles

Enviando una Solicitud:

1. Haz clic en el endpoint que deseas usar (por ejemplo, `/api/sri-for-sm`)
2. Haz clic en el botón "Try it out"
3. Completa los parámetros requeridos en el cuerpo de la solicitud
4. Haz clic en "Execute"
5. Visualiza la respuesta a continuación

Códigos de Respuesta de la API

- **200** - Éxito, resultado devuelto en el cuerpo de la respuesta
- **400** - Solicitud Incorrecta, parámetros inválidos
- **504** - Tiempo de Espera del Gateway, sin respuesta de la red SS7 dentro de los 10 segundos

---

Métricas del Cliente MAP

Métricas Disponibles

Métricas de Solicitud:

- `map_requests_total` - Número total de solicitudes MAP enviadas
  - Etiquetas: `operation` (valores: `sri`, `sri_for_sm`, `prn`, `authentication_info`, etc.)
- `map_request_errors_total` - Número total de errores de solicitudes MAP
  - Etiquetas: `operation`

- `map_request_duration_milliseconds` - Histograma de duraciones de solicitudes MAP
  - Etiquetas: `operation`
- `map_pending_requests` - Número actual de solicitudes MAP pendientes (gauge)

Ejemplos de Consultas Prometheus

```
# Total de solicitudes SRI-para-SM en la última hora
increase(map_requests_total{operation="sri_for_sm"}[1h])

# Tiempo de respuesta promedio para solicitudes SRI
rate(map_request_duration_milliseconds_sum{operation="sri"}[5m]) /
rate(map_request_duration_milliseconds_count{operation="sri"}[5m])

# Tasa de errores para todas las operaciones MAP
sum(rate(map_request_errors_total[5m])) by (operation)

# Solicitudes pendientes actuales
map_pending_requests
```

Solución de Problemas del Cliente MAP

Problema: Tiempo de Espera de Solicitudes

Síntomas:

- La API devuelve 504 Gateway Timeout
- Sin respuesta del HLR/MSC

Verificaciones:

1. Verifica que la conexión M3UA esté ACTIVA:

```
# En la consola IEx
:sys.get_state(:map_client_asp)
```

2. Verifica la conectividad de red al STP
3. Verifica el contexto de enrutamiento y la dirección SCCP
4. Revisa los registros en busca de errores SCCP

Problema: Errores SCCP

Síntomas:

- La API devuelve respuestas de error SCCP
- Los registros muestran mensajes de "servicio unitdata SCCP"

Códigos de Error SCCP Comunes:

- **Sin Traducción**: Título Global no encontrado en la tabla de enrutamiento del STP
- **Fallo de Subsistema**: Subsistema de destino (HLR SSN 6) no está disponible
- **Fallo de Red**: Congestión o fallo de red

Soluciones:

- Contacta al administrador del STP para verificar la configuración de enrutamiento
- Verifica que el Título Global de destino sea alcanzable
- Verifica si el subsistema de destino está operativo

Documentación Relacionada

- ← Volver a la Documentación Principal
- Guía de Características Comunes - Interfaz Web, API, Monitoreo

- Guía STP - Configuración de enrutamiento
- Guía del Centro SMS - Entrega de SMS
- Referencia Técnica - Especificaciones de protocolo

OmniSS7 por Omnitouch Network Services

Guía de Configuración del Centro de SMS (SMSc)

← Volver a la Documentación Principal

Esta guía proporciona una configuración detallada para usar OmniSS7 como un **Centro de SMS (SMSc)** en el frontend con **OmniMessage** como la plataforma de almacenamiento y entrega de mensajes en el backend.

Integración de OmniMessage

El modo SMSc de OmniSS7 funciona como un frontend de señalización SS7 que se conecta con **OmniMessage**, una plataforma de SMS de grado de operador. Esta arquitectura separa las preocupaciones:

- **OmniSS7 (Frontend SMSc)**: Maneja toda la señalización del protocolo SS7/MAP, el enrutamiento SCCP y la comunicación de red
- **OmniMessage (Backend SMS)**: Gestiona el almacenamiento de mensajes, la cola, la lógica de reintentos, el seguimiento de entrega y las decisiones de enrutamiento

¿Por qué OmniMessage?

OmniMessage proporciona capacidades de mensajería SMS de grado de operador con características que incluyen:

- **Gestión de Cola de Mensajes**: Almacenamiento persistente con lógica de reintentos configurable y encolado por prioridad
- **Seguimiento de Entrega**: Estado de entrega en tiempo real, informes de entrega (DLR) y seguimiento de razones de fallo
- **Soporte Multi-SMSc**: Múltiples instancias de frontend pueden conectarse a un único backend de OmniMessage para balanceo de carga y

redundancia

- **Inteligencia de Enrutamiento**: Reglas de enrutamiento avanzadas basadas en destino, remitente, contenido del mensaje y hora del día
- **Limitación de Tasa**: Controles TPS (transacciones por segundo) por ruta para prevenir congestión de red
- **Diseño API-Primero**: API HTTP RESTful para integración con sistemas de facturación, portales de clientes y aplicaciones de terceros
- **Analítica e Informes**: Estadísticas de volumen de mensajes, tasas de éxito de entrega y métricas de rendimiento

Todos los datos de mensajes, estado de entrega y configuraciones de enrutamiento se almacenan y gestionan en OmniMessage. OmniSS7 consulta a OmniMessage a través de llamadas API HTTPS para recuperar mensajes pendientes, actualizar el estado de entrega y registrarse como un frontend activo.

Importante: El modo SMSc de OmniSS7 es un **frontend de señalización solamente**. Toda la lógica de enrutamiento de mensajes, gestión de colas, algoritmos de reintentos, seguimiento de entrega y reglas de negocio son manejadas por OmniMessage. Esta guía cubre la configuración del protocolo SS7/MAP en OmniSS7. Para información sobre enrutamiento de mensajes, configuración de colas, informes de entrega, limitación de tasa y analítica, **consulte la documentación de OmniMessage**.

Tabla de Contenidos

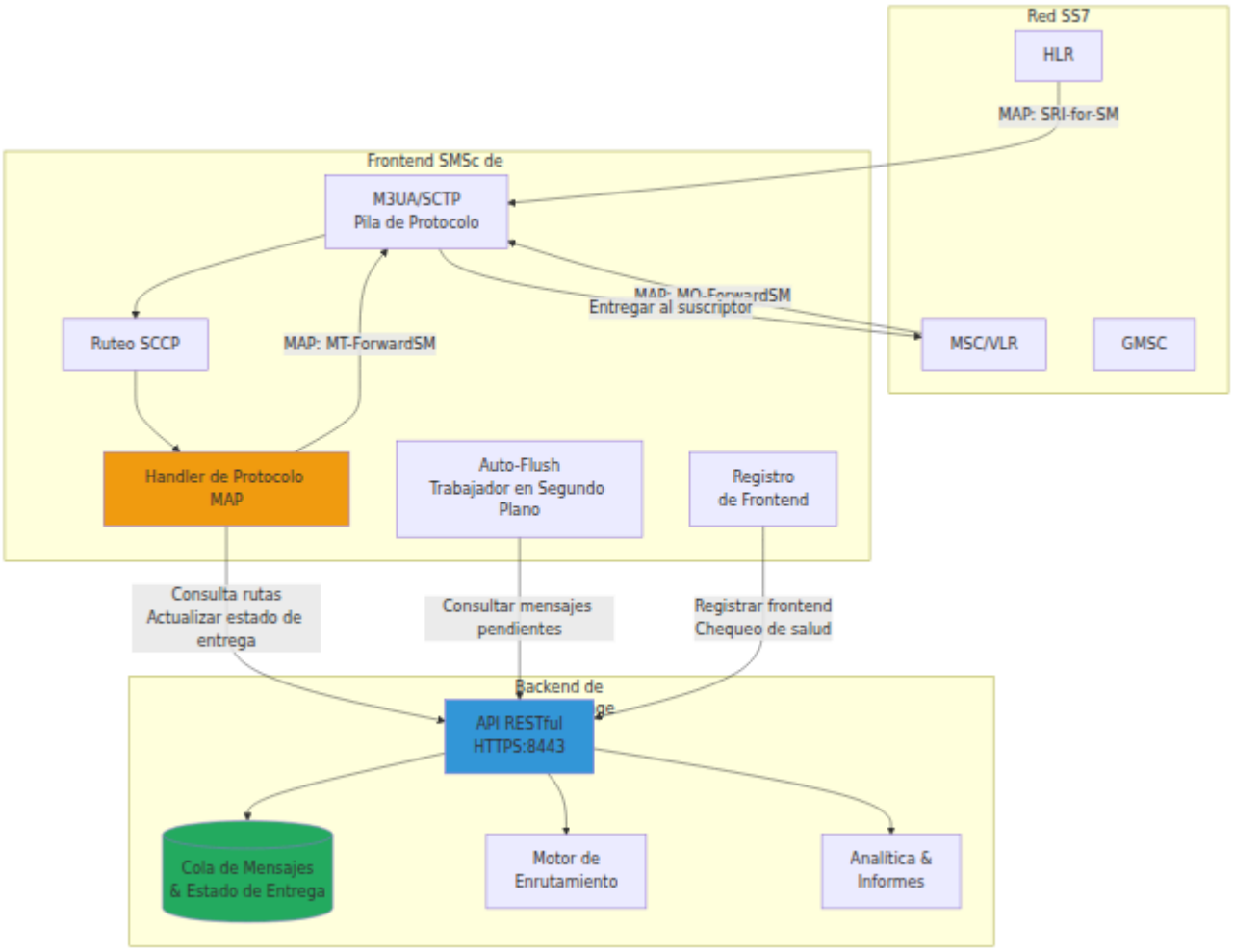
¿Qué es el Modo Centro de SMS?

Nota: Esta sección cubre únicamente la configuración de señalización SS7 de OmniSS7. Para reglas de enrutamiento de mensajes, gestión de colas, seguimiento de entrega y configuración de lógica de negocio, consulte la **documentación del producto OmniMessage**.

Modo Centro de SMS permite que OmniSS7 funcione como un SMSc para:

- **Entrega MT-SMS**: Entrega de SMS Terminados a Móviles a suscriptores
- **Manejo de MO-SMS**: Recepción y enrutamiento de SMS Originados en Móviles
- **Encolado de Mensajes**: Cola de mensajes respaldada por base de datos con lógica de reintentos
- **Auto-Flush**: Entrega automática de SMS desde la cola
- **Informes de Entrega**: Seguimiento del estado de entrega de mensajes

Arquitectura del Centro de SMS



Habilitando el Modo SMSc

OmniSS7 puede operar en diferentes modos. Para usarlo como un SMSc, necesita habilitar el modo SMSc en la configuración.

Cambiando al Modo SMSc

El archivo `config/runtime.exs` de OmniSS7 contiene tres modos operativos preconfigurados. Para habilitar el modo SMSc:

1. **Abrir** `config/runtime.exs`
2. **Encontrar** las tres secciones de configuración (líneas 53-204):
   - Configuración 1: Modo STP (líneas 53-95)

    - Configuración 2: Modo HLR (líneas 97-142)
    - Configuración 3: Modo SMSc (líneas 144-204)
3. **Comentar** cualquier otra configuración activa (agregar `#` a cada línea)
4. **Descomentar** la configuración SMSc (eliminar `#` de las líneas 144-204)
5. **Personalizar** los parámetros de configuración según sea necesario
6. **Reiniciar** la aplicación: `iex -S mix`

Configuración del Modo SMSc

La configuración completa de SMSc se ve así:

```
config :omniss7,
  # Banderas de modo - Habilitar características STP + SMSc
  # Nota: map_client_enabled es verdadero porque SMSc necesita
capacidades de enrutamiento
  map_client_enabled: true,
  hlr_mode_enabled: false,
  smsc_mode_enabled: true,

  # Configuración de la API del Backend de OmniMessage
  smsc_api_base_url: "https://10.179.3.219:8443",
  # Identificación del SMSc para registro con el backend
  smsc_name: "ipsmgw",
  # Dirección GT del Centro de Servicio para operaciones SMS
  smsc_service_center_gt_address: "5551234567",

  # Configuración de Auto Flush (procesamiento de cola de SMS en
segundo plano)
  auto_flush_enabled: true,
  auto_flush_interval: 10_000,
  auto_flush_dest_smsc: "ipsmgw",
  auto_flush_tps: 10,

  # Configuración de Conexión M3UA
  # Conectar como ASP para enviar/recibir operaciones MAP SMS
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :stp_client_asp,
    # Endpoint local (sistema SMSc)
    local_ip: {10, 179, 4, 12},
    local_port: 2905,
    # Endpoint remoto STP
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1
  }

config :control_panel,
  use_additional_pages: [
    {SS7.Web.EventsLive, "/events", "Eventos SS7"},
    {SS7.Web.TestClientLive, "/client", "Cliente SS7"},
    {SS7.Web.M3UAStatusLive, "/m3ua", "M3UA"},
    {SS7.Web.RoutingLive, "/routing", "Enrutamiento"},
```

```
    {SS7.Web.RoutingTestLive, "/routing_test", "Prueba de
Enrutamiento"},
    {SS7.Web.SmscLinksLive, "/smsc_links", "Enlaces SMSc"}
  ],
  page_order: ["/events", "/client", "/m3ua", "/routing",
"/routing_test", "/smsc_links", "/application", "/configuration"]
```

Parámetros de Configuración a Personalizar

Para una referencia completa de todos los parámetros de configuración, consulte la Referencia de Configuración.

Parámetro	Tipo	Predeterminado	D
`smsc_api_base_url`	Cadena	*Requerido*	**Endpo** backer OmniM
`smsc_name`	Cadena	`"{hostname}_SMSc"`	Su ide para re
`smsc_service_center_gt_address`	Cadena	*Requerido*	Título de Ser
`auto_flush_enabled`	Booleano	`true`	Habilit autom
`auto_flush_interval`	Entero	`10_000`	Interva proces en mili
`auto_flush_dest_smsc`	Cadena	*Requerido*	Nombr destin
`auto_flush_tps`	Entero	`10`	Tasa d de me (transa
`local_ip`	Tupla	*Requerido*	Direcci sistem
`local_port`	Entero	`2905`	Puerto
`remote_ip`	Tupla	*Requerido*	Direcci conect
`remote_port`	Entero	`2905`	Puerto

Parámetro	Tipo	Predeterminado	D
`routing_context`	Entero	`1`	ID de c enruta

¿Qué Ocurre Cuando se Habilita el Modo SMSc?

Cuando `smsc_mode_enabled: true` y `map_client_enabled: true`, la interfaz web mostrará:

- **Eventos SS7** - Registro de eventos
- **Cliente SS7** - Pruebas de operaciones MAP
- **M3UA** - Estado de conexión
- **Enrutamiento** - Gestión de tabla de rutas (STP habilitado)
- **Prueba de Enrutamiento** - Pruebas de ruta (STP habilitado)
- **Enlaces SMSc** - Estado de la API SMSc + gestión de cola de SMS ← *Específico de SMSc*
- **Recursos** - Monitoreo del sistema
- **Configuración** - Visor de configuración

La pestaña **Enlaces HLR** estará oculta.

Notas Importantes

- El modo SMSc requiere `map_client_enabled: true` para capacidades de enrutamiento
- **Backend de OmniMessage**: La API de OmniMessage debe ser accesible en la `smsc_api_base_url` configurada
- **Registro de Frontend**: El sistema se registra automáticamente con OmniMessage cada **5 minutos** a través del módulo `SMS.FrontendRegistry`
- **Tiempo de Espera de Solicitud API**: Todas las solicitudes API de OmniMessage tienen un **tiempo de espera de 5 segundos codificado**
- **Tiempo de Espera de Solicitud MAP**: SRI-for-SM tiene un tiempo de espera de diálogo de 10 segundos. MT-ForwardSM tiene un **tiempo de**

espera de diálogo de 30 segundos para acomodar respuestas lentas de VLR en redes 2G/CS

- Auto-flush procesa automáticamente la cola de SMS en segundo plano
- La conexión M3UA al STP es necesaria para enviar/recibir operaciones MAP SMS
- Después de cambiar modos, debe reiniciar la aplicación para que los cambios tengan efecto
- **Interfaz Web**: Consulte la Guía de Interfaz Web para información sobre el uso de la interfaz web
- **Acceso a la API**: Consulte la Guía de API para documentación de la API REST y acceso a Swagger UI

Configuración de la API HTTP

Configuración del Backend de OmniMessage

OmniSS7 se comunica con OmniMessage a través de la API REST HTTPS para gestionar la entrega de mensajes, rastrear el estado del suscriptor y registrarse como un frontend activo:

```
config :omniss7,
  # URL base de la API de OmniMessage
  smsc_api_base_url: "https://10.5.198.200:8443",
  # Identificador del nombre del SMSc para registro
(predeterminado a hostname_SMSc si está vacío)
  smsc_name: "omni-smsc01",
  # Dirección GT del Centro de Servicio para operaciones SMS
  smsc_service_center_gt_address: "5551234567"
```

Parámetros de Configuración:

Parámetro	Tipo	Requerido	Predeterr
`smsc_api_base_url`	Cadena	Sí	`"https://loca`
`smsc_name`	Cadena	No	`""` (usa `" {hostname}_SM`
`smsc_service_center_gt_address`	Cadena	No	`"5551234567"`

Registro de Frontend

El sistema se registra automáticamente con OmniMessage al inicio y **se vuelve a registrar cada 5 minutos** a través del módulo `SMS.FrontendRegistry`. Esto permite a OmniMessage:

- Rastrear frontends activos para balanceo de carga
- Monitorear tiempo de actividad y estado de salud

- Recopilar información de configuración
- Gestionar el enrutamiento SMS distribuido a través de múltiples frontends

Detalles de Implementación:

- **Intervalo de Registro**: 5 minutos (codificado)
- **Proceso**: Iniciado automáticamente cuando `smsc_mode_enabled: true`

Carga Útil de Registro:

```
{
  "frontend_name": "omni-smsc01",
  "configuration": "{...}",
  "frontend_type": "SS7",
  "hostname": "smsc-server01",
  "uptime_seconds": 12345
}
```

Nota: El nombre del frontend se toma del parámetro de configuración `smsc_name`. Si no se establece, se predetermina a `"{hostname}_SMSc"`.

Comunicación de la API de OmniMessage

Cuando OmniSS7 recibe operaciones MAP de la red SS7 o procesa la cola de mensajes, se comunica con OmniMessage para:

- **Registrarse como un frontend activo** y reportar estado de salud
- **Enviar mensajes móviles originados (MO)** recibidos de suscriptores
- **Recuperar mensajes móviles terminados (MT)** de la cola para entrega
- **Actualizar estado de entrega** con informes de éxito/fallo
- **Consultar información de enrutamiento** para el reenvío de mensajes

Endpoint	Método	Propósito	Cuerpo de Solicitud
`/api/frontends`	POST	Registrar instancia de frontend	`{"frontend_name": "...", "frontend_type": "SMSc", "hostname": "...", "uptime_seconds": ...}`
`/api/messages_raw`	POST	Insertar nuevo mensaje SMS	`{"source_msisdn": "...", "source_smsc": "...", "message_body": "..."}`
`/api/messages`	GET	Obtener cola de mensajes	Encabezado: `smsc: <smsc_name>`
`/api/messages/{id}`	PATCH	Marcar mensaje como entregado	`{"deliver_time": "...", "dest_smsc": "..."}`
`/api/messages/{id}`	PUT	Actualizar estado del mensaje	`{"dest_smsc": null}`
`/api/locations`	POST	Insertar/actualizar ubicación del suscriptor	`{"msisdn": "...", "imsi": "...", "location": "...", "ims_capable": true, "csfb": false, "expires":`

Endpoint	Método	Propósito	Cuerpo de Solicitud
			`"...", "user_agent": "...", "ran_location": "...", "imei": "...", "registered": "..."}`
`/api/events`	POST	Añadir seguimiento de eventos	`{"message_id": ..., "name": "...", "description": "..."}`
`/api/status`	GET	Chequeo de salud	-

Formato de Respuesta de la API

Todas las respuestas de la API utilizan formato JSON con las siguientes convenciones:

- **Respuestas de éxito**: HTTP 200-201 con cuerpo JSON que contiene datos de resultado
- **Respuestas de error**: HTTP 4xx/5xx con detalles de error en el cuerpo de respuesta
- **Tiempos**: Formato ISO 8601 (por ejemplo, `"2025-10-21T12:34:56Z"`)
- **IDs de Mensaje**: Identificadores enteros o de cadena

Módulos de Cliente de API

El sistema SMS consta de tres módulos principales:

1. SMSc.APIClient

Módulo principal del cliente API que proporciona toda la comunicación HTTP API con OmniMessage:

- `frontend_register/4` - Registrar frontend con OmniMessage
- `insert_message/3` - Insertar mensaje SMS en bruto (versión de 3 parámetros compatible con Python)
- `insert_location/9` - Insertar/actualizar datos de ubicación del suscriptor
- `get_message_queue/2` - Recuperar mensajes pendientes de la cola
- `mark_dest_smsc/3` - Marcar mensaje como entregado o fallido
- `add_event/3` - Añadir seguimiento de eventos para mensajes
- `flush_queue/2` - Procesar mensajes pendientes (SRI-for-SM + MT-forwardSM)
- `auto_flush/2` - Bucle de procesamiento continuo de cola

2. SMS.FrontendRegistry

Maneja el registro periódico del frontend con el backend:

- Se registra automáticamente al inicio
- Se vuelve a registrar cada 5 minutos
- Usa `smsc_name` de la configuración (retrocede al nombre de host)
- Recopila información de configuración y tiempo de actividad del sistema

3. SMS.Utils

Funciones utilitarias para operaciones SMS:

- `generate_tp_scts/0` - Generar marca de tiempo SMS en formato TPDU

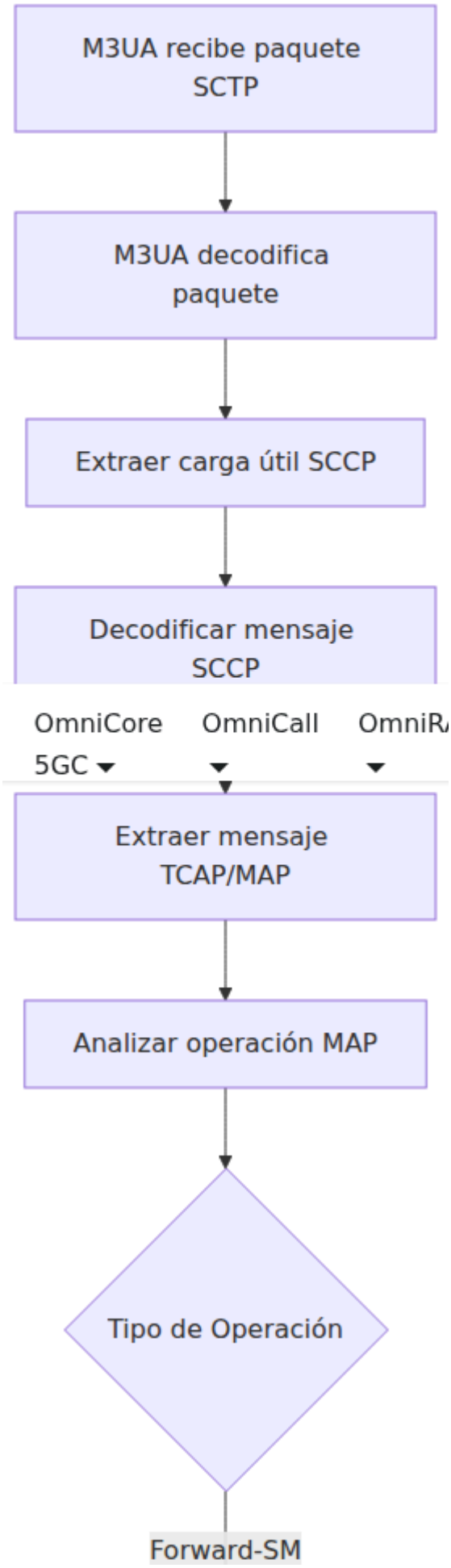
Flujos de Mensajes SMS

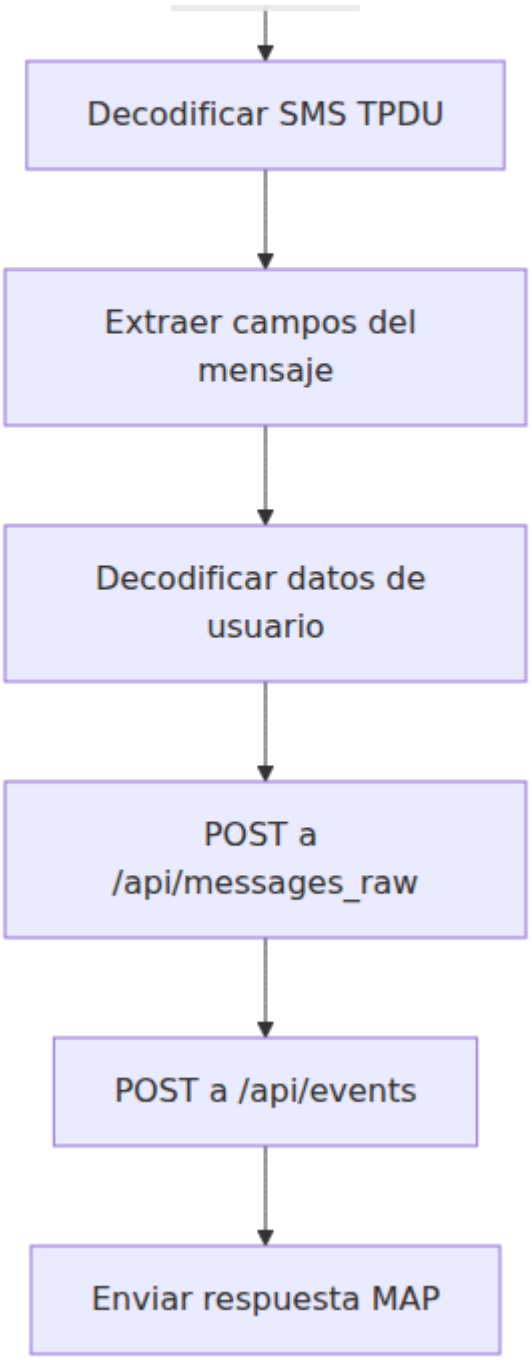
Flujo de SMS Entrante (Originado en Móvil)


M3UA recibe paquete SCTP
M3UA decodifica paquete
Extraer carga útil SCCP
Decodificar mensaje SCCP
Extraer mensaje TCAP/MAP
Analizar operación MAP
Tipo de Operación
Forward-SM

Flujo de SMS Saliente (Terminado a Móvil)

Cuando se configura una API HSS/HLR (`hlr_api_base_url`), el IP-SM-GW puede entregar SMS MT a **suscriptores en red** sin un viaje de ida y vuelta SRI-for-SM consultando directamente la API HSS para obtener el IMSI real del suscriptor y la dirección VLR de servicio. Una **caché VLR** en el Seguimiento de Suscriptores evita búsquedas repetidas en HSS para el mismo suscriptor.


M3UA recibe paquete SCTP
M3UA decodifica paquete
Extraer carga útil SCCP
Decodificar mensaje SCCP
OmniCore 5GC
OmniCall
OmniR
Extraer mensaje TCAP/MAP
Analizar operación MAP
Tipo de Operación
Forward-SM



Pasos Clave Explicados:

- **Chequeo de Caché VLR**: Antes de consultar la API HSS, el sistema verifica la caché VLR del Rastreador de Suscriptores para una entrada válida (IMSI + dirección VLR dentro de TTL). Un hit en caché omite completamente la API HSS. Consulte Caché de Direcciones VLR para más detalles.

- **Entrega Directa de API HSS (en red)**: Cuando se configura `hlr_api_base_url`, el IP-SM-GW consulta la API HSS con el MSISDN de destino para obtener el **IMSI real** del suscriptor y la **dirección VLR de**

servicio. Si ambos están disponibles, el MT-ForwardSM se envía directamente al VLR, omitiendo completamente SRI-for-SM.

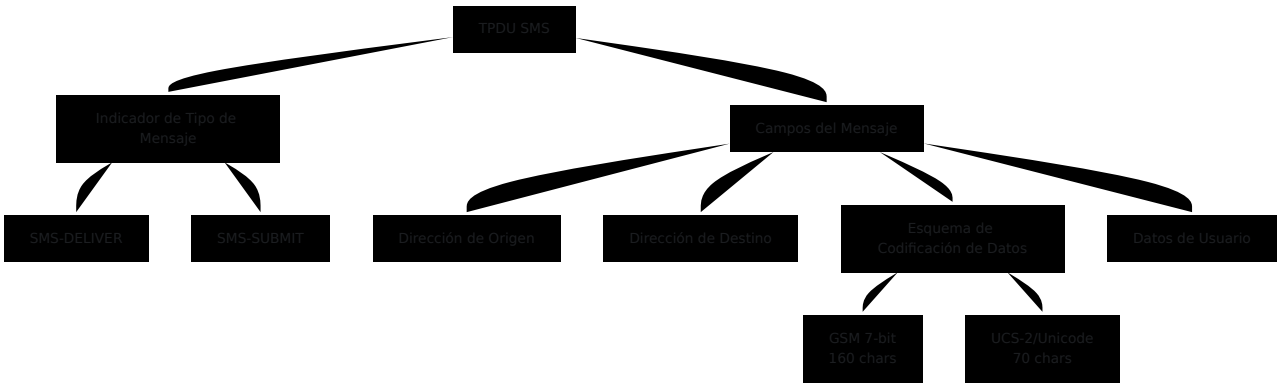
- **Suscriptor Ausente (en red)**: Si la API HSS encuentra al suscriptor pero no se le asigna un VLR de servicio, el mensaje se marca como fallido con una entrada de ubicación expirada. Se reintenta la entrega cuando el HLR envía un alertServiceCenter después de que el suscriptor se vuelve a registrar.

- **Fallback SRI-for-SM (fuera de red)**: Cuando la API HSS no está configurada o el suscriptor no se encuentra en el HSS, se utiliza el flujo estándar SRI-for-SM. El HLR responde con un IMSI sintético y un número de nodo de red. Consulte SRI-for-SM en la Guía HLR.

- **Limpiar VLR en Caso de Fallo**: Cuando la entrega de MT-ForwardSM falla (tiempo de espera, error SCCP, etc.), se limpia la dirección VLR en caché para que el próximo intento vuelva a consultar la API HSS en busca de un nuevo VLR.

- **Tiempo de Espera de MT-ForwardSM**: El diálogo MT-ForwardSM tiene un **tiempo de espera de 30 segundos** para acomodar respuestas más lentas de VLR en 2G/CS.

Estructura TPDU SMS



Manejo del Centro de Servicio de Alerta

El SMSc puede recibir mensajes **alertServiceCenter** del HLR para rastrear el estado de alcanzabilidad del suscriptor.

Para información sobre cómo el HLR envía mensajes alertServiceCenter, consulte Integración del Centro de Servicio de Alerta en la Guía HLR.

¿Qué es alertServiceCenter?

Cuando un suscriptor realiza un UpdateLocation en el HLR (es decir, se registra con un nuevo VLR/MSC), el HLR puede notificar a los sistemas SMSc que el suscriptor ahora es alcanzable enviando un mensaje **alertServiceCenter** (código de operación MAP 64).

Configuración

El tiempo de expiración de la ubicación se configura en el HLR:

```
config :omniss7,
  # Tiempo de expiración de la ubicación cuando SMSc recibe
alertServiceCenter (predeterminado: 48 horas)
  hlr_alert_location_expiry_seconds: 172800
```

Comportamiento

Cuando el SMSc recibe un mensaje alertServiceCenter:

1. **Decodificar MSISDN**: Extraer el MSISDN del suscriptor del mensaje (formato TBCD)
2. **Eliminar prefijo TON/NPI**: Quitar prefijos comunes como "19", "11", "91" (por ejemplo, "19123123213" → "123123213")
3. **Resolver IMSI**: Si `hlr_api_base_url` está configurado, consultar la API HSS para resolver el **IMSI real** del suscriptor. Si la búsqueda en HSS falla o no se configura ninguna API, retroceder a generar un IMSI sintético utilizando el mismo mapeo que SRI-for-SM. Usar el IMSI real es crítico para la entrega MT a suscriptores 2G/CS en red: el MSC/VLR requiere el IMSI correcto en el MT-ForwardSM.
4. **POST a /api/location**: Actualizar la base de datos de ubicación con:
   - `msisdn`: Número de teléfono del suscriptor (limpiado)
   - `imsi`: IMSI sintético
   - `location`: Nombre del SMSc (por ejemplo, "ipsmgw")
   - `expires`: Hora actual + `hlr_alert_location_expiry_seconds`
   - `csfb`: true (suscriptor alcanzable a través de Circuit-Switched Fallback)
   - `ims_capable`: false (esto es registro 2G/3G CS, no IMS/VoLTE)
   - `user_agent`: GT del HLR que envió la alerta (para seguimiento)
   - `ran_location`: "SS7"
5. **Rastrear en el Rastreador de Suscriptores SMSc**: Registrar al suscriptor con GT de HLR, estado=activo, contadores de mensajes en 0
6. **Enviar ACK**: Responder al HLR con el reconocimiento de alertServiceCenter

Manejo de Suscriptores Ausentes

Cuando el SMSc intenta entregar un mensaje y recibe un error de "suscriptor ausente" durante SRI-for-SM (para más información sobre SRI-for-SM, consulte SRI-for-SM en la Guía HLR):

1. **Detectar ausencia**: SRI-for-SM devuelve el error `absentSubscriberDiagnosticSM`

2. **Expirar ubicación**: POST a /api/location con `expires=0` para marcar al suscriptor como no alcanzable
3. **Agente de usuario**: Establecer en "SS7_AbsentSubscriber" para identificar la fuente
4. **Actualizar rastreador**: Marcar al suscriptor como `fallido` en el Rastreador de Suscriptores SMSc

Esto asegura que la base de datos de ubicación y el rastreador reflejen con precisión el estado de alcanzabilidad del suscriptor.

Diagrama de Flujo

Parse error on line 7: ... Nota sobre HLR,SMSc: El suscriptor ----------^
Expecting 'SOLID_OPEN_ARROW', 'DOTTED_OPEN_ARROW', 'SOLID_ARROW', 'BIDIRECTIONAL_SOLID_ARROW', 'DOTTED_ARROW', 'BIDIRECTIONAL_DOTTED_ARROW', 'SOLID_CROSS', 'DOTTED_CROSS', 'SOLID_POINT', 'DOTTED_POINT', got ','

Intente de nuevo

Endpoint de API

POST /api/location

```
{
  "msisdn": "15551234567",
  "imsi": "001010123456789",
  "location": "ipsmgw",
  "ims_capable": false,
  "csfb": true,
  "expires": "2025-11-01T12:00:00Z",
  "user_agent": "15551111111",
  "ran_location": "SS7",
  "imei": "",
  "registered": "2025-10-30T12:00:00Z"
}
```

Nota: El campo `user_agent` contiene el GT del HLR que envió el alertServiceCenter, permitiendo al SMSc rastrear qué HLR está proporcionando actualizaciones de ubicación.

Para suscriptores ausentes, `expires` se establece en la hora actual (expiración inmediata).

Prevención de Bucles

El SMSc implementa **prevención automática de bucles** para evitar bucles infinitos de enrutamiento de mensajes cuando los mensajes se originan en redes SS7, mientras que aún permite la entrega legítima MO→MT entre suscriptores en red.

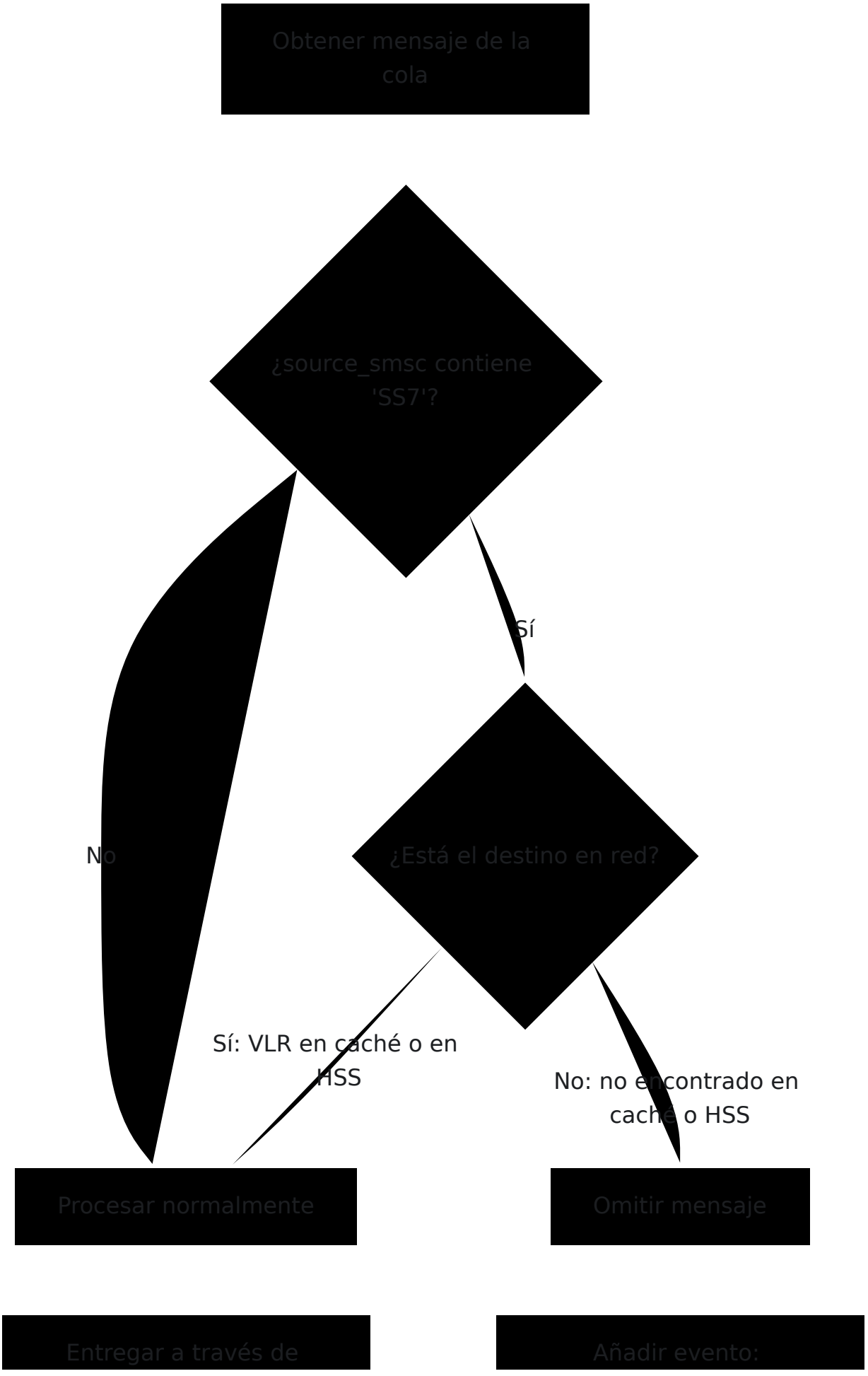
¿Por qué es Importante la Prevención de Bucles?

Cuando el SMSc recibe mensajes SMS originados en móviles (MO) de la red SS7, los inserta en la cola de mensajes con un campo `source_smsc` que identifica su origen (por ejemplo, `"SS7_MO_15551234567"`). Sin prevención de bucles, estos mensajes podrían ser:

1. Recibidos de la red SS7 → Encolados con `source_smsc` conteniendo "SS7"
2. Recuperados de la cola → Procesados para entrega
3. Enviados de vuelta a la red SS7 → Creando un bucle

Cómo Funciona

La lógica de prevención de bucles distingue entre destinos **en red** y **fuera de red**. Los mensajes de SS7 solo se bloquean cuando el suscriptor de destino **no está en red** — esto previene bucles de enrutamiento en tránsito mientras permite la entrega legítima MO→MT entre suscriptores en la misma red (por ejemplo, MSC a MSC a través del IP-SM-GW).


Obtener mensaje de la cola
¿source_smsc contiene 'SS7'?
Sí
No
¿Está el destino en red?
Sí: VLR en caché o en HSS
No: no encontrado en caché o HSS
Procesar normalmente
Omitir mensaje
Entregar a través de
Añadir evento:

Detección en Red

Cuando un mensaje se origina en SS7, el sistema verifica si el suscriptor de destino está en red antes de aplicar la prevención de bucles:

1. **Caché VLR** (más rápido): Verificar si el suscriptor tiene una dirección VLR en caché con IMSI válido en el Rastreador de Suscriptores
2. **API HSS** (autoritativa): Si `hlr_api_base_url` está configurado y la caché falla, consultar la API HSS por MSISDN. Si el suscriptor existe en el HSS, está en red.

Si ninguna de las verificaciones encuentra al suscriptor, el destino se considera **fuera de red** y se bloquea el mensaje.

Implementación

Al procesar mensajes de la cola, el SMSc verifica el campo `source_smsc`:

- **Si `source_smsc` contiene "SS7" y el destino está fuera de red**:
  - El mensaje se omite
  - Se añade un evento: `"Prevención de Bucles"` con una descripción que indica que el destino está fuera de red
  - El mensaje se marca como fallido a través de una solicitud PUT
  - Registrado con nivel de advertencia
- **Si `source_smsc` contiene "SS7" y el destino está en red**:
  - El mensaje se procesa normalmente — este es un flujo legítimo MO→MT
  - La entrega procede a través de la API HSS / ruta de caché VLR
- **Si `source_smsc` no contiene "SS7"**:

- El mensaje se procesa normalmente (por ejemplo, mensajes de la API web, SMPP u otras fuentes no SS7)

Valores de SMSC de Origen

Los mensajes pueden tener varios valores de `source_smsc`:

Origen	Valor de Ejemplo	Acción
Red SS7 (MO-FSM), destino fuera de red	`"SS7_MO_15551234567"`	**Omitido** - Prevención de bucles
Red SS7 (MO-FSM), destino en red	`"SS7_MO_15551234567"`	**Permitido** - Entrega en red MO→MT
API Externa/SMPP	`"ipsmgw"` o `"api_gateway"`	Procesado normalmente
Otro SMSc	`"smsc-node-01"`	Procesado normalmente

Seguimiento de Eventos

Cuando un mensaje se omite debido a la prevención de bucles, se registra un evento:

```
{
  "message_id": 12345,
  "name": "Prevención de Bucles",
  "description": "Mensaje omitido - source_smsc
'SS7_MO_15551234567' contiene 'SS7' y el destino está fuera de
red, previniendo el bucle de mensajes"
}
```

Este evento es visible en:

- **Interfaz Web**: Página de Eventos SS7 (`/events`)

- **Base de Datos**: Tabla `events` a través de la API
- **Registros**: Entradas de registro de nivel de advertencia

Configuración

La prevención de bucles está **siempre habilitada** y no puede ser deshabilitada. Esta es una característica de seguridad crítica para prevenir la interrupción de la red por bucles de mensajes. El bypass en red requiere tanto `smsc_mode_enabled: true` como que la caché VLR o `hlr_api_base_url` estén configuradas.

Ejemplos de Escenarios

Escenario 1: MO-SMS de SS7 a suscriptor fuera de red (bloqueado)

```
1. Teléfono móvil → MSC/VLR → IP-SM-GW (a través de MO-ForwardSM)
2. IP-SM-GW recibe MO-FSM, inserta en cola: source_smsc =
"SS7_MO_2471900"
3. Auto-flush recupera el mensaje de la cola
4. Destino no en caché VLR o HSS → fuera de red
5. IP-SM-GW detecta "SS7" en source_smsc + destino fuera de red →
OMITE
6. Evento registrado: "Prevención de Bucles"
7. No se envía SRI-for-SM ni MT-ForwardSM (bucle prevenido)
```

Escenario 2: MO-SMS de SS7 a suscriptor en red (permitido)

```
1. Teléfono móvil → MSC/VLR → IP-SM-GW (a través de MO-ForwardSM)
2. IP-SM-GW recibe MO-FSM, inserta en cola: source_smsc =
"SS7_MO_2471900"
3. Auto-flush recupera el mensaje de la cola
4. Destino encontrado en API HSS → suscriptor en red
5. IP-SM-GW permite la entrega a pesar del origen SS7
6. HSS devuelve IMSI real + VLR → MT-ForwardSM enviado
directamente al VLR
```

Seguimiento de Suscriptores SMSc

El SMSc incluye un GenServer de **Rastreador de Suscriptores** que mantiene el estado en tiempo real para los suscriptores basado en mensajes alertServiceCenter y en intentos de entrega de mensajes.

Propósito

El rastreador proporciona:

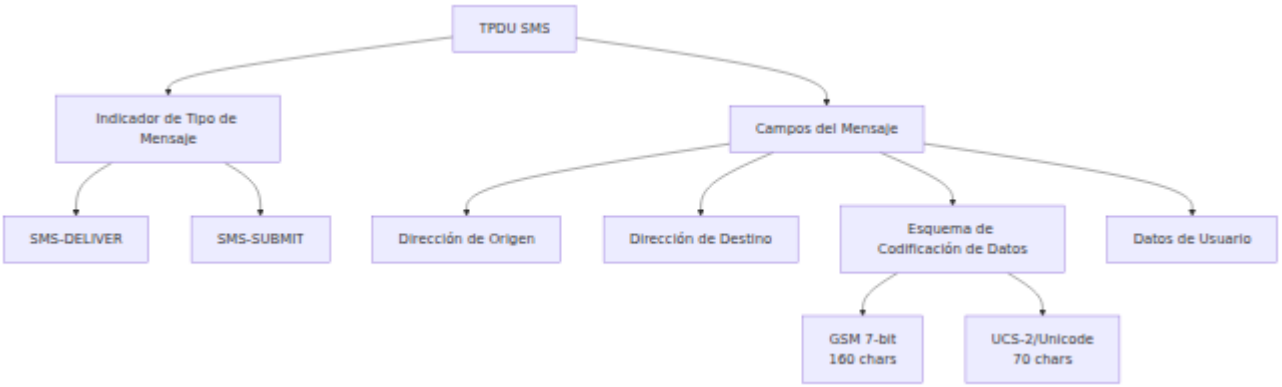
- **Monitoreo de alcanzabilidad**: Qué suscriptores son actualmente alcanzables
- **Rastreo de HLR**: Qué HLR envió el alertServiceCenter para cada suscriptor
- **Contadores de mensajes**: Número de mensajes enviados/recibidos por suscriptor
- **Rastreo de fallos**: Marcar a los suscriptores como fallidos cuando los intentos de entrega fallan
- **Visibilidad en la Interfaz Web**: Panel de control en tiempo real que muestra todos los suscriptores rastreados

Información Rastreada

Para cada suscriptor, el rastreador almacena:

Campo	Descripción	Ejemplo
`msisdn`	Número de teléfono del suscriptor (clave)	"15551234567"
`imsi`	IMSI del suscriptor (real de HSS si está disponible)	"001010123456789"
`hlr_gt`	GT de HLR que envió alertServiceCenter	"15551111111"
`messages_sent`	Conteo de mensajes MT-FSM enviados	5
`messages_received`	Conteo de mensajes MO-FSM recibidos	2
`status`	`:active` o `:failed`	`:active`
`updated_at`	Marca de tiempo Unix de la última actualización	1730246400
`vlr_address`	Dirección VLR de servicio en caché	"14155550100"
`vlr_cached_at`	Marca de tiempo Unix cuando se cacheó la VLR por última vez	1730246400

Transiciones de Estado



Comportamiento

Cuando se recibe alertServiceCenter:

- Crear o actualizar la entrada del suscriptor
- Establecer `status = :active`
- Registrar GT de HLR
- Resolver IMSI real de la API HSS (si está configurada), de lo contrario usar IMSI sintético
- Preservar caché VLR existente y contadores de mensajes

Cuando la API HSS devuelve el suscriptor con VLR:

- Cachear la dirección VLR y el IMSI real en el rastreador
- Las entregas subsiguientes utilizan la VLR en caché sin volver a consultar el HSS

Cuando se recibe MO-ForwardSM:

- Cachear el GT de llamada SCCP como la dirección VLR del suscriptor (el GT de llamada en un MO-ForwardSM es el VLR que retransmitió el mensaje)
- Crear una entrada en el rastreador si no existe

Cuando falla la entrega de MT-ForwardSM:

- Limpiar la dirección VLR en caché para que el próximo intento de entrega vuelva a consultar la API HSS

Cuando SRI-for-SM tiene éxito:

- Incrementar el contador `messages_sent`
- Actualizar la marca de tiempo `updated_at`

Cuando SRI-for-SM falla:

- Establecer `status = :failed`
- Mantener en el rastreador para monitoreo

Cuando se elimina el suscriptor:

- Eliminar de la tabla ETS
- Ya no aparece en la Interfaz Web

Interfaz Web - Página de Suscriptores SMSc

Ruta: `/smsc_subscribers` **Auto-refresco**: Cada 2 segundos

Nota: Esta página solo está disponible cuando se ejecuta en modo SMSc. Después de descomentar la configuración SMSc en `config/runtime.exs`, debe reiniciar la aplicación para que la ruta esté disponible.

La página de **Suscriptores SMSc** proporciona monitoreo en tiempo real de todos los suscriptores rastreados:

Características

1. **Tabla de Suscriptores**
    - MSISDN, IMSI, GT de HLR, GT de VLR (dirección VLR de servicio en caché)
    - Contadores de mensajes enviados/recibidos
    - Insignia de estado (Activo/Fallido) con codificación de colores
    - Marca de tiempo de última actualización y duración
    - Botón de eliminación para suscriptores individuales
2. **Estadísticas Resumidas**

- Total de suscriptores rastreados
- Conteo de suscriptores activos
- Conteo de suscriptores fallidos
- Número de HLR únicos

3. **Acciones**

- Limpiar Todo: Eliminar todos los suscriptores rastreados
- Eliminar: Eliminar suscriptor individual

Ejemplo de Vista

```
┌──────────────────────────────────────────────────────────────────────────────
│ Suscriptores Rastreado SMSc                                        Total:
├──────────────────────────────────────────────────────────────────────────────
│ MSISDN          IMSI              HLR GT         VLR GT         Msg
│                                                                   S/
├──  ──────────────────────────────────────────────────────────────────────────
│ 15551234567     001010123456789   15551111111    14155550100    5/2
│ 15559876543     001010987654321   15551111111    -              0/0
│ 15551112222     001010111222233   15552222222    14155550200    3/1
└──────────────────────────────────────────────────────────────────────────────

Resumen:  Total: 3  |  Activo: 2  |  Fallido: 1  |  HLRs Únicos: 2
```

Funciones de API

El rastreador expone estas funciones para acceso programático:

```
# Llamado cuando se recibe alertServiceCenter
SMSc.SubscriberTracker.alert_received(msisdn, imsi, hlr_gt)

# Incrementar contadores de mensajes
SMSc.SubscriberTracker.message_sent(msisdn)
SMSc.SubscriberTracker.message_received(msisdn)

# Marcar como fallido (falla de SRI-for-SM)
SMSc.SubscriberTracker.mark_failed(msisdn)

# Eliminar del seguimiento
SMSc.SubscriberTracker.remove_subscriber(msisdn)

# Funciones de consulta
SMSc.SubscriberTracker.get_active_subscribers()
SMSc.SubscriberTracker.get_subscriber(msisdn)
SMSc.SubscriberTracker.count_subscribers()
SMSc.SubscriberTracker.clear_all()
```

Integración

El rastreador se integra automáticamente con:

- **Manejador alertServiceCenter**: Llama a `alert_received/3` en la actualización de ubicación exitosa
- **Manejador SRI-for-SM**: Incrementa `messages_sent` en el enrutamiento exitoso
- **Manejo de suscriptor ausente**: Llama a `mark_failed/1` cuando el suscriptor está ausente
- **Errores de suscriptor desconocido**: Llama a `mark_failed/1` cuando falla SRI-for-SM

Caché de Direcciones VLR

Al operar como un IP-SM-GW con una API HSS configurada, el Rastreador de Suscriptores almacena en caché las direcciones VLR para evitar consultar el HSS en cada intento de entrega MT.

Cómo Funciona

La caché VLR se pobla de dos fuentes:

1. **Búsqueda en API HSS**: Cuando la tubería de entrega consulta la API HSS y obtiene un suscriptor con un VLR de servicio, el IMSI y la dirección VLR se almacenan en caché en el Rastreador de Suscriptores.
2. **MO-ForwardSM**: Cuando llega un MO-ForwardSM de un suscriptor, el GT de la parte llamante SCCP (que es el VLR/MSC que retransmitió el mensaje) se almacena en caché como la dirección VLR del suscriptor.

En los intentos de entrega MT subsiguientes, la caché se verifica primero. Si existe una entrada válida (tanto IMSI como VLR presentes, dentro de TTL), se omite completamente la API HSS.

Invalidación de Caché

La caché VLR se limpia cuando:

- **Falla la entrega de MT-ForwardSM** (tiempo de espera, error SCCP, rechazo de VLR) — la VLR en caché puede estar obsoleta
- **Expira TTL** — configurable a través de `vlr_cache_ttl_seconds`, las entradas más antiguas que el TTL se tratan como fallos de caché

La caché **no** se limpia en alertServiceCenter — la entrada VLR existente se preserva ya que el suscriptor puede seguir siendo alcanzable en el mismo VLR.

Configuración

```
config :omniss7,
  # Endpoint de API HSS/HLR (requerido para caché VLR y entrega en
red)
  hlr_api_base_url: "https://10.179.2.140:8443",

  # TTL de caché VLR en segundos (predeterminado: 3600 = 1 hora)
  vlr_cache_ttl_seconds: 3600
```

Parámetro	Tipo	Requerido	Predeterminado	Des
`hlr_api_base_url`	Cadena	No	`nil`	URL la AI HSS, Cuar esta habi búsc susc en r cach la er MT c Cuar `nil` las e utiliz flujo SM.
`vlr_cache_ttl_seconds`	Entero	No	`3600`	Edac máx segu para una VLR cach cons válic Desp este el pr inter entr vuel cons

Parámetro	Tipo	Requerido	Predeterminado	Des
				API I Valo bajo aum tráfi API I mejo prec para susc móv

Diagrama de Secuencia

Parse error on line 7: ... Nota sobre Queue: Entrega MT para MS... ----------^ Expecting 'SOLID_OPEN_ARROW', 'DOTTED_OPEN_ARROW', 'SOLID_ARROW', 'BIDIRECTIONAL_SOLID_ARROW', 'DOTTED_ARROW', 'BIDIRECTIONAL_DOTTED_ARROW', 'SOLID_CROSS', 'DOTTED_CROSS', 'SOLID_POINT', 'DOTTED_POINT', got 'TXT'

Intente de nuevo

Configuración de Auto-Flush de Cola de SMS

El servicio **Auto-Flush** procesa automáticamente los mensajes SMS pendientes.

Para referencia de parámetros de configuración, consulte Configuración de Auto-Flush en la Referencia de Configuración.

Configuración

```
config :omniss7,
  auto_flush_enabled: true,           # Habilitar/deshabilitar
auto-flush
  auto_flush_interval: 10_000,        # Intervalo de sondeo en
milisegundos
  auto_flush_dest_smsc: nil,          # Filtro: nil = todos
  auto_flush_tps: 10                  # Máximo de transacciones
por segundo
```

Cómo Funciona

1. **Sondeo**: Cada `auto_flush_interval` milisegundos, consulta la API para mensajes pendientes
2. **Filtrado**: Opcionalmente filtrar por `auto_flush_dest_smsc`
3. **Limitación de Tasa**: Procesar hasta `auto_flush_tps` mensajes por ciclo
4. **Entrega**: Para cada mensaje:
   - Enviar **SRI-for-SM** (Enviar Información de Enrutamiento para Mensaje Corto) al HLR para obtener información de enrutamiento
     - El HLR devuelve un IMSI sintético calculado a partir del MSISDN
     - El HLR devuelve la dirección GT del SMSc a donde se debe enviar MT-ForwardSM
     - Consulte Detalles de SRI-for-SM en la Guía HLR para documentación completa
   - En éxito, enviar **MT-forwardSM** al MSC/VLR
   - Actualizar el estado del mensaje a través de la API (entregado/fallido)
   - Añadir seguimiento de eventos a través de la API

> **Profundización Técnica**: Para una explicación completa de cómo funciona SRI-for-SM, incluyendo el mapeo de MSISDN a IMSI, la configuración de dirección del centro de servicio y la generación de IMSI sintético que preserva la privacidad, consulte la sección SRI-for-SM en la Guía de Configuración HLR.

Métricas SMSc

Métricas Disponibles

Métricas de Cola SMS:

- `smsc_queue_depth` - Número actual de mensajes pendientes
- `smsc_messages_delivered_total` - Total de mensajes entregados con éxito
- `smsc_messages_failed_total` - Total de mensajes que fallaron en la entrega
- `smsc_delivery_duration_milliseconds` - Histograma de tiempos de entrega

Consultas de Ejemplo:

```
# Profundidad de cola actual
smsc_queue_depth

# Tasa de éxito de entrega (últimos 5 minutos)
rate(smsc_messages_delivered_total[5m]) /
(rate(smsc_messages_delivered_total[5m]) +
rate(smsc_messages_failed_total[5m]))

# Tiempo promedio de entrega
rate(smsc_delivery_duration_milliseconds_sum[5m]) /
rate(smsc_delivery_duration_milliseconds_count[5m])
```

Solución de Problemas SMSc

Problema: Mensajes No Entregados

Verificaciones:

1. Verificar que auto-flush esté habilitado
2. Comprobar la conexión a la base de datos

3. Monitorear registros para errores
4. Verificar que la conexión M3UA esté ACTIVA
5. Comprobar límites TPS

Problema: Alta Profundidad de Cola

Causas Posibles:

- Límite TPS demasiado bajo
- Problemas de tiempo de espera en HLR
- Problemas de conectividad de red
- Números de destino inválidos

Soluciones:

- Aumentar `auto_flush_tps`
- Verificar disponibilidad de HLR
- Revisar registros de mensajes fallidos

API MT-forwardSM

Enviar SMS a través de la API

Endpoint de API: `POST /api/MT-forwardSM`

Solicitud:

```
{
  "imsi": "234509876543210",
  "destination_serviceCentre": "447999555111",
  "originating_serviceCenter": "447999123456",
  "smsPDU":
"040B917477218345F60000157030185714 0C0BD4F29C0E9281C4E1F11A"
}
```

Respuesta:

```
{
  "result": "success",
  "message_id": "12345"
}
```

Documentación Relacionada

Documentación de OmniSS7:

- ← Volver a la Documentación Principal
- Guía de Configuración HLR - Configuración y operaciones en modo HLR
  - **Detalles Técnicos de SRI-for-SM** - Documentación completa sobre el mapeo de MSISDN a IMSI y configuración del centro de servicio
- Guía de Características Comunes - Interfaz Web, API, Monitoreo
- Guía del Cliente MAP - Operaciones MAP
- Referencia Técnica - Especificaciones de protocolo

Documentación de OmniMessage: Para la configuración de enrutamiento de mensajes, gestión de colas, seguimiento de entrega, limitación de tasa y analítica, consulte la **documentación del producto OmniMessage**. OmniMessage contiene toda la lógica de enrutamiento de mensajes, algoritmos de reintentos de cola, manejo de informes de entrega y motor de reglas de negocio.

OmniSS7 por Omnitouch Network Services

Guía de Configuración de M3UA y M2PA STP

← Volver a la Documentación Principal

Esta guía proporciona una configuración detallada para usar OmniSS7 como un **Punto de Transferencia de Señalización (STP)**.

Tabla de Contenidos

¿Qué es un Punto de Transferencia de Señalización (STP)?

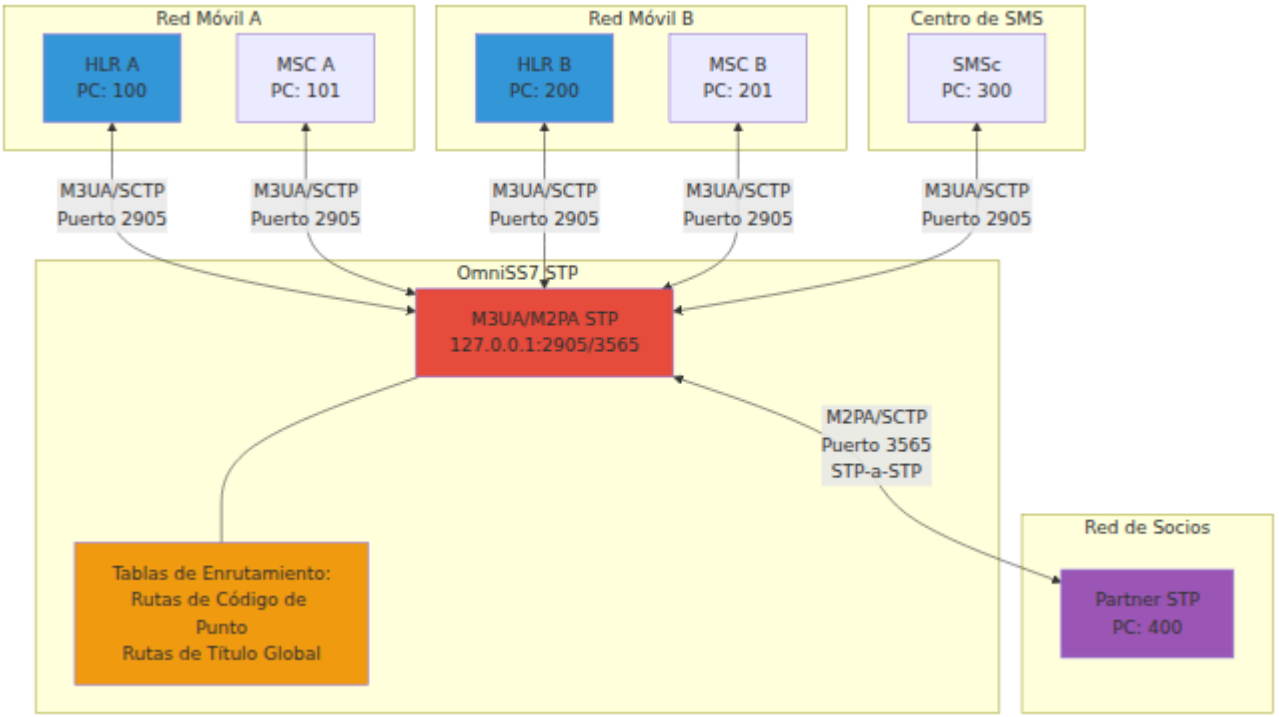
Un **Punto de Transferencia de Señalización (STP)** es un elemento crítico de la red en redes de señalización SS7 y basadas en IP que enruta mensajes de señalización entre nodos de la red.

Funciones del STP

- **Enrutamiento de Mensajes**: Rutea el tráfico de señalización SS7 basado en el Código de Punto (PC) de destino o Título Global (GT)
- **Traducción de Protocolo**: Conecta redes SS7 tradicionales con redes M3UA/SCTP basadas en IP
- **Distribución de Carga**: Distribuye el tráfico entre múltiples destinos utilizando enrutamiento basado en prioridades

- **Puerta de Enlace de Red**: Conecta diferentes redes de señalización y proveedores de servicios
- **Ocultamiento de Topología**: Puede reescribir direcciones para ocultar la topología interna de la red

Diagrama de Red STP



Roles de Red STP Explicados

ASP (Proceso de Servidor de Aplicación)

- **Rol**: Cliente que se conecta a un SGP/STP remoto
- **Dirección**: Conexión saliente
- **Caso de Uso**: Su STP se conecta al STP de una red asociada

SGP (Proceso de Puerta de Enlace de Señalización)

- **Rol**: Servidor que acepta conexiones de ASPs

- **Dirección**: Conexión entrante
- **Caso de Uso**: Redes asociadas se conectan a su STP

AS (Servidor de Aplicación)

- **Definición**: Agrupación lógica de uno o más ASPs
- **Propósito**: Proporciona redundancia y balanceo de carga
- **Caso de Uso**: Múltiples ASPs sirviendo el mismo destino

Interfaz con Redes TDM a través de Puertas de Enlace de Señalización

OmniSS7 es una plataforma de señalización basada en IP que utiliza protocolos SIGTRAN (M3UA/M2PA sobre SCTP). Para intercambiar señalización con **redes SS7 basadas en TDM** heredadas, necesita una **Puerta de Enlace de Señalización (SGW)** para unir los dos mundos.

¿Qué es una Puerta de Enlace de Señalización?

Una **Puerta de Enlace de Señalización (SGW)** es un elemento de red que convierte la señalización SS7 entre:

- **Lado TDM**: SS7 tradicional usando MTP1/MTP2/MTP3 sobre enlaces E1/T1
- **Lado IP**: SIGTRAN usando M3UA o M2PA sobre SCTP/IP

El SGW actúa como un traductor de protocolos, permitiendo que aplicaciones basadas en IP como OmniSS7 se comuniquen con equipos TDM heredados (conmutadores, STPs, HLRs) que no soportan señalización IP.

Arquitectura de TDM a IP

Comparación de Pilas de Protocolo

Capa	TDM SS7	Puerta de Enlace de Señalización	IP (SIGTRAN)
Parte de Usuario	SCCP/TCAP/MAP	← Transparente →	SCCP/TCAP/MAP
Red	MTP3	Conversión	M3UA/M2PA
Enlace	MTP2	Terminación	SCTP
Física	MTP1 (E1/T1)	Terminación	IP/Ethernet

El SGW termina MTP1/MTP2 en el lado TDM y presenta los datos de usuario MTP3 a través de M3UA o M2PA en el lado IP. Las capas superiores (SCCP, TCAP, MAP, CAP) pasan de manera transparente.

Conectando OmniSS7 a una Puerta de Enlace de Señalización

OmniSS7 se conecta a una Puerta de Enlace de Señalización como un **ASP (Proceso de Servidor de Aplicación)**, mientras que el SGW actúa como un **SGP (Proceso de Puerta de Enlace de Señalización)**.

Ejemplo de Configuración

```
config :omniss7,
  # Conectar a la Puerta de Enlace de Señalización como ASP
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :sgw_connection,
    # Punto final local (OmniSS7)
    local_ip: {10, 0, 0, 1},
    local_port: 0,                    # Puerto local dinámico
    # Punto final de la Puerta de Enlace de Señalización
    remote_ip: {10, 0, 0, 100},       # Dirección IP del SGW
    remote_port: 2905,                # Puerto M3UA del SGW
    routing_context: 1                # Asignado por el SGW
  },

  # O configurar como un par para enrutamiento STP
  peers: [
    %{
      peer_id: 1,
      name: "TDM_Gateway",
      role: :client,                  # OmniSS7 inicia la conexión
      local_ip: {10, 0, 0, 1},
      local_port: 0,
      remote_ip: {10, 0, 0, 100},     # Dirección IP del SGW
      remote_port: 2905,
      routing_context: 1,
      point_code: 100,                # Rango de código de punto
detrás del SGW
      network_indicator: :international
    }
  ],

  # Rutar códigos de punto de red TDM a través de la puerta de
enlace
  m3ua_routes: [
    %{
      dest_pc: 100,                   # Código de punto del MSC
TDM
      peer_id: 1,                     # Ruta a través del par SGW
      priority: 1,
      network_indicator: :international
    },
    %{
```

```
      dest_pc: 200,                    # Código de punto del HLR
TDM
      peer_id: 1,
      priority: 1,
      network_indicator: :international
    }
  ]
```

Consideraciones de Configuración del SGW

Al configurar su Puerta de Enlace de Señalización para trabajar con OmniSS7:

Parámetro	Descripción	Valor Típico
IP Remota	Dirección IP de OmniSS7	Dirección IP de su servidor
Puerto Remoto	Puerto SCTP de OmniSS7	2905
Contexto de Enrutamiento	Identificador AS en el SGW	Asignado por el administrador del SGW
Códigos de Punto	Códigos de punto de red TDM accesibles a través del SGW	Específico de la red
Modo de Tráfico	Modo de manejo de tráfico ASP	Override o Loadshare

Escenarios de Despliegue

Escenario 1: Aplicación IP Accediendo a la Red TDM

OmniSS7 envía consultas MAP (SRI-SM, PRN) a HLRs basados en TDM a través del SGW:

```
OmniSS7 (ASP) → SGW (SGP) → Red TDM → HLR
     M3UA          MTP2           MTP3
```

Escenario 2: OmniSS7 como STP Entre IP y TDM

OmniSS7 enruta tráfico entre elementos de red basados en IP y redes TDM:

```
SMSC IP → OmniSS7 STP → SGW → HLR TDM
               ↓
         HLR basado en IP
```

Escenario 3: SGW Doble para Redundancia

Conéctese a dos Puertas de Enlace de Señalización para alta disponibilidad:

```
peers: [
  %{
    peer_id: 1,
    name: "SGW_Primary",
    role: :client,
    remote_ip: {10, 0, 0, 100},
    remote_port: 2905,
    point_code: 100,
    # ... otra configuración
  },
  %{
    peer_id: 2,
    name: "SGW_Backup",
    role: :client,
    remote_ip: {10, 0, 0, 101},
    remote_port: 2905,
    point_code: 100,
    # ... otra configuración
  }
],

m3ua_routes: [
  # Ruta primaria a través de SGW_Primary
  %{dest_pc: 100, peer_id: 1, priority: 1, network_indicator:
:international},
  # Ruta de respaldo a través de SGW_Backup
  %{dest_pc: 100, peer_id: 2, priority: 2, network_indicator:
:international}
]
```

Habilitando el Modo M3UA STP

OmniSS7 puede operar en diferentes modos. Para usarlo como un STP, debe habilitar el modo STP en la configuración.

Cambiando al Modo STP

El `config/runtime.exs` de OmniSS7 contiene tres modos operativos preconfigurados. Para habilitar el modo STP:

1. **Abrir** `config/runtime.exs`
2. **Encontrar** las tres secciones de configuración (líneas 53-174):
   - Configuración 1: Modo STP (líneas 53-85)
   - Configuración 2: Modo HLR (líneas 87-123)
   - Configuración 3: Modo SMSc (líneas 125-174)
3. **Comentar** la configuración actualmente activa (agregar `#` a cada línea)
4. **Descomentar** la configuración STP (eliminar `#` de las líneas 53-85)
5. **Personalizar** los parámetros de configuración según sea necesario
6. **Reiniciar** la aplicación: `iex -S mix`

Configuración del Modo STP

La configuración completa del STP se ve así:

```
config :omniss7,
  # Banderas de modo - Habilitar características STP solamente
  map_client_enabled: true,
  hlr_mode_enabled: false,
  smsc_mode_enabled: false,

  # Configuración de Conexión M3UA
  # Conectar como ASP (Proceso de Servidor de Aplicación) a
STP/SGW remoto
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :stp_client_asp,
    # Punto final local (este sistema)
    local_ip: {10, 179, 4, 10},
    local_port: 2905,
    # Punto final remoto STP/SGW
    remote_ip: {10, 179, 4, 11},
    remote_port: 2905,
    routing_context: 1
  }
```

Parámetros de Configuración a Personalizar

Para una referencia completa de todos los parámetros de configuración, consulte la Referencia de Configuración.

Parámetro	Tipo	Predeterminado	Descripción	Ejer
`map_client_enabled`	Booleano	`true`	Habilitar capacidades de cliente MAP y enrutamiento	`tru`
`local_ip`	Tupla o Lista	*Requerido*	Dirección IP(s) de su sistema. Única: `{10, 0, 0, 1}` o Lista para multihoming: `[{10, 0, 0, 1}, {10, 0, 0, 2}]`	`{10` `179,` `10}`
`local_port`	Entero	`2905`	Puerto SCTP local	`290`
`remote_ip`	Tupla o Lista	*Requerido*	Dirección IP(s) remota de STP/SGW. Única o Lista para multihoming	`{10` `179,` `11}`
`remote_port`	Entero	`2905`	Puerto SCTP remoto	`290`
`routing_context`	Entero	`1`	ID de contexto de enrutamiento M3UA	`1`

Parámetro	Tipo	Predeterminado	Descripción	Ejer
`enable_gt_routing`	Booleano	`false`	Habilitar enrutamiento de Título Global (además del enrutamiento por PC)	`tru`

> **Consejo**: Utilice multihoming SCTP proporcionando una lista de direcciones IP para `local_ip` y/o `remote_ip` para habilitar la conmutación por error automática. Consulte la Guía de Multihoming SCTP.

¿Qué Sucede Cuando se Habilita el Modo STP?

Cuando `map_client_enabled: true`, la interfaz web mostrará:

- **Eventos SS7** - Registro de eventos
- **Cliente SS7** - Pruebas de operación MAP
- **M3UA** - Estado de conexión
- **Enrutamiento** - Gestión de tabla de rutas ← *específico de STP*
- **Prueba de Enrutamiento** - Pruebas de ruta ← *específico de STP*
- **Recursos** - Monitoreo del sistema
- **Configuración** - Visor de configuración

Las pestañas **Enlaces HLR** y **Enlaces SMSc** estarán ocultas.

Notas Importantes

- El protocolo SCTP (protocolo IP 132) debe ser permitido a través de los firewalls
- El puerto M3UA predeterminado es 2905 (estándar de la industria)
- Asegúrese de que haya suficientes recursos del sistema para manejar el tráfico de enrutamiento

- **Persistencia de Enrutamiento**: Todas las rutas configuradas a través de la Interfaz Web o API se almacenan en la **base de datos Mnesia** y **sobreviven a los reinicios**
- **Fusión de Configuración**: Las rutas de `runtime.exs` se cargan al inicio y se fusionan con las rutas de Mnesia
- Después de cambiar modos, debe reiniciar la aplicación para que los cambios surtan efecto
- **Interfaz Web**: Consulte la Guía de Interfaz Web para gestionar rutas a través de la interfaz web
- **Acceso API**: Consulte la Guía API para la documentación de API REST y acceso a Swagger UI

Modo STP Independiente

Además de las capacidades de enrutamiento STP disponibles cuando `map_client_enabled: true`, puede ejecutar un **servidor STP M3UA independiente** que escucha conexiones entrantes.

Habilitando STP Independiente

Agregue esta configuración a `config/runtime.exs`:

```
config :omniss7,
  sctp_handler: %{
    enabled: true,
    local_ip: {127, 0, 0, 1},    # Dirección IP para escuchar
    local_port: 2905,             # Puerto para escuchar
    point_code: 100               # Código de punto propio de este
STP
  }
```

Parámetros de Configuración STP

Parámetro	Tipo	Predeterminado	Descripción	Ejemplo
`enabled`	Booleano	`false`	Habilitar servidor STP independiente	`true`
`local_ip`	Tupla	`{127, 0, 0, 1}`	Dirección IP para escuchar conexiones	`{0, 0, 0, 0}`
`local_port`	Entero	`2905`	Puerto para escuchar	`2905`
`point_code`	Entero	*Requerido*	Código de punto SS7 propio de este STP	`100`

Cuándo Usar STP Independiente

- **Enrutamiento Puro**: Cuando solo necesita enrutamiento M3UA sin funcionalidad de cliente MAP
- **STP Central**: Para crear un enrutador de señalización central para múltiples elementos de red
- **Arquitectura Hub**: Conectar múltiples HLRs, MSCs y SMSCs a través de un STP central

Nota: Puede habilitar tanto `map_client_m3ua` como `sctp_handler` simultáneamente si necesita tanto conexiones salientes como funcionalidad STP entrante.

Persistencia de Tabla de Enrutamiento (Mnesia)

Todas las tablas de enrutamiento (pares, rutas de Código de Punto y rutas de Título Global) se almacenan en una **base de datos Mnesia** para persistencia.

Cómo Funciona el Enrutamiento

1. **Rutas de runtime.exs**: Las rutas definidas en `config/runtime.exs` bajo `peers` (o `m3ua_peers` heredados), `m3ua_routes`, y `m3ua_gt_routes` se cargan al inicio de la aplicación
2. **Rutas de Interfaz Web**: Las rutas agregadas a través de la página de Enrutamiento de la Interfaz Web se almacenan en Mnesia
3. **Fusión de Rutas**: Al reiniciar, las rutas de runtime.exs se fusionan con las rutas existentes de Mnesia (sin duplicados)
4. **Persistencia**: Todas las rutas configuradas a través de la Interfaz Web **sobreviven a los reinicios de la aplicación**

Tipo de Almacenamiento Mnesia

Controle cómo se almacenan las tablas de enrutamiento. Para más detalles sobre la configuración de la base de datos, consulte Parámetros de Base de Datos en la Referencia de Configuración.

```
config :omniss7,
  mnesia_storage_type: :disc_copies  # o :ram_copies para pruebas
```

Tipo de Almacenamiento	Descripción	Persistencia	Caso de Uso
`:disc_copies`	Almacenamiento respaldado en disco (predeterminado)	**Sobrevive a reinicios**	Entornos de producción
`:ram_copies`	Solo en memoria	Perdido en reinicio	Pruebas, desarrollo

Predeterminado: `:disc_copies`

Ubicación de la Base de Datos Mnesia

Mnesia almacena tablas de enrutamiento en el directorio Mnesia de la aplicación:

- **Ubicación**: `Mnesia.{nombre_del_nodo}/` (por ejemplo, `Mnesia.nonode@nohost/`)
- **Tablas**: `m3ua_peer`, `m3ua_route`, `m3ua_gt_route`

Gestión de Rutas

Tiene tres opciones para gestionar rutas:

1. **Runtime.exs** - Configuración estática cargada al inicio
2. **Interfaz Web** - Gestión interactiva de rutas (consulte Guía de Interfaz Web)
3. **API REST** - Gestión programática de rutas (consulte Guía API)

Mejor Práctica: Utilice `runtime.exs` para la configuración base y la Interfaz Web para cambios dinámicos de rutas durante la operación.

Configurando Pares M3UA

Los pares representan puntos finales de conexión M3UA (otros STPs, HLRs, MSCs, SMSCs). Agregue pares a `config/runtime.exs`.

Ejemplo de Configuración de Pares

> **Nota**: La clave de configuración `peers` es el estándar actual. La clave heredada `m3ua_peers` aún es compatible para mantener la compatibilidad.

```
config :omniss7,
  peers: [
    # Conexión saliente a Partner STP (rol: :client)
    %{
      peer_id: 1,                           # Identificador único
      name: "Partner_STP_West",             # Nombre descriptivo
      role: :client,                        # :client para
saliente, :server para entrante
      local_ip: {10, 0, 0, 1},             # IP local para enlazar
      local_port: 0,                        # 0 = asignación
dinámica de puerto
      remote_ip: {10, 0, 0, 10},           # IP del par remoto
      remote_port: 2905,                    # Puerto del par
remoto
      routing_context: 1,                   # Contexto de
enrutamiento M3UA
      point_code: 100,                      # Código de punto de
este par
      network_indicator: :international     # :international o
:national
    },

    # Conexión al HLR Local (rol: :client)
    %{
      peer_id: 2,
      name: "Local_HLR",
      role: :client,
      local_ip: {10, 0, 0, 1},
      local_port: 0,
      remote_ip: {10, 0, 0, 20},
      remote_port: 2905,
      routing_context: 2,
      point_code: 200,
      network_indicator: :international
    },

    # Conexión entrante desde MSC Remoto (rol: :server)
    # Para rol :server, el STP espera una conexión entrante
    %{
      peer_id: 3,
      name: "Remote_MSC",
      role: :server,                        # Aceptar conexión
entrante
```

```
      remote_ip: {10, 0, 0, 30},           # IP fuente esperada
      remote_port: 2905,                    # Puerto fuente
esperado (0 = aceptar de cualquier puerto)
      routing_context: 3,
      point_code: 300,
      network_indicator: :international
    },

    # Conexión entrante con puerto fuente dinámico (sin filtrado
de puerto)
    %{
      peer_id: 4,
      name: "Dynamic_Client",
      role: :server,
      remote_ip: {10, 0, 0, 40},           # IP fuente esperada
      remote_port: 0,                       # 0 = aceptar
conexiones de cualquier puerto fuente
      routing_context: 4,
      point_code: 400,
      network_indicator: :international
    }
  ]
```

Parámetros de Configuración de Pares

Parámetro	Tipo	Requerido	Descripción
`peer_id`	Entero	Sí	Identificador numérico único para el par
`name`	Cadena	Sí	Nombre legible por humanos para registros y monitoreo
`role`	Átomo	Sí	`:client` (saliente) o `:server` (entrante)
`local_ip`	Tupla o Lista	Sí (cliente)	Dirección(es) IP local(es) para enlazar. Única: `{10, 0, 0, 1}` o Múltiples para multihoming SCTP: `[{10, 0, 0, 1}, {10, 0, 0, 2}]`
`local_port`	Entero	Sí (cliente)	Puerto local (0 para dinámico)
`remote_ip`	Tupla o Lista	Sí	**Rol de cliente**: Tupla única o lista para par remoto multihomed. **Rol de servidor**: Solo tupla única - la IP desde la que se conecta el par remoto (ver nota a continuación)
`remote_port`	Entero	Sí	Puerto del par remoto (0 para entrante = aceptar de cualquier puerto fuente)
`routing_context`	Entero	Sí	Identificador de contexto de enrutamiento M3UA

Parámetro	Tipo	Requerido	Descripción
`point_code`	Entero	Sí	Código de punto SS7 de este par
`network_indicator`	Átomo	No	`:international` o `:national`

> **Multihoming SCTP para Rol de Servidor (Conexiones Entrantes)**: Al aceptar conexiones entrantes de un par remoto multihomed, solo necesita especificar la **única dirección IP** que el par remoto utiliza para iniciar la conexión SCTP (SCTP INIT). Esto debe ser una **tupla única**, no una lista. SCTP descubrirá automáticamente las otras direcciones IP multihomed del par durante el apretón de manos de asociación. El formato de lista para `remote_ip` solo se utiliza para **rol de cliente** al conectarse a un par remoto multihomed.
>
> **Multihoming SCTP para Rol de Cliente (Conexiones Salientes)**: Para redundancia de red al conectarse a pares remotos, puede configurar múltiples direcciones IP tanto para `local_ip` como para `remote_ip` utilizando listas. Esto permite la conmutación por error automática si una ruta de red falla. Consulte la Guía de Multihoming SCTP para ejemplos de configuración detallados y mejores prácticas.

Filtrado de Puerto Fuente para Conexiones Entrantes

Para **conexiones entrantes** (rol: `:server`), el parámetro `remote_port` controla el filtrado de puertos fuente:

- **Puerto Específico** (por ejemplo, `remote_port: 2905`): Solo aceptar conexiones de ese puerto fuente exacto
  - Proporciona seguridad adicional al validar el puerto fuente
  - Usar cuando el par remoto utiliza un puerto fuente fijo
- **Cualquier Puerto** (`remote_port: 0`): Aceptar conexiones de cualquier puerto fuente

- Útil cuando el par remoto utiliza puertos fuente dinámicos/efímeros
- Solo valida la dirección IP fuente
- Más flexible pero ligeramente menos seguro

Ejemplo:

```
# Aceptar solo de 10.5.198.200:2905 (puerto específico)
%{
  peer_id: 1,
  name: "Strict_Peer",
  role: :server,
  remote_ip: {10, 5, 198, 200},
  remote_port: 2905,
  # ... otra configuración
}

# Aceptar de 10.5.198.200 con cualquier puerto fuente
%{
  peer_id: 2,
  name: "Flexible_Peer",
  role: :server,
  remote_ip: {10, 5, 198, 200},
  remote_port: 0,  # Aceptar de cualquier puerto fuente
  # ... otra configuración
}
```

Soporte del Protocolo M2PA

OmniSS7 soporta tanto **M3UA** como **M2PA** para el transporte de señalización SS7.

¿Qué es M2PA?

M2PA (Capa de Adaptación de Usuario MTP2 a Par) es un protocolo estandarizado por IETF (RFC 4165) para transportar mensajes MTP3 de SS7 sobre redes IP utilizando SCTP.

M3UA vs M2PA: Diferencias Clave

Característica	M3UA	M2PA
Arquitectura	Cliente/Servidor (ASP/SGW)	Par a Par
Caso de Uso	Puente entre SS7 e IP	Enlaces directos punto a punto
Gestión del Estado del Enlace	Nivel de aplicación (ASPUP/ASPAC)	Estilo MTP2 (Alineación, Prueba, Listo)
Números de Secuencia	Sin secuenciación inherente	BSN/FSN de 24 bits para entrega ordenada
Despliegue Típico	Puerta de enlace SS7 a IP, STP	Enlaces de señalización directos entre nodos
RFC	RFC 4666	RFC 4165

Orientación sobre la Selección de Protocolo

Recomendación: Usar M3UA por defecto. Solo usar M2PA cuando sea específicamente requerido.

Cuándo Usar M3UA (Recomendado)

M3UA es el protocolo recomendado para la mayoría de los despliegues:

- **Despliegues STP**: Implementaciones estándar de punto de transferencia de señalización
- **Funciones de Puerta de Enlace**: Conexión de redes SS7 con señalización basada en IP
- **Conexiones de Elementos de Red**: Conectar HLRs, MSCs, SMSCs y otros elementos de red a su STP

- **Puerta de Enlace de Señalización (SGW)**: Puerta de enlace central que acepta conexiones de múltiples Servidores de Aplicación
- **Topologías Flexibles**: Arquitecturas cliente/servidor con control centralizado
- **Redes de Múltiples Proveedores**: Estándar de la industria ampliamente soportado (RFC 4666)

Utilice M3UA para conectar elementos de red (HLR, MSC, SMSC, VLR, etc.) a su STP.

Cuándo Usar M2PA (Solo Casos Especiales)

M2PA solo debe usarse en escenarios específicos:

- **Enlaces STP a STP**: Conexiones directas punto a punto entre Puntos de Transferencia de Señalización en una red multi-STP
- **Reemplazo de TDM Legado**: Reemplazo de enlaces TDM SS7 tradicionales cuando el sistema remoto requiere específicamente M2PA
- **Compatibilidad MTP2 Requerida**: Al conectarse a sistemas heredados que exigen gestión del estado del enlace estilo MTP2
- **Requisito del Socio**: Cuando un socio o interconexión requiere específicamente el protocolo M2PA

Importante: No use M2PA para conectar elementos de red (HLR, MSC, SMSC) a su STP - use M3UA en su lugar. M2PA está diseñado para interconexiones STP a STP donde ambos lados operan como nodos de enrutamiento.

Configurando Pares M2PA

Los pares M2PA se configuran de la misma manera que los pares M3UA, con un parámetro adicional `protocol`.

Configuración de Par M2PA

Agregue pares M2PA a su configuración de `peers` en `config/runtime.exs` (tanto los pares M3UA como M2PA comparten la misma sección de configuración, diferenciados por el parámetro `protocol`):

Parámetros Clave para M2PA:

Parámetro	Valor	Descripción
`protocol`	`:m2pa`	Especifica el protocolo M2PA (predeterminado a `:m3ua` si se omite)
`role`	`:client` o `:server`	Dirección de conexión
`local_port`	Entero	Puerto SCTP local (el puerto estándar de M2PA es **3565**)
`remote_port`	Entero	Puerto SCTP remoto (el puerto estándar de M2PA es **3565**)
`point_code`	Entero	Su código de punto
`adjacent_point_code`	Entero	Código de punto del par remoto (específico de M2PA)
`send_sltm`	Booleano	Controla el comportamiento SLTM (ver Configuración SLTM a continuación)
`network_indicator`	Átomo	`:international` o `:national` - debe coincidir con el par remoto

Nota: M2PA utiliza **puerto 3565** como el estándar de la industria (diferente del puerto 2905 de M3UA).

Configuración SLTM

SLTM (Mensaje de Prueba de Enlace de Señalización) y **SLTA (Reconocimiento de Prueba de Enlace de Señalización)** son mensajes de mantenimiento MTP3 utilizados para verificar la conectividad de extremo a

extremo después de que un enlace M2PA alcanza el estado LISTO. Según ITU-T Q.707, un lado envía SLTM y el otro responde con SLTA para confirmar que el enlace está operativo.

Comportamiento SLTM

El parámetro `send_sltm` controla qué lado inicia la prueba SLTM:

Valor de `send_sltm`	Comportamiento
`true`	Enviamos SLTM cuando el enlace se vuelve LISTO, esperamos SLTA
`false`	Esperamos que el par envíe SLTM, respondemos con SLTA
No establecido (predeterminado)	Sigue Q.707: el respondedor SCTP envía SLTM, el iniciador SCTP espera

Comportamiento Predeterminado (Cumplimiento de Q.707):

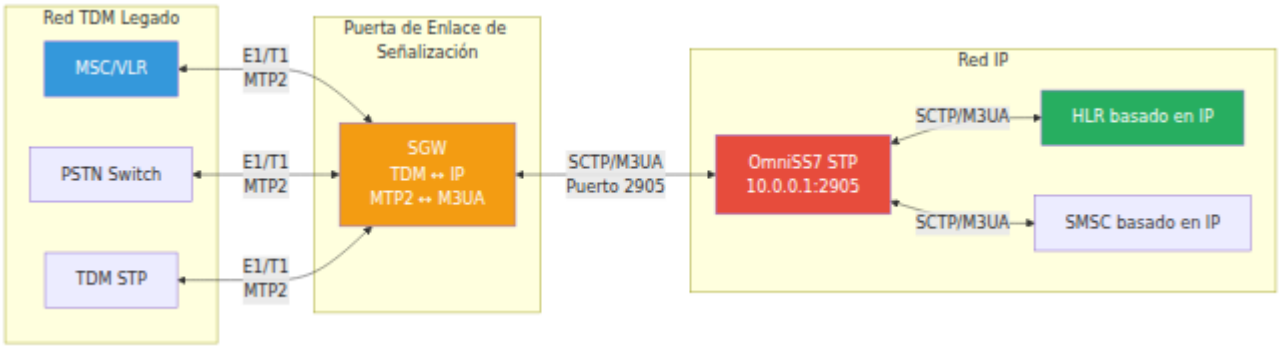
- Si `initiate_connection: true` (iniciador/cliente SCTP) → Esperamos que el par envíe SLTM
- Si `initiate_connection: false` (respondedor/servidor SCTP) → Enviamos SLTM

Cuándo Sobrescribir los Valores Predeterminados de SLTM

Sobrescriba el comportamiento predeterminado al conectarse a equipos que no siguen las convenciones Q.707:

```
# Ejemplo: Forzar a nuestro lado a enviar SLTM independientemente
del rol SCTP
%{
  peer_id: 100,
  name: "Partner_STP",
  protocol: :m2pa,
  role: :client,
  local_ip: {10, 0, 0, 1},
  local_port: 3565,
  remote_ip: {10, 0, 0, 2},
  remote_port: 3565,
  point_code: 7415,
  adjacent_point_code: 15528,
  network_indicator: :international,
  send_sltm: true  # Sobrescribir: Enviamos SLTM incluso si somos
el iniciador SCTP
}
```

Flujo de Mensaje SLTM



Solución de Problemas de Problemas SLTM

Síntoma: El enlace alcanza LISTO pero no fluye tráfico

Ambos lados pueden estar esperando que el otro envíe SLTM. Verifique los registros para:

```
"esperando que el par envíe SLTM"
```

Resolución: Configure `send_sltm: true` en un lado para romper el estancamiento.

Síntoma: SLTM enviado pero no se recibe SLTA

Causas posibles:

1. `adjacent_point_code` es incorrecto (SLTM enviado con DPC incorrecto)
2. Desajuste de indicador de red (`:international` vs `:national`)
3. El par remoto no está configurado para responder a nuestro código de punto

Resolución: Verifique que `adjacent_point_code` coincida con el código de punto propio del STP (configurado en `sctp_handler.point_code`).

Estados de Enlace M2PA

Los enlaces M2PA progresan a través de varios estados durante la inicialización:

1. **Abajo** - No se establece conexión
2. **Alineación** - Fase de sincronización inicial (~1 segundo)
3. **Prueba** - Verificación de calidad del enlace (~2 segundos)
4. **Listo** - Enlace activo y listo para tráfico

La progresión del estado del enlace asegura una señalización confiable antes de que se intercambie tráfico.

Gestión de Pares M2PA a través de la Interfaz Web

La página de **Enrutamiento** en la Interfaz Web proporciona soporte completo para gestionar pares M2PA:

1. **Navegar** a la página de Enrutamiento
2. **Seleccionar** la pestaña "Pares"
3. **Hacer clic** en "Agregar Nuevo Par"
4. **Elegir** "M2PA (RFC 4165)" del menú desplegable de Protocolo
5. **Completar** la configuración del par:
   - Nombre del Par (identificador descriptivo)
   - Protocolo: M2PA

    - Rol: cliente o servidor
    - Código de Punto (su PC)
    - Direcciones IP Locales/Remotas
    - Puertos Locales/Remotos (típicamente 3565 para M2PA)
    - Indicador de Red (internacional o nacional)
6. **Hacer clic** en "Guardar Par"

La tabla de pares muestra el tipo de protocolo con codificación de color:

- **Azul** - pares M3UA
- **Verde** - pares M2PA

Comportamiento de Enrutamiento M2PA

Los pares M2PA se integran sin problemas con el sistema de enrutamiento de OmniSS7:

- **Rutas de Código de Punto**: Funcionan idénticamente para M2PA y M3UA
- **Rutas de Título Global**: Totalmente soportadas en enlaces M2PA
- **Prioridad de Ruta**: Los pares M2PA y M3UA pueden mezclarse en las mismas tablas de enrutamiento
- **Reenvío de Mensajes**: Los mensajes pueden llegar a M2PA y ser enrutados a M3UA, y viceversa

Métricas M2PA

M2PA proporciona métricas completas de Prometheus para monitorear la salud del enlace y el tráfico:

Métricas de Tráfico:

- `m2pa_messages_sent_total` - Total de mensajes MTP3 enviados por enlace
- `m2pa_messages_received_total` - Total de mensajes MTP3 recibidos por enlace
- `m2pa_bytes_sent_total` - Total de bytes enviados a través de M2PA
- `m2pa_bytes_received_total` - Total de bytes recibidos a través de M2PA

Todas las métricas de tráfico están etiquetadas por: `link_name`, `point_code`, `adjacent_pc`

Métricas de Estado del Enlace:

- `m2pa_link_state_changes_total` - Transiciones de estado del enlace (DOWN → ALIGNMENT → PROVING → READY)
  - Etiquetas: `link_name`, `from_state`, `to_state`

Métricas de Error:

- `m2pa_errors_total` - Total de errores por tipo
  - `decode_error` - Fallos de decodificación de mensajes M2PA
  - `encode_error` - Fallos de codificación de mensajes M2PA
  - `sctp_send_error` - Fallos de transmisión SCTP
  - Etiquetas: `link_name`, `error_type`

Acceso a Métricas:

- Punto final de Prometheus: `http://su-servidor:8080/metrics`
- Las métricas se registran automáticamente al inicio de la aplicación

Mejores Prácticas M2PA

1. **Selección de Puerto**: Use el puerto 3565 para M2PA (estándar de la industria)
2. **Monitoreo de Enlace**: Monitoree los cambios de estado del enlace a través de métricas
3. **Reglas de Firewall**: Asegúrese de que SCTP (protocolo IP 132) esté permitido
4. **Códigos de Punto**: Asegúrese de que los códigos de punto adyacentes estén configurados correctamente en ambos lados
5. **Indicador de Red**: Debe coincidir entre pares (internacional o nacional)
6. **Pruebas**: Utilice la página de Prueba de Enrutamiento para verificar la conectividad después de la configuración

Requisitos de Socket M2PA

M2PA utiliza sockets compartidos de SCTP.SocketHandler. Todos los pares M2PA utilizan automáticamente el SCTP.SocketHandler para la gestión de sockets, lo que permite que múltiples pares compartan eficientemente el mismo puerto SCTP.

Requisitos:

- SCTP.SocketHandler debe estar habilitado y en funcionamiento
- El socket compartido debe configurarse antes de que los pares M2PA comiencen

Ejemplo:

```
# Habilitar SCTP.SocketHandler
sctp_handler: %{
  enabled: true,
  local_ip: {10, 179, 4, 10},
  local_port: 3565,
  point_code: 100
}

# Los pares M2PA utilizan automáticamente el socket compartido
peers: [
  %{
    peer_id: 1,
    name: "M2PA_Link_1",
    protocol: :m2pa,
    role: :client,
    local_ip: {10, 179, 4, 10},
    local_port: 3565,
    remote_ip: {10, 179, 4, 20},
    remote_port: 3565,
    point_code: 100,
    adjacent_point_code: 200
  },
  %{
    peer_id: 2,
    name: "M2PA_Link_2",
    protocol: :m2pa,
    role: :client,
    local_ip: {10, 179, 4, 10},
    local_port: 3565,           # Mismo puerto compartido entre
pares
    remote_ip: {10, 179, 4, 30},
    remote_port: 3565,
    point_code: 100,
    adjacent_point_code: 300
  }
]
```

Requisitos SCTP:

- Entrega ordenada (sin bandera `unordered`)
- PPID 5 (identificador M2PA según RFC 4165)

Configurando Enrutamiento por Código de Punto

El enrutamiento por Código de Punto dirige mensajes basados en el **Código de Punto de Destino (DPC)** en el encabezado MTP3.

Entendiendo los Códigos de Punto en la Pila de Protocolo SS7

Los códigos de punto existen en diferentes capas de la pila de protocolo SS7. Entender esta distinción es importante:

Capas de la Pila de Protocolo:

```
┌─────────────────────────────────────────────┐
│  Capa de Aplicación (SCCP/TCAP/MAP)         │
├─────────────────────────────────────────────┤
│  Capa MTP3                                  │ ← Enrutamiento de Datos
de Usuario
│  - Etiqueta de Enrutamiento: DPC, OPC, SLS  │ ← Usado
para enrutamiento STP
│  - Octeto de Información de Servicio (SIO)  │
├─────────────────────────────────────────────┤
│  M3UA o M2PA (Capa de Adaptación)           │ ← Protocolo de
Transporte
│  - Datos de Protocolo (contiene MTP3)       │
│  - Gestión de Red (DUNA/DAVA)               │ ← Estado de la Red
├─────────────────────────────────────────────┤
│  SCTP (Transporte)                          │
└─────────────────────────────────────────────┘
```

Dos Tipos de Códigos de Punto:

1. **Códigos de Punto de Capa MTP3** (Usados para Enrutamiento):
    - Ubicados en la etiqueta de enrutamiento MTP3 (DPC, OPC)
    - Presentes en el parámetro de Datos de Protocolo M3UA (etiqueta 528)
    - Presentes en mensajes de Datos de Usuario M2PA

    - **El STP utiliza estos valores DPC para decisiones de enrutamiento**
    - Estos determinan dónde se entrega finalmente el mensaje

2. **Códigos de Punto de Capa M3UA** (Usados para Gestión de Red):
    - Presentes en mensajes de gestión M3UA (DUNA, DAVA, SCON, DUPU)
    - Indican códigos de punto afectados para el estado de la red
    - Informan a los pares qué destinos están disponibles/no disponibles
    - No se utilizan para enrutamiento de datos de usuario

Cómo Funciona el Enrutamiento STP:

- **Para mensajes M3UA DATA**: El STP extrae el mensaje MTP3 del parámetro de Datos de Protocolo (etiqueta 528), que contiene la etiqueta de enrutamiento MTP3 (DPC, OPC, SLS). El DPC de la capa MTP3 se utiliza para buscar rutas.
- **Para mensajes de Datos de Usuario M2PA**: El STP extrae el mensaje MTP3 del campo de datos de usuario M2PA, luego lee el DPC de la etiqueta de enrutamiento MTP3.
- **Mensajes de gestión M3UA**: Los mensajes de gestión de red (DUNA, DAVA, SCON) contienen códigos de punto afectados en la capa M3UA para el estado de señalización entre pares.

Rutas Básicas de Código de Punto

Agregue rutas a `config/runtime.exs`:

```
config :omniss7,
  m3ua_routes: [
    # Rutar todo el tráfico para PC 100 al par 1 (Partner STP)
    %{
      dest_pc: 100,                    # Código de punto de
destino
      peer_id: 1,                      # Par a través del cual
enrutar
      priority: 1,                     # Prioridad (menor = mayor
prioridad)
      network_indicator: :international
      # mask: 14                       # Opcional: predeterminado
a 14 (coincidencia exacta)
    },

    # Rutar todo el tráfico para PC 200 al par 2 (HLR Local)
    %{
      dest_pc: 200,
      peer_id: 2,
      priority: 1,
      network_indicator: :international
    },

    # Ejemplo de balanceo de carga: PC 300 con rutas primaria y de
respaldo
    %{
      dest_pc: 300,
      peer_id: 3,                      # Ruta primaria
      priority: 1,
      network_indicator: :international
    },
    %{
      dest_pc: 300,
      peer_id: 4,                      # Ruta de respaldo (número
de prioridad más alto)
      priority: 2,
      network_indicator: :international
    }
  ]
```

Nota: El campo `mask` es opcional y predetermina a `14` (coincidencia exacta). Solo especifique `mask` cuando necesite enrutamiento basado en rangos

(consulte la sección de Máscaras de Código de Punto a continuación).

Lógica de Enrutamiento

1. El STP recibe un mensaje M3UA DATA o un mensaje de Datos de Usuario M2PA
2. El STP extrae el **mensaje MTP3** del campo de Datos de Protocolo (M3UA) o del campo de Datos de Usuario (M2PA)
3. El STP lee el **Código de Punto de Destino (DPC)** de la etiqueta de enrutamiento MTP3
4. Busca en la tabla de enrutamiento el DPC coincidente (considerando máscaras)
5. Si existen múltiples rutas, selecciona la ruta con **máscara más específica** (valor de máscara más alto), luego **número de prioridad más bajo**
6. Envuelve el mensaje MTP3 en M3UA DATA o Datos de Usuario M2PA para el par de destino
7. Rutea el mensaje al par correspondiente
8. Si el par seleccionado está inactivo, intenta la siguiente ruta de mayor prioridad

Máscaras de Código de Punto

Los códigos de punto son valores de 14 bits (rango 0-16383). Por defecto, las rutas coinciden exactamente con un solo código de punto (máscara `/14`). Sin

embargo, puede usar **máscaras de código de punto** para crear rutas que coincidan con **rangos** de códigos de punto.

Entendiendo Máscaras

La máscara especifica cuántos **bits más significativos** deben coincidir entre el PC de destino de la ruta y el DPC del mensaje entrante. Los bits restantes pueden ser cualquier valor, creando un rango de códigos de punto coincidentes.

Tabla de Referencia de Máscaras:

Máscara	Códigos de Punto Coincidentes	Caso de Uso
`/14`	1 PC (coincidencia exacta)	Destino único (predeterminado)
`/13`	2 PCs	Pequeño rango
`/12`	4 PCs	Pequeño rango
`/11`	8 PCs	Pequeño rango
`/10`	16 PCs	Rango medio
`/9`	32 PCs	Rango medio
`/8`	64 PCs	Rango medio
`/7`	128 PCs	Rango medio-grande
`/6`	256 PCs	Rango grande
`/5`	512 PCs	Rango grande
`/4`	1,024 PCs	Rango muy grande
`/3`	2,048 PCs	Rango muy grande
`/2`	4,096 PCs	Rango extremadamente grande
`/1`	8,192 PCs	La mitad de todos los PCs
`/0`	16,384 PCs	Todos los PCs (ruta predeterminada/de respaldo)

Ejemplos de Máscaras de Código de Punto

Nota: El campo `mask` es **opcional** en todos los ejemplos. Si se omite, predetermina a `14` (coincidencia exacta).

Ejemplo 1: Código de Punto Único (Comportamiento Predeterminado)

```
# Sin campo de máscara (recomendado para un solo PC)
%{
  dest_pc: 1000,
  peer_id: 1,
  priority: 1,
  network_indicator: :international
}
# La máscara predetermina a 14 - Coincide: Solo PC 1000

# Máscara explícita (mismo resultado)
%{
  dest_pc: 1000,
  peer_id: 1,
  priority: 1,
  mask: 14,                              # Coincidencia exacta
explícita
  network_indicator: :international
}
# Coincide: Solo PC 1000
```

Ejemplo 2: Pequeño Rango

```
%{
  dest_pc: 1000,
  peer_id: 2,
  priority: 1,
  mask: 12,                              # Coincide con 4 PCs
  network_indicator: :international
}
# Coincide: PC 1000, 1001, 1002, 1003
```

Ejemplo 3: Rango Medio

```
%{
  dest_pc: 1000,
  peer_id: 3,
  priority: 1,
  mask: 8,                                # Coincide con 64 PCs
  network_indicator: :international
}
# Coincide: PC 1000-1063 (64 códigos de punto consecutivos)
```

Ejemplo 4: Ruta Predeterminada/de Respaldo

```
%{
  dest_pc: 0,
  peer_id: 4,
  priority: 10,                           # Baja prioridad (número
alto)
  mask: 0,                                # Coincide con todos los
PCs
  network_indicator: :international
}
# Coincide: Todos los códigos de punto (0-16383)
# Usar como ruta de captura predeterminada con baja prioridad
```

Combinando Rutas Específicas y Rutas enmascaradas

Puede combinar rutas específicas con rutas enmascaradas para un enrutamiento flexible:

```
config :omniss7,
  m3ua_routes: [
    # Ruta específica para PC 100 (tiene prioridad)
    %{
      dest_pc: 100,
      peer_id: 1,
      priority: 1,
      network_indicator: :international
      # mask predetermina a 14 (coincidencia exacta)
    },

    # Ruta de rango para PCs 1000-1063
    %{
      dest_pc: 1000,
      peer_id: 2,
      priority: 1,
      mask: 8,                          # Coincide con 64 PCs
      network_indicator: :international
    },

    # Ruta predeterminada/de respaldo para todos los demás PCs
    %{
      dest_pc: 0,
      peer_id: 3,
      priority: 10,                     # Baja prioridad
      mask: 0,                          # Coincide con todos los
PCs
      network_indicator: :international
    }
  ]
```

Decisión de Enrutamiento para DPC 1000:

1. Coincide con la ruta de máscara `/14` (PC 1000 exactamente) - **Seleccionada** (más específica)
2. También coincide con la ruta de máscara `/8` (rango de PC 1000-1063) - Ignorada (menos específica)
3. También coincide con la ruta de máscara `/0` (todos los PCs) - Ignorada (menos específica)

Decisión de Enrutamiento para DPC 1015:

1. No coincide con la ruta de máscara `/14` (PC 1000 solamente)
2. Coincide con la ruta de máscara `/8` (rango de PC 1000-1063) - **Seleccionada** (coincidencia más específica)
3. También coincide con la ruta de máscara `/0` (todos los PCs) - Ignorada (menos específica)

Decisión de Enrutamiento para DPC 5000:

1. No coincide con la ruta de máscara `/14`
2. No coincide con la ruta de máscara `/8`
3. Coincide con la ruta de máscara `/0` (todos los PCs) - **Seleccionada** (única coincidencia, ruta de respaldo)

Mejores Prácticas

1. **Omitir `mask` para Destinaciones Únicas**: Para coincidencias exactas de códigos de punto, omita el campo de máscara por completo (predetermina a `/14`)
2. **Usar `/14` Explícitamente Solo Cuando Sea Necesario**: Solo especifique `mask: 14` cuando necesite dejarlo claro en la documentación o al mezclar con rutas de rango
3. **Usar Máscaras de Rango para Bloques de Red**: Rutee segmentos completos de red a pares específicos con máscaras `/0` a `/13`
4. **Usar `/0` como Respaldo**: Cree una ruta predeterminada con baja prioridad para capturar tráfico no coincidente
5. **El Más Específico Gana**: El motor de enrutamiento siempre selecciona la ruta coincidente más específica (valor de máscara más alto) primero
6. **Prioridad como Desempate**: Si múltiples rutas tienen la misma máscara, el número de prioridad más bajo gana

---

Configurando Enrutamiento por Título Global (GT)

El enrutamiento por Título Global permite el **enrutamiento basado en contenido** utilizando números de teléfono o valores IMSI en lugar de códigos

de punto. Para la traducción avanzada de direcciones de Título Global basada en la parte que llama/recibe, consulte la Guía de NAT de Título Global.

Requisitos Previos

- Habilitar el enrutamiento GT: `enable_gt_routing: true` en `config/runtime.exs`

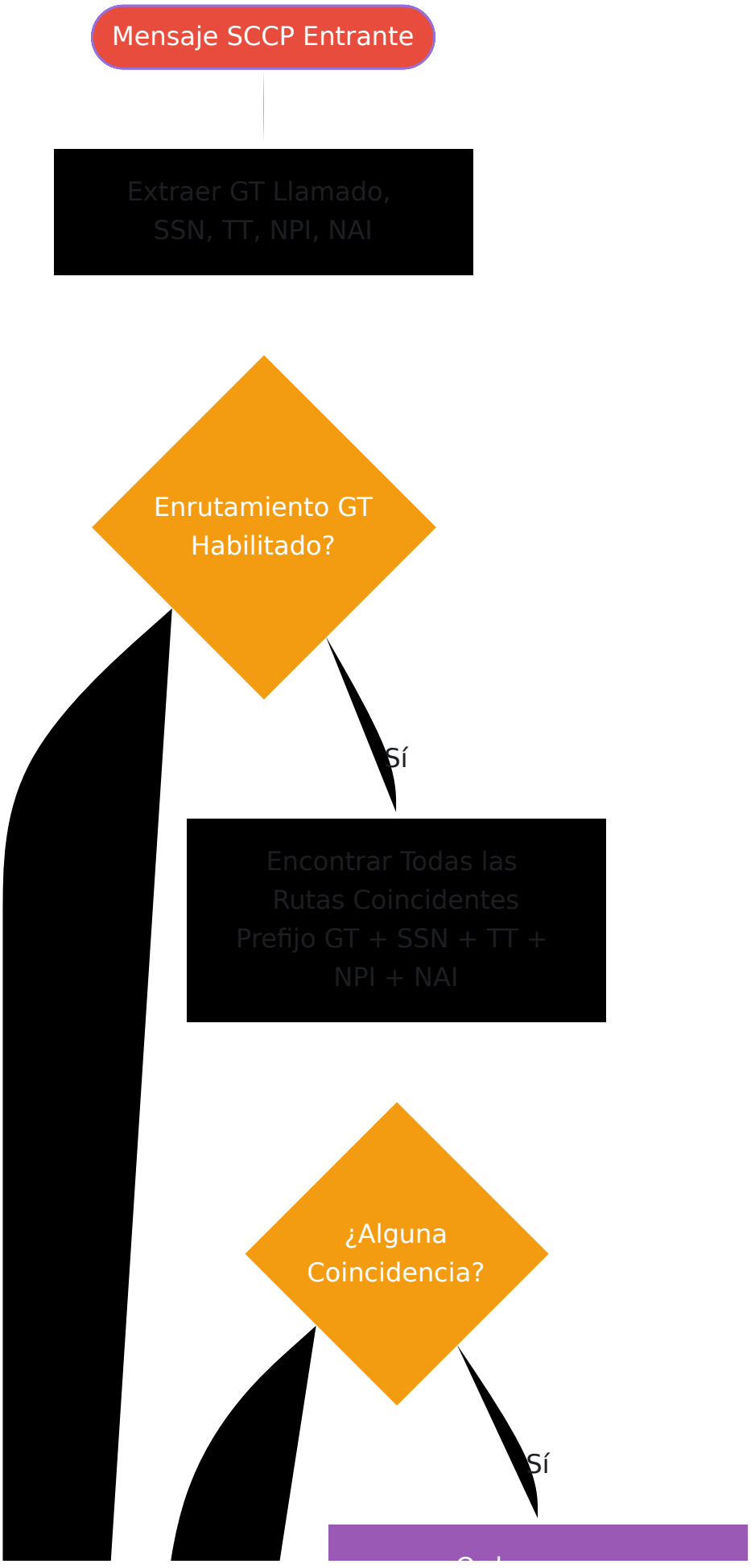
Configuración de Rutas GT

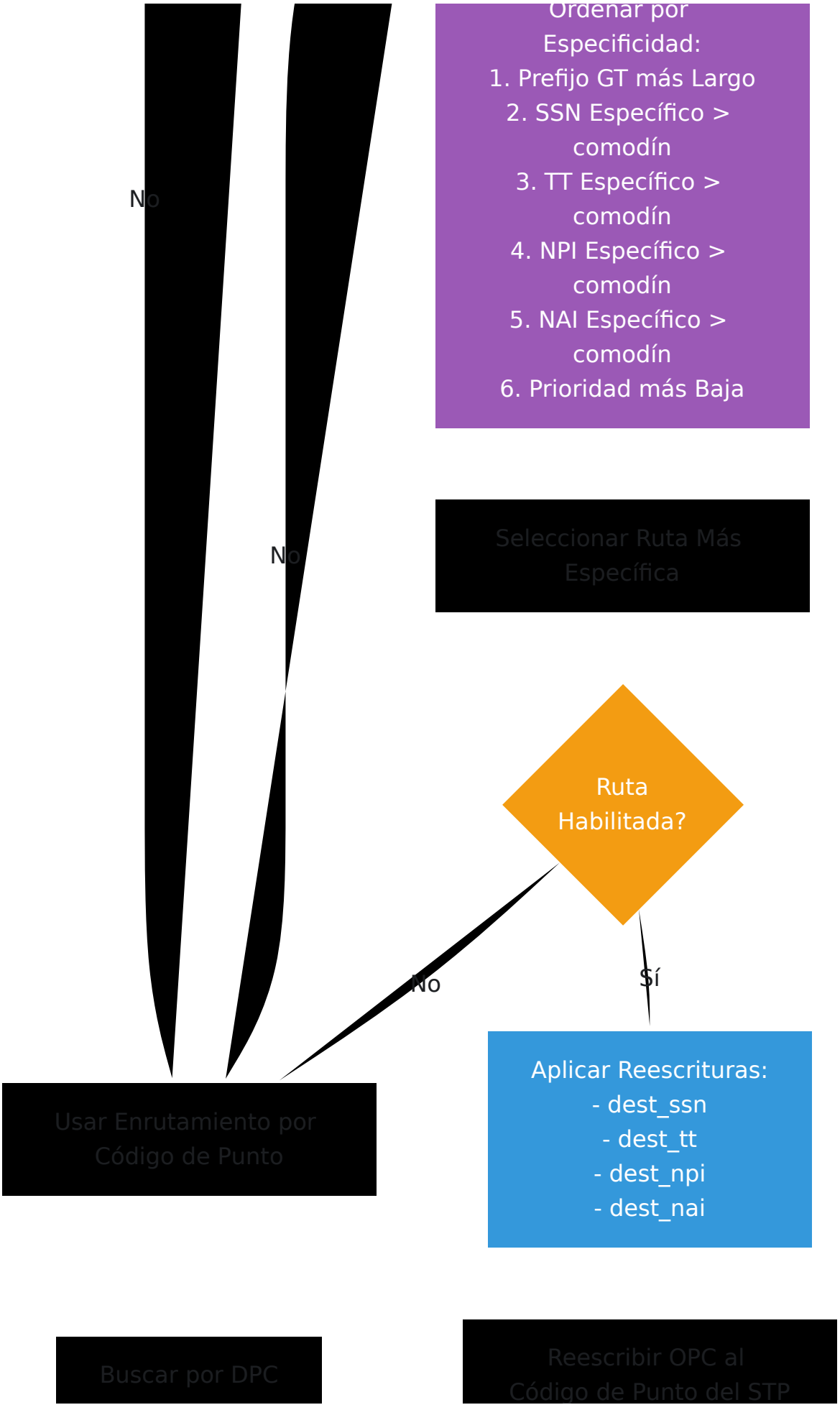
```
config :omniss7,
  # Habilitar enrutamiento GT
  enable_gt_routing: true,

  m3ua_gt_routes: [
    # Rutar todos los números del Reino Unido (prefijo 44) al par
1
    %{
      gt_prefix: "44",                # Prefijo de Título Global
a coincidir
      peer_id: 1,                     # Par de destino
      priority: 1,                    # Prioridad (menor = mayor)
      description: "Números del Reino Unido"        # Descripción
para registro
    },

    # Rutar números de EE. UU. (prefijo 1) al par 2
    %{
      gt_prefix: "1",
      peer_id: 2,
      priority: 1,
      description: "Números de EE. UU."
    },

    # Ruta más específica: números móviles del Reino Unido que
comienzan con 447
    %{
      gt_prefix: "447",               # La coincidencia más larga
gana
      peer_id: 3,
      priority: 1,
      description: "Números móviles del Reino Unido"
    },

    # Enrutamiento específico de SSN (opcional)
    %{
      gt_prefix: "555",
      source_ssn: 8,                  # Solo coincidir si SSN
fuente = 8 (SMSC)
      peer_id: 4,
      dest_ssn: 6,                    # Reescribir SSN de destino
```

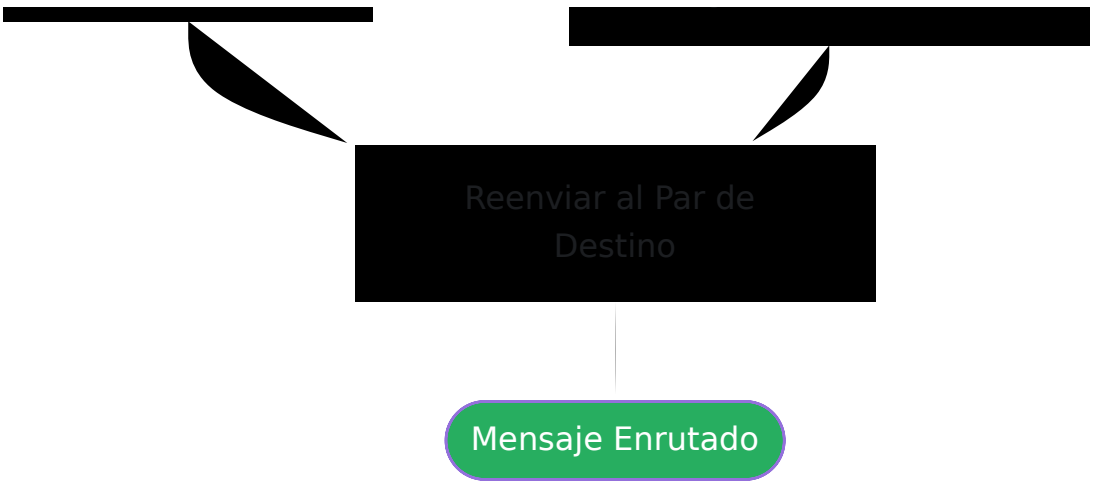
```
a 6 (HLR)
      priority: 1,
      description: "Tráfico SMS para prefijo 61"
    }
  ]
```

Lógica de Enrutamiento GT

El algoritmo de enrutamiento GT sigue este proceso de decisión:


Mensaje SCCP Entrante
Extraer GT Llamado, SSN, TT, NPI, NAI
Enrutamiento GT Habilitado?
Sí
Encontrar Todas las Rutas Coincidentes Prefijo GT + SSN + TT + NPI + NAI
¿Alguna Coincidencia?
Sí


Ordenar por Especificidad:
1. Prefijo GT más Largo
2. SSN Específico > comodín
3. TT Específico > comodín
4. NPI Específico > comodín
5. NAI Específico > comodín
6. Prioridad más Baja
No
No
Seleccionar Ruta Más Específica
Ruta Habilitada?
No
Sí
Aplicar Reescrituras:
- dest_ssn
- dest_tt
- dest_npi
- dest_nai
Usar Enrutamiento por Código de Punto
Buscar por DPC
Reescribir OPC al Código de Punto del STP



Pasos de Enrutamiento:

1. **Coincidencia de Prefijo más Largo**: El STP encuentra todas las rutas GT donde el prefijo coincide con el comienzo del Título Global
    - Ejemplo: GT "447712345678" coincide con "44" y "447", pero "447" gana (coincidencia más larga)
2. **Coincidencia de SSN** (Opcional):
    - Si se especifica `source_ssn`, la ruta solo coincide cuando el SSN de la Parte Llamada de SCCP es igual a ese valor
    - Si `source_ssn` es `nil`, la ruta coincide con cualquier SSN (comodín)
3. **Coincidencia de TT/NPI/NAI** (Opcional):
    - Si se especifican `source_tt`, `source_npi` o `source_nai`, las rutas deben coincidir con esos indicadores
    - Los valores `nil` actúan como comodines (coinciden con cualquier valor)
4. **Selección Basada en Especificidad**:
    - Las rutas con criterios de coincidencia más específicos ganan sobre los comodines
    - Orden de prioridad: Longitud del Prefijo GT → SSN → TT → NPI → NAI → Número de Prioridad
5. **Reescritura de Indicadores** (Opcional):

- Si se especifican `dest_ssn`, `dest_tt`, `dest_npi` o `dest_nai`, el STP reescribe esos indicadores
- Útil para normalización de protocolo e interconexión de red

6. **Recaída al Código de Punto**:
   - Si no coincide ninguna ruta GT, el STP recae al enrutamiento por Código de Punto usando el DPC

Ejemplos de Configuración de GT

```
config :omniss7,
  enable_gt_routing: true,

  m3ua_gt_routes: [
    # Ejemplo 1: Coincidir y transformar el Tipo de Traducción
    %{
      gt_prefix: "44",
      peer_id: 1,
      source_tt: 0,      # Coincidir TT=0 (Desconocido)
      dest_tt: 3,        # Transformar a TT=3 (Nacional)
      priority: 1,
      description: "Números del Reino Unido: transformación TT
0→3"
    },

    # Ejemplo 2: Coincidir NPI específico y transformar NAI
    %{
      gt_prefix: "1",
      peer_id: 2,
      source_npi: 1,     # Coincidir NPI=1 (ISDN/Teléfono)
      source_nai: 4,     # Coincidir NAI=4 (Internacional)
      dest_nai: 3,       # Transformar a NAI=3 (Nacional)
      priority: 1,
      description: "Números de EE. UU.: NAI
Internacional→Nacional"
    },

    # Ejemplo 3: Enrutamiento combinado de SSN e indicadores
    %{
      gt_prefix: "33",
      source_ssn: 8,     # Coincidir tráfico de SMS
      source_tt: 0,      # Coincidir TT=0
      dest_ssn: 6,       # Reescribir SSN a HLR
      dest_tt: 2,        # Transformar a TT=2
      dest_npi: 1,       # Establecer NPI=1 (ISDN)
      dest_nai: 4,       # Establecer NAI=4 (Internacional)
      peer_id: 3,
      priority: 1,
      description: "SMS francés: normalización completa"
    },
```

```
    # Ejemplo 4: TT comodín, NPI específico
    %{
      gt_prefix: "49",
      source_tt: nil,    # Coincidir cualquier TT (comodín)
      source_npi: 6,     # Coincidir NPI=6 (Datos)
      dest_npi: 1,       # Transformar a NPI=1 (ISDN)
      peer_id: 4,
      priority: 1,
      description: "Normalización de red de datos alemana"
    }
  ]
```

Valores Comunes de TT/NPI/NAI

Tipo de Traducción (TT):

- `0` = Desconocido
- `1` = Internacional
- `2` = Nacional
- `3` = Específico de la Red

Indicador de Plan de Numeración (NPI):

- `0` = Desconocido
- `1` = ISDN/Teléfono (E.164)
- `3` = Datos (X.121)
- `4` = Telex (F.69)
- `6` = Móvil Terrestre (E.212)

Indicador de Naturaleza de Dirección (NAI):

- `0` = Desconocido
- `1` = Número de Suscriptor
- `2` = Reservado para Uso Nacional
- `3` = Número Significativo Nacional
- `4` = Número Internacional

Ejemplo de Decisión de Enrutamiento para TT/NPI/NAI

Para un mensaje entrante con:

- GT: "447712345678"
- SSN: 8
- TT: 0
- NPI: 1
- NAI: 4

Con estas rutas configuradas:

```
# Ruta A: TT comodín
%{gt_prefix: "447", peer_id: 1, priority: 1}

# Ruta B: TT específico
%{gt_prefix: "447", source_tt: 0, peer_id: 2, priority: 1}

# Ruta C: TT específico + NPI
%{gt_prefix: "447", source_tt: 0, source_npi: 1, peer_id: 3,
priority: 1}
```

Resultado: La Ruta C es seleccionada (más específica: coincide GT + TT + NPI)

El mensaje es reenviado con indicadores transformados según los valores de `dest_tt`, `dest_npi`, `dest_nai` de la Ruta C.

Ejemplos de Enrutamiento GT

GT Llamado	SSN Fuente	TT	NPI	NAI	Ruta Coincidente	Razón
447712345678	6	-	-	-	"447" → par 3	Coincidencia de prefijo más larga
441234567890	6	-	-	-	"44" → par 1	Coincidencia de prefijo, no hay ruta más específica
12125551234	6	-	-	-	"1" → par 2	Coincidencia de prefijo para números de EE. UU.
555881234567	8	-	-	-	"555" (SSN 8) → par 4	Coincidencia de GT + SSN, reescribe SSN a 6
555881234567	6	-	-	-	"555" (comodín SSN) → par X	Coincidencia de GT, sin reescritura de SSN
441234567890	6	0	1	4	"44" (TT=0) → par 1	Coincidencia de GT + TT, transforma TT a 3

GT Llamado	SSN Fuente	TT	NPI	NAI	Ruta Coincidente	Razón
12125551234	8	0	1	4	"1" (TT=0, NPI=1, NAI=4)	Más

Guía del Gateway USSD

← Volver a la Documentación Principal

Esta guía cubre el **Gateway USSD** de OmniSS7, que conecta diálogos USSD SS7/MAP a callbacks HTTP/JSON, permitiendo a los desarrolladores de terceros construir aplicaciones USSD con un simple endpoint HTTP.

Tabla de Contenidos

Descripción General

El Gateway USSD maneja dos direcciones de tráfico USSD:

- **Originado en Móvil (Entrante)** — Un suscriptor marca un código corto (por ejemplo, `*100#`). El gateway recibe el `processUnstructuredSS-Request` MAP (código de operación 59), lo reenvía a tu callback HTTP y retransmite tu respuesta de vuelta a través de SS7.
- **Originado en la Red (Saliente)** — Tu aplicación envía un mensaje USSD a un suscriptor a través de la API REST. El gateway envía un

`unstructuredSS-Request` MAP (código de operación 60) a través de SS7 y enruta la respuesta del suscriptor a tu callback.

Ambas direcciones soportan **diálogos de múltiples turnos** — menús interactivos donde el suscriptor responde y recibe mensajes de seguimiento.

Características Clave

Propiedad	Valor
Transporte	HTTP POST sincrónico por turno
Codificación	Alfabeto GSM de 7 bits por defecto (DCS 0x0F) según 3GPP TS 23.038
Longitud máxima del texto	182 caracteres (configurable)
Seguimiento de sesión	UUID generado por el gateway por diálogo
Autenticación	Ninguna (confía en la red SS7)
Enrutamiento	Coincidencia de prefijo de código corto con URLs de callback

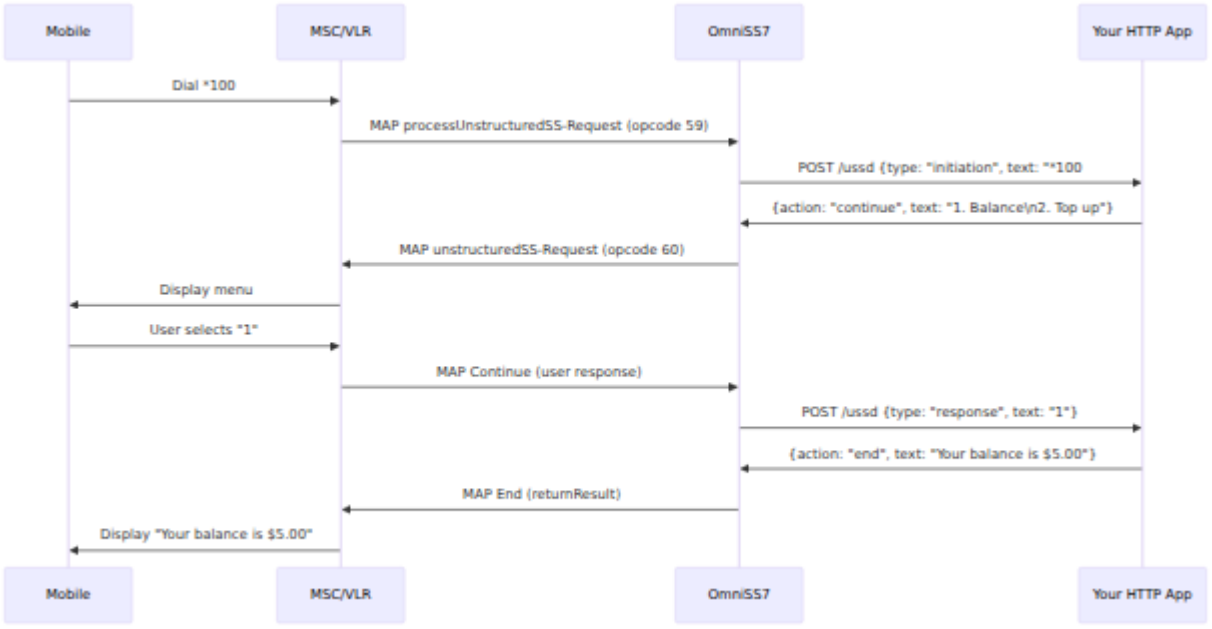
Referencias de 3GPP

Especificación	Relevancia
3GPP TS 23.090	USSD Etapa 2 — arquitectura y procedimientos
3GPP TS 24.090	USSD Etapa 3 — detalles del protocolo
3GPP TS 29.002	Protocolo MAP — USSD-Arg, USSD-Res, códigos de operación 59/60/61
3GPP TS 23.038	Alfabeto GSM de 7 bits por defecto y esquema de codificación de datos

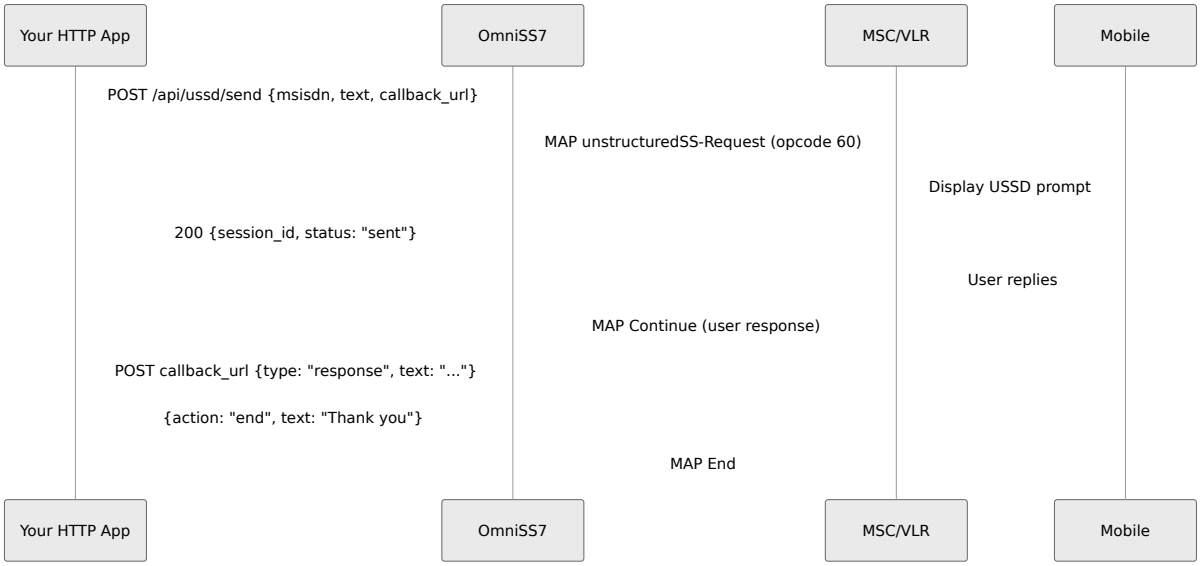
---

Arquitectura

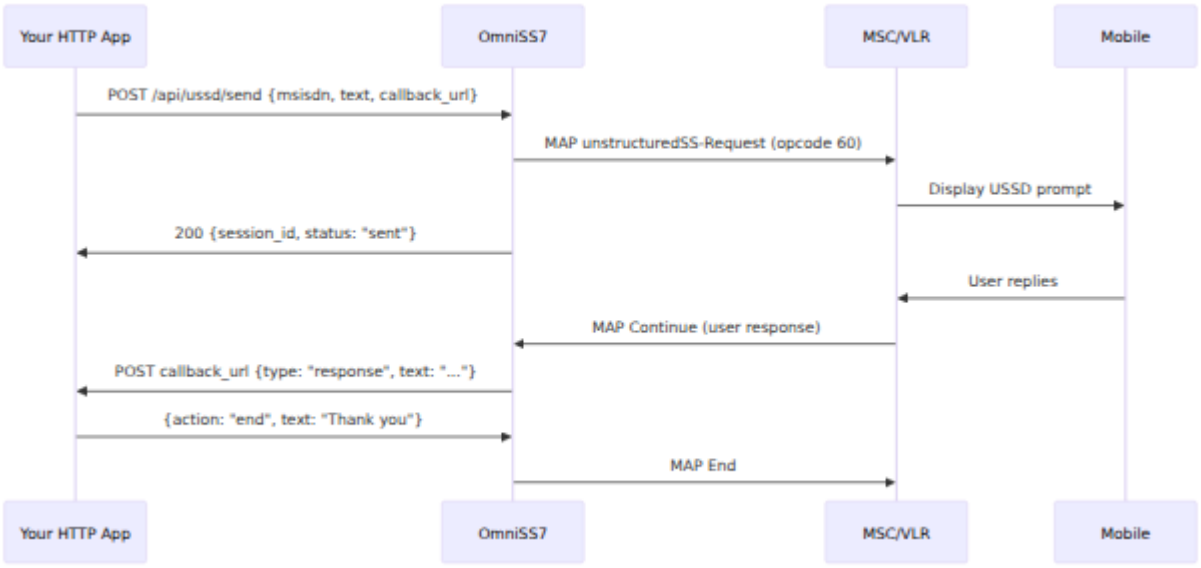
Flujo Originado en Móvil (Entrante)



Flujo Originado en la Red (Push Saliente)



Descripción General de Componentes



Habilitando el Gateway USSD

El Gateway USSD requiere que se habilite el **modo Cliente MAP**, además de su propia bandera de características.

```
config :omniss7,
  map_client_enabled: true,
  ussd_gateway_enabled: true
```

El gateway también requiere una conexión M3UA funcional (consulta la Guía del Cliente MAP para la configuración de M3UA).

Configuración

Parámetros del Gateway USSD

```
config :omniss7,
  ussd_gateway_enabled: true,
  ussd_gateway: %{
    # Enrutamiento de códigos cortos — coincidencia de prefijo más
largo
    routes: [
      %{pattern: "*100", url: "http://balance-app:9000/ussd"},
      %{pattern: "*200", url: "http://topup-app:9000/ussd"},
      %{pattern: "*",    url: "http://default-app:9000/ussd"}
    ],

    # Tiempos de espera de sesión
    session_timeout_ms: 180_000,    # Duración total de la sesión
(3 minutos)
    turn_timeout_ms: 30_000,        # Máximo tiempo de espera para
la respuesta del suscriptor por turno (30 segundos)

    # Configuración del callback HTTP
    http_timeout_ms: 5_000,         # Tiempo de espera para el
POST HTTP a tu aplicación (5 segundos)

    # Límites de texto
    max_text_length: 182            # Máximo de 7 bits GSM (se
trunca con advertencia si se excede)
  }
```

Referencia de Parámetros

Parámetro	Tipo	Requerido	Por Defecto	
`ussd_gateway_enabled`	Booleano	Sí	`false`	
`ussd_gateway.routes`	Lista de Mapas	Sí	`[]`	
`ussd_gateway.session_timeout_ms`	Entero	No	`180_000`	

Parámetro	Tipo	Requerido	Por Defecto	
`ussd_gateway.turn_timeout_ms`	Entero	No	`30_000`	
`ussd_gateway.http_timeout_ms`	Entero	No	`5_000`	
`ussd_gateway.max_text_length`	Entero	No	`182`	

Parámetro	Tipo	Requerido	Por Defecto	

Parámetros de Ruta

Cada entrada en la lista `routes` es un mapa:

Parámetro	Tipo	Requerido	Descripción
`pattern`	Cadena	Sí	Prefijo de código corto a coincidir. Usa `"*"` como un fallback general. Los prefijos más largos tienen prioridad.
`url`	Cadena	Sí	URL del endpoint HTTP para recibir POSTs de callback para códigos cortos coincidentes.

Coincidencia de Rutas

Las rutas se coinciden por **prefijo más largo primero**. Para la cadena de marcado `*100#`:

1. `"*100"` coincide (longitud 4) — seleccionada
2. `"*10"` coincide (longitud 3) — omitida, más corta
3. `"*"` coincide (longitud 1) — fallback

Si no coincide ninguna ruta, el gateway devuelve un error MAP al móvil y registra una advertencia.

Protocolo de Callback HTTP

Tu aplicación recibe solicitudes HTTP POST del gateway y responde con instrucciones JSON.

Solicitud del Gateway a Tu Aplicación

Content-Type: `application/json`

Primer turno (iniciación de sesión):

```
{
  "session_id": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
  "msisdn": "+254712345678",
  "type": "initiation",
  "text": "*100#",
  "turn": 1
}
```

Turnos subsiguientes (el suscriptor respondió):

```
{
  "session_id": "a1b2c3d4-e5f6-7890-abcd-ef1234567890",
  "msisdn": "+254712345678",
  "type": "response",
  "text": "1",
  "turn": 2
}
```

Campos de Solicitud

Campo	Tipo	Descripción
`session_id`	Cadena	UUID generado por el gateway. Único por diálogo USSD. Usa esto para correlacionar turnos.
`msisdn`	Cadena	MSISDN del suscriptor (si está disponible del mensaje MAP). Puede estar vacío para algunas redes.
`type`	Cadena	`"initiation"` para el primer turno, `"response"` para las respuestas subsiguientes del suscriptor.
`text`	Cadena	La cadena de marcado (por ejemplo, `*100#`) en la iniciación, o la entrada del suscriptor (por ejemplo, `1`) en la respuesta.
`turn`	Entero	Contador de turnos comenzando en 1. Se incrementa con cada interacción del suscriptor.

Respuesta de Tu Aplicación

Tu aplicación debe responder con JSON que contenga un `action` y `text`:

Continuar (mostrar menú, esperar entrada del suscriptor):

```
{
  "action": "continue",
  "text": "1. Balance\n2. Recarga\n3. Transferir"
}
```

Finalizar (mostrar mensaje final, cerrar sesión):

```
{
  "action": "end",
  "text": "Tu saldo es $5.00"
}
```

Campos de Respuesta

Campo	Tipo	Requerido	Descripción
`action`	Cadena	Sí	`"continue"` para mantener la sesión abierta y esperar la entrada del suscriptor, o `"end"` para mostrar un mensaje final y cerrar la sesión.
`text`	Cadena	Sí	Texto para mostrar en el dispositivo del suscriptor. La longitud máxima está gobernada por `max_text_length` (por defecto 182). Usa `\n` para saltos de línea.

USSD Originado en la Red (API Push)

Envía un mensaje USSD a un suscriptor desde tu aplicación.

Endpoint

`POST /api/ussd/send`

Solicitud

```
{
  "msisdn": "+254712345678",
  "text": "Tienes una factura pendiente. Responde 1 para pagar.",
  "callback_url": "http://billing-app:9000/ussd"
}
```

Campos de Solicitud

Campo	Tipo	Requerido	Descripción
`msisdn`	Cadena	Sí	MSISDN del suscriptor de destino en formato internacional.
`text`	Cadena	Sí	Texto inicial USSD para mostrar. Codificado como GSM de 7 bits.
`callback_url`	Cadena	Sí	URL para recibir la respuesta del suscriptor a través del protocolo de callback estándar.

Respuesta

Éxito (200 OK):

```
{
  "session_id": "xyz-789-abc-123",
  "status": "sent"
}
```

Respuestas de error:

Estado HTTP	Cuerpo	Causa
400	`{"error": "invalid request", "required": ["msisdn", "text", "callback_url"]}`	Campos requeridos faltantes
400	`{"error": "gsm7_encode_failed", ...}`	El texto contiene caracteres no en el alfabeto GSM de 7 bits
500	`{"error": "send_failed", ...}`	Fallo en el envío de M3UA (verifica la conectividad)
503	`{"error": "USSD gateway not enabled"}`	`ussd_gateway_enabled` es `false`

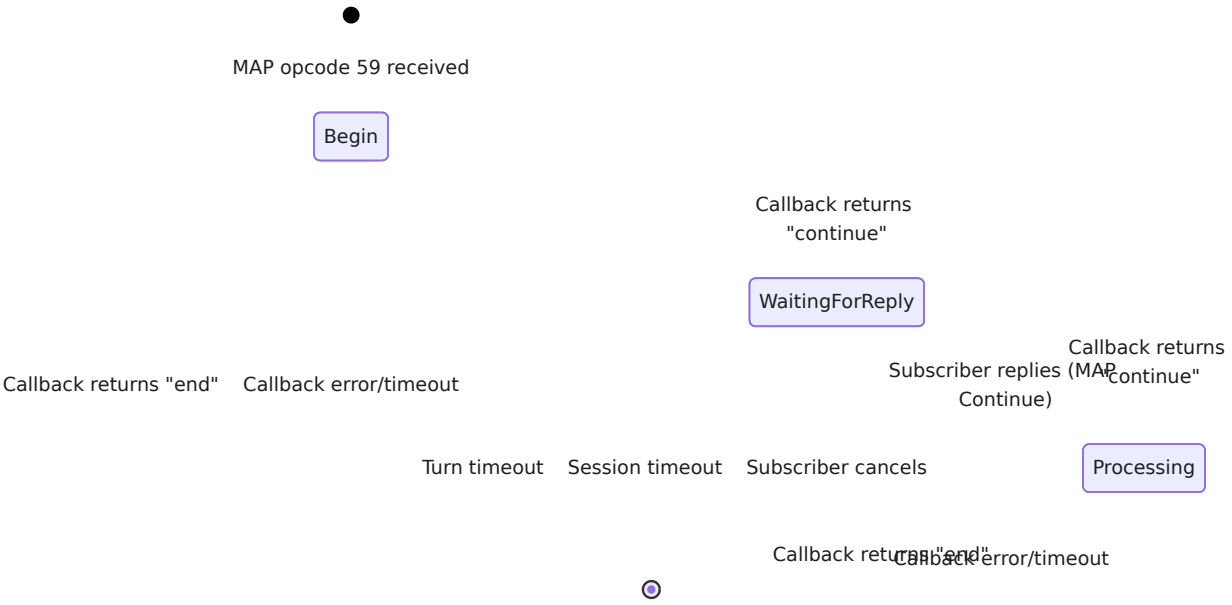
Ejemplo de cURL

```
curl -X POST http://localhost:8080/api/ussd/send \
  -H "Content-Type: application/json" \
  -d '{
    "msisdn": "+254712345678",
    "text": "Tienes una factura pendiente. Responde 1 para
pagar.",
    "callback_url": "http://billing-app:9000/ussd"
  }'
```

Ciclo de Vida de la Sesión

Cada diálogo USSD se rastrea como una **sesión** con un `session_id` único.

Estados de la Sesión



Sesión de Un Solo Turno

Si tu callback devuelve `"end"` en el primer turno, no se crea una sesión persistente. El gateway envía un MAP End con el resultado y retorna inmediatamente.

Sesión de Múltiples Turnos

Si tu callback devuelve `"continue"`, el gateway:

1. Crea un **Session GenServer** registrado en `UssdGateway.Registry`
2. Envía un MAP Continue con código de operación 60 (unstructuredSS-Request) al móvil
3. Espera la respuesta del suscriptor (hasta `turn_timeout_ms`)
4. Reenvía la respuesta a tu callback
5. Repite hasta que tu callback devuelva `"end"` o ocurra un tiempo de espera

Comportamiento de Tiempo de Espera

Tiempo de Espera	Por Defecto	Efecto
Tiempo de espera de turno	30 segundos	Si el suscriptor no responde dentro de esta ventana, la sesión se termina con un error MAP.
Tiempo de espera de sesión	3 minutos	Duración total de la sesión. Termina la sesión independientemente de la actividad.
Tiempo de espera de callback HTTP	5 segundos	Si tu aplicación no responde a tiempo, el gateway envía un error MAP al móvil y termina la sesión.

Manejo de Errores

El gateway maneja fallos de manera elegante y siempre intenta enviar una respuesta de error MAP al móvil para que el suscriptor vea un mensaje significativo en lugar de un tiempo de espera de red.

Escenario	Acción del Gateway	Código de Error MAP
Tiempo de espera de callback HTTP o 5xx	Terminar sesión, enviar MAP End con error	34 (systemFailure)
JSON inválido del callback	Terminar sesión, enviar MAP End con error	34 (systemFailure)
El texto USSD excede `max_text_length`	Truncar texto, registrar advertencia, continuar normalmente	N/A (truncado, no es un error)
Tiempo de espera del suscriptor (sin respuesta)	Terminar sesión, enviar MAP End con error	34 (systemFailure)
Ninguna ruta coincide con el código corto	Enviar MAP End con error, registrar advertencia	34 (systemFailure)
Caída del Session GenServer	La sesión muere, el suscriptor ve un tiempo de espera de red	N/A (salida del proceso)
Gateway USSD no habilitado	Devolver Facility Not Supported	21 (facilityNotSupported)

Métricas y Monitoreo

El Gateway USSD expone métricas de Prometheus en el endpoint estándar `/metrics` (puerto 8080).

Métricas USSD

Métrica: `ussd_requests_total` **Tipo**: Contador **Descripción**: Total de solicitudes USSD procesadas **Etiquetas**:

- `direction` — `"inbound"` (originado en móvil) o `"outbound"` (push originado en la red)

Métrica: `ussd_active_sessions` **Tipo**: Medidor **Descripción**: Número de sesiones USSD actualmente activas

Métrica: `map_request_duration_milliseconds` **Tipo**: Histograma **Descripción**: Duración de las operaciones de envío USSD en milisegundos **Etiquetas**:

- `operation` — `"ussd_send"` para solicitudes de push salientes

Ejemplos de Consultas Prometheus

```
# Tasa de solicitudes USSD por dirección
rate(ussd_requests_total[5m])

# Sesiones activas
ussd_active_sessions

# Latencia USSD saliente (p95)
histogram_quantile(0.95,
rate(map_request_duration_milliseconds_bucket{operation="ussd_send"}
[5m]))
```

Servidor de Callback de Ejemplo

Python (Flask)

```python
from flask import Flask, request, jsonify

app = Flask(__name__)
sessions = {}

@app.route('/ussd', methods=['POST'])
def ussd():
    data = request.json
    session_id = data['session_id']
    text = data['text']
    turn = data['turn']

    if data['type'] == 'initiation':
        sessions[session_id] = {'state': 'main_menu'}
        return jsonify({
            'action': 'continue',
            'text': '¡Bienvenido!\n1. Ver saldo\n2. Comprar tiempo
aire\n3. Transferir'
        })

    state = sessions.get(session_id, {}).get('state')

    if state == 'main_menu':
        if text == '1':
            del sessions[session_id]
            return jsonify({
                'action': 'end',
                'text': 'Tu saldo es $5.00'
            })
        elif text == '2':
            sessions[session_id]['state'] = 'buy_airtime'
            return jsonify({
                'action': 'continue',
                'text': 'Ingresa la cantidad:'
            })
        else:
            del sessions[session_id]
            return jsonify({
```

```
                'action': 'end',
                'text': 'Opción inválida. Adiós.'
            })

    elif state == 'buy_airtime':
        del sessions[session_id]
        return jsonify({
            'action': 'end',
            'text': f'Has comprado ${text} de tiempo aire.
¡Gracias!'
        })

    return jsonify({'action': 'end', 'text': 'Sesión expirada.'})

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=9000)
```

Node.js (Express)

```
const express = require('express');
const app = express();
app.use(express.json());

const sessions = new Map();

app.post('/ussd', (req, res) => {
  const { session_id, text, type } = req.body;

  if (type === 'initiation') {
    sessions.set(session_id, { state: 'main_menu' });
    return res.json({
      action: 'continue',
      text: '¡Bienvenido!\n1. Ver saldo\n2. Comprar tiempo aire'
    });
  }

  const session = sessions.get(session_id);
  if (!session) {
    return res.json({ action: 'end', text: 'Sesión expirada.' });
  }

  if (session.state === 'main_menu' && text === '1') {
    sessions.delete(session_id);
    return res.json({ action: 'end', text: 'Tu saldo es $5.00' });
  }

  sessions.delete(session_id);
  return res.json({ action: 'end', text: 'Adiós.' });
});

app.listen(9000, () => console.log('Callback USSD en el puerto
9000'));
```

Solución de Problemas

El Marcado USSD Devuelve "Servicio No Disponible"

Síntomas: El suscriptor marca un código corto y recibe inmediatamente un error de red.

Causas posibles:

- `ussd_gateway_enabled` es `false`
- Ninguna ruta coincide con el código corto marcado
- La conexión M3UA está caída

Resolución:

1. Verifica `ussd_gateway_enabled: true` en la configuración
2. Asegúrate de que un patrón de ruta coincida con el código corto (recuerda incluir `*` como fallback)
3. Verifica el estado del par M3UA en la interfaz web (página de Pares)

El Callback No Recibe Solicitudes

Síntomas: Los registros del gateway muestran el inicio de USSD pero tu aplicación nunca recibe el POST HTTP.

Causas posibles:

- La URL de callback es inalcanzable desde el host de OmniSS7
- El firewall bloquea HTTP saliente desde OmniSS7
- La aplicación de callback no está en ejecución

Resolución:

1. Prueba la conectividad: `curl -v http://your-app:9000/ussd` desde el host de OmniSS7
2. Verifica las reglas del firewall para HTTP saliente

3. Asegúrate de que tu aplicación de callback esté escuchando en el puerto configurado

Las Sesiones Expiran Prematuramente

Síntomas: Las sesiones de múltiples turnos terminan con "systemFailure" antes de que el suscriptor pueda responder.

Causas posibles:

- `turn_timeout_ms` es demasiado corto para tu base de suscriptores
- `http_timeout_ms` es demasiado corto para el tiempo de procesamiento de tu aplicación
- Latencia de red entre OmniSS7 y tu servidor de callback

Resolución:

1. Aumenta `turn_timeout_ms` (el valor por defecto de 30 segundos debería ser suficiente para la mayoría de los casos)
2. Aumenta `http_timeout_ms` si tu aplicación necesita más tiempo de procesamiento
3. Despliega el servidor de callback cerca de OmniSS7 para reducir la latencia

Errores de Codificación GSM de 7 bits

Síntomas: Errores `gsm7_encode_failed` en los registros o respuestas 400 de `/api/ussd/send`.

Causas posibles:

- El texto contiene caracteres fuera del alfabeto GSM de 7 bits por defecto (por ejemplo, emoji, caracteres CJK)

Resolución:

- Restringe el texto USSD al conjunto básico de caracteres GSM: letras ASCII, dígitos, puntuación común y algunos caracteres griegos/nórdicos

- Consulta la Sección 6.2.1 de 3GPP TS 23.038 para la tabla completa de caracteres

---

Documentación Relacionada

- **Guía de API** — Referencia completa de la API REST (todos los endpoints incluyendo `/api/ussd/send`)
- **Guía del Cliente MAP** — Configuración de conexión M3UA requerida para USSD
- **Referencia de Configuración** — Todos los parámetros de configuración
- **Guía de Características Comunes** — Interfaz web, monitoreo y configuración de Prometheus

Guía de la Interfaz Web

← Volver a la Documentación Principal

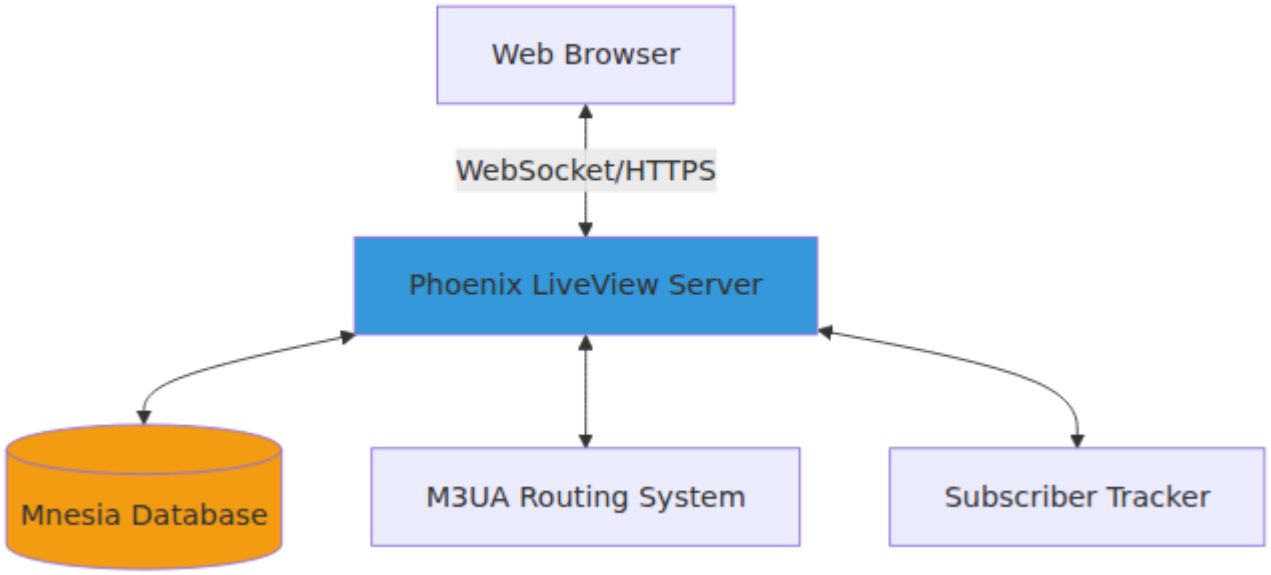
Esta guía proporciona documentación completa para el uso de la **Interfaz Web** de OmniSS7 (interfaz Phoenix LiveView).

Tabla de Contenidos

Descripción General

La Interfaz Web de OmniSS7 es una aplicación **Phoenix LiveView** que proporciona capacidades de monitoreo y gestión en tiempo real. Las páginas disponibles dependen de cuál modo operativo está activo (STP, HLR o SMSc).

Arquitectura de la Interfaz Web



Configuración del Servidor

- **Protocolo**: HTTPS
- **Puerto**: 443 (configurado en `config/runtime.exs`)
- **IP Predeterminada**: 0.0.0.0 (escucha en todas las interfaces)
- **Certificados**: Ubicados en `priv/cert/`

URL de Acceso: `https://[server-ip]:443`

Acceso a la Interfaz Web

Requisitos Previos

1. **Certificados SSL**: Asegúrese de que los certificados SSL válidos estén presentes en `priv/cert/`:
   - `omnitouch.crt` - Archivo de certificado
   - `omnitouch.pem` - Archivo de clave privada
2. **Aplicación en Ejecución**: Inicie la aplicación con `iex -S mix`

3. **Cortafuegos**: Asegúrese de que el puerto 443 esté abierto para el tráfico HTTPS

Páginas Disponibles por Modo

Página	Modo STP	Modo HLR	Modo SMSc	Descripción
Eventos SS7	✅	✅	✅	Registro de eventos y captura de mensajes SCCP
Cliente SS7	✅	✅	✅	Pruebas de operación MAP manual
M3UA	✅	✅	✅	Estado de conexión M3UA
Enrutamiento	✅	✅	✅	Gestión de tabla de enrutamiento M3UA
Prueba de Enrutamiento	✅	✅	✅	Pruebas y validación de rutas
Enlaces HLR	✅	✅	✅	Estado de API HLR y gestión de suscriptores
Suscriptores Activos	✅	✅	✅	Seguimiento de ubicación de suscriptores en tiempo real (HLR)
Enlaces SMSc	✅	✅	✅	Estado de API SMSc y gestión de colas
Suscriptores SMSc	✅	✅	✅	Seguimiento de suscriptores en tiempo real (SMSc)

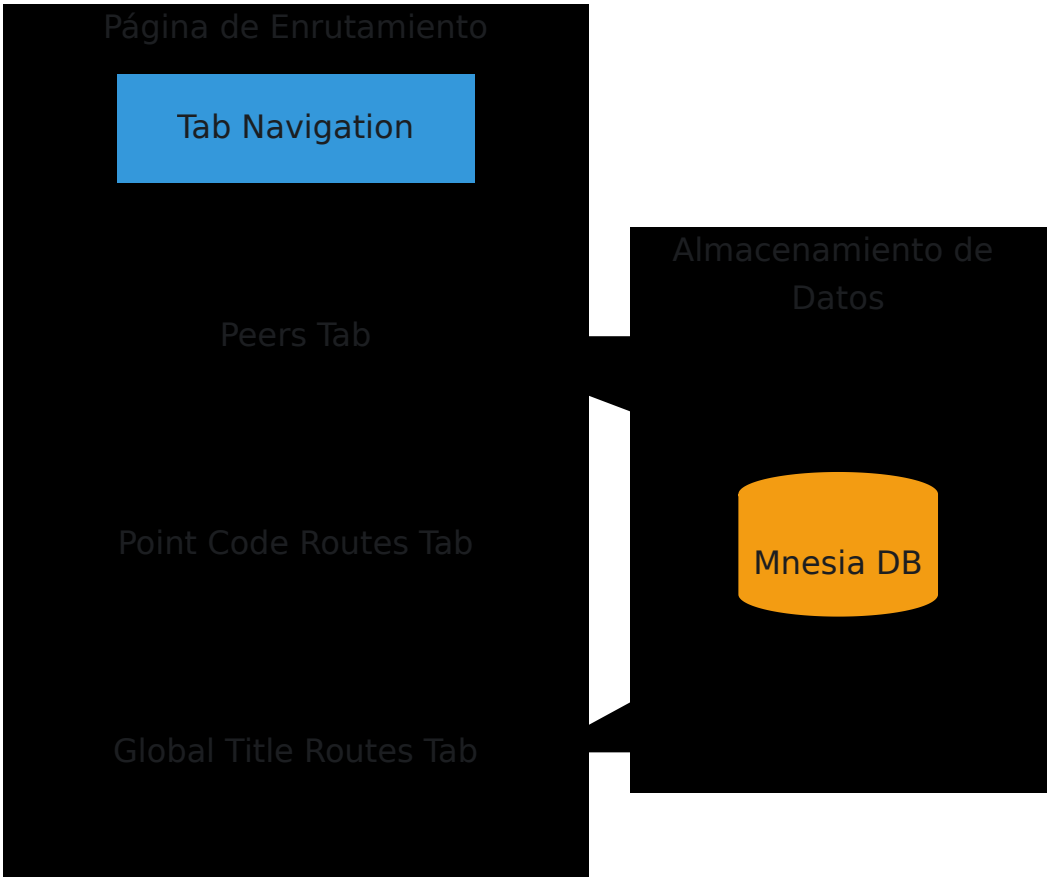
Página	Modo STP	Modo HLR	Modo SMSc	Descripción
Aplicación	□	□	□	Recursos del sistema y monitoreo
Configuración	□	□	□	Visor de configuración

Gestión de Enrutamiento

Página: `/routing`
Modos: STP, SMSc
Auto-Actualización: Cada 5 segundos

La página de gestión de enrutamiento proporciona una interfaz con pestañas para gestionar las tablas de enrutamiento M3UA.

Diseño de la Página



Pestaña de Peers

Gestione las conexiones de pares M3UA (otros STPs, HLRs, MSCs, SMSCs).

Columnas de la Tabla de Peers

Columna	Descripción	Ejemplo
ID	Identificador único del par	`1`
Nombre	Nombre legible por humanos del par	`"STP_West"`
Rol	Rol de conexión	`client`, `server`, `stp`
Point Code	Código de punto SS7 del par	`100`
Remoto	IP:Puerto remoto	`10.0.0.10:2905`
Estado	Estado de conexión	`active`, `aspup`, `down`
Acciones	Botones de Editar/Eliminar	-

Agregar un Par

1. **Haga clic** en la pestaña Peers
2. **Complete** los campos del formulario:
   - **Peer ID**: Generado automáticamente si se deja vacío
   - **Peer Name**: Nombre descriptivo (requerido)
   - **Role**: Seleccione `client`, `server` o `stp`
   - **Point Code**: Código de punto SS7 (requerido)
   - **Local IP**: Dirección IP de su sistema
   - **Local Port**: 0 para asignación de puerto dinámica
   - **Remote IP**: Dirección IP del par
   - **Remote Port**: Puerto del par (típicamente 2905)
   - **Routing Context**: ID de contexto de enrutamiento M3UA
   - **Network Indicator**: `international` o `national`
3. **Haga clic** en "Add Peer"

Persistencia: El par se guarda inmediatamente en Mnesia y sobrevive al reinicio.

Editar un Par

1. **Haga clic** en el botón "Edit" en la fila del par
2. **Modifique** los campos del formulario según sea necesario
3. **Haga clic** en "Update Peer"

Nota: Si cambia el Peer ID, el par antiguo se elimina y se crea uno nuevo.

Eliminar un Par

1. **Haga clic** en el botón "Delete" en la fila del par
2. **Confirme** la eliminación (todas las rutas que utilizan este par también serán eliminadas)

Indicadores de Estado del Par

Estado	Color	Descripción
`active`	Verde	El par está conectado y enrutando mensajes
`aspup`	Amarillo	ASP está activo pero aún no está en funcionamiento
`down`	Rojo	El par está desconectado

Pestaña de Rutas de Código de Punto

Configure reglas de enrutamiento basadas en Códigos de Punto de destino.

Columnas de la Tabla de Rutas

Columna	Descripción	Ejemplo
Destino PC	Código de punto objetivo (formato zona.área.id)	`1.2.3` (100)
Máscara	Máscara de subred para coincidencia de PC	`/14` (exacto), `/8` (rango)
Peer ID	Par objetivo para esta ruta	`1`
Peer Name	Nombre del par objetivo	`"STP_West"`
Prioridad	Prioridad de la ruta (1 = más alta)	`1`
Red	Indicador de red	`international`
Acciones	Botones de Editar/Eliminar	-

Agregar una Ruta de Código de Punto

1. **Haga clic** en la pestaña "Point Code Routes"
2. **Complete** los campos del formulario:
   - **Código de Punto de Destino**: Ingrese como `zona.área.id` (por ejemplo, `1.2.3`) o como entero (0-16383)
   - **Máscara**: Seleccione la máscara `/14` para coincidencia exacta, valores más bajos para rangos
   - **Peer ID**: Seleccione el par objetivo del menú desplegable
   - **Prioridad**: Ingrese la prioridad (1 = más alta, predeterminado)
   - **Indicador de Red**: Seleccione `international` o `national`
3. **Haga clic** en "Add Route"

Formato de Código de Punto: Puede ingresar códigos de punto en dos formatos:

- **Formato 3-8-3**: `zona.área.id` (por ejemplo, `1.2.3`)
- **Formato Entero**: `0-16383` (por ejemplo, `1100`)

El sistema convierte automáticamente entre formatos.

Entendiendo las Máscaras

Los códigos de punto son valores de 14 bits (0-16383). La máscara especifica cuántos bits más significativos deben coincidir:

Máscara	PCs Coincidentes	Caso de Uso
`/14`	1 (coincidencia exacta)	Ruta a destino específico
`/13`	2 PCs	Pequeño rango
`/8`	64 PCs	Rango medio
`/0`	Todos los 16,384 PCs	**Ruta predeterminada/de respaldo**

Ejemplos:

- `PC 1000 /14` → Coincide solo con PC 1000
- `PC 1000 /8` → Coincide con PC 1000-1063 (64 PCs consecutivos)
- `PC 0 /0` → Coincide con todos los códigos de punto (ruta predeterminada)

Tarjeta de Referencia de Máscara de Código de Punto

La página web incluye una referencia interactiva que muestra todos los valores de máscara y sus rangos.

Pestaña de Rutas de Título Global

Configure reglas de enrutamiento basadas en direcciones de Título Global SCCP.

Requisito: El enrutamiento de Título Global debe estar habilitado en la configuración:

```
config :omniss7,
  enable_gt_routing: true
```

Columnas de la Tabla de Rutas

Columna	Descripción	Ejemplo
GT Prefix	Prefijo GT de la parte llamada (vacío = respaldo)	`"1234"`, `""`
Source SSN	Coincidencia en SSN de la parte llamada (opcional)	`6` (HLR), `any`
Peer ID	Par objetivo	`1`
Peer	Nombre del par	`"HLR_West (1)"`
Dest SSN	Reescribir SSN al reenviar (opcional)	`6`, `preserve`
Prioridad	Prioridad de la ruta	`1`
Descripción	Descripción de la ruta	`"Números de EE.UU."`
Acciones	Botones de Editar/Eliminar	-

Agregar una Ruta de Título Global

1. **Haga clic** en la pestaña "Global Title Routes"
2. **Complete** los campos del formulario:
   - **GT Prefix**: Deje vacío para ruta de respaldo, o ingrese dígitos (por ejemplo, `"1234"`)
   - **Source SSN**: Opcional - filtrar por SSN de la parte llamada
   - **Peer ID**: Seleccione el par objetivo
   - **Dest SSN**: Opcional - reescribir SSN al reenviar
   - **Prioridad**: Prioridad de la ruta (1 = más alta)

- **Descripción**: Descripción legible por humanos

3. **Haga clic** en "Add Route"

Rutas de Respaldo: Si el GT Prefix está vacío, la ruta actúa como un catch-all para GTs que no coinciden con ninguna otra ruta.

Valores Comunes de SSN

La página incluye una tarjeta de referencia con valores comunes de SSN:

SSN	Elemento de Red
6	HLR (Registro de Ubicación de Hogar)
7	VLR (Registro de Ubicación de Visitantes)
8	MSC (Centro de Conmutación Móvil)
9	EIR (Registro de Identidad de Equipos)
10	AUC (Centro de Autenticación)
142	RANAP
145	gsmSCF (Función de Control de Servicio)
146	SGSN

Reescritura de SSN

- **Source SSN**: Coincidencia en el SSN de la Parte Llamante en los mensajes entrantes
- **Dest SSN**: Si se establece, reescribe el SSN de la Parte Llamante al reenviar
  - Vacío = preservar el SSN original
  - Valor = reemplazar con este SSN

Caso de Uso: Enrutar mensajes con SSN=6 (HLR) a un par, y reescribir a SSN=7 (VLR) en el lado saliente.

Persistencia de la Tabla de Enrutamiento

Todas las rutas se almacenan en Mnesia y sobreviven a los reinicios de la aplicación.

Cómo Persisten las Rutas

1. **Cambios en la Interfaz Web**: Todas las operaciones de agregar/editar/eliminar se guardan inmediatamente en Mnesia
2. **Reinicio de la Aplicación**: Las rutas se cargan desde Mnesia al iniciar
3. **Fusión de runtime.exs**: Las rutas estáticas de `config/runtime.exs` se fusionan con las rutas de Mnesia (sin duplicados)

Prioridad de Ruta

Cuando múltiples rutas coinciden con un destino:

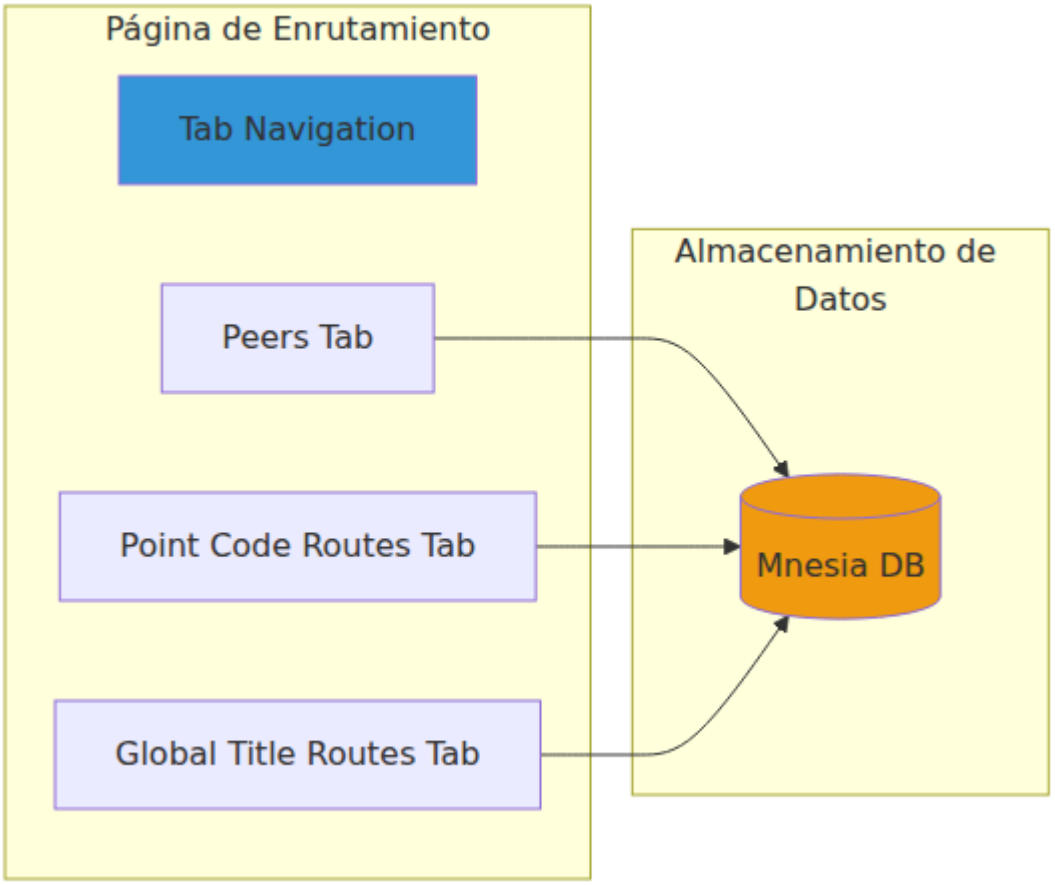
1. **Más Específico Primero**: Los valores de máscara más altos (más específicos) tienen prioridad
2. **Campo de Prioridad**: Los números de prioridad más bajos enrutaran primero (1 = mayor prioridad)
3. **Estado del Par**: Solo se utilizan rutas a pares `active`

Suscriptores Activos

Página: `/subscribers`
Modo: Solo HLR
Auto-Actualización: Cada 2 segundos

Muestra el seguimiento en tiempo real de los suscriptores que han enviado solicitudes de UpdateLocation.

Características de la Página



Columnas de la Tabla de Suscriptores

Columna	Descripción	Ejemplo
IMSI	IMSI del suscriptor	`"50557123456789"`
Número VLR	Dirección GT VLR actual	`"555123155"`
Número MSC	Dirección GT MSC actual	`"555123155"`
Actualizado En	Marca de tiempo de la última UpdateLocation	`"2025-10-25 14:23:45 UTC"`
Duración	Tiempo desde el registro	`"2h 15m 34s"`

Resumen de Estadísticas

Cuando hay suscriptores presentes, se muestra una tarjeta de resumen:

- **Total Activo**: Número total de suscriptores registrados
- **VLRs Únicos**: Número de direcciones VLR distintas
- **MSCs Únicos**: Número de direcciones MSC distintas

Limpiar Suscriptores

Botón Limpiar Todo: Elimina todos los suscriptores activos del rastreador.

Confirmación: Requiere confirmación antes de limpiar (no se puede deshacer).

Caso de Uso: Limpiar registros de suscriptores obsoletos después del mantenimiento de la red o pruebas.

Auto-Actualización

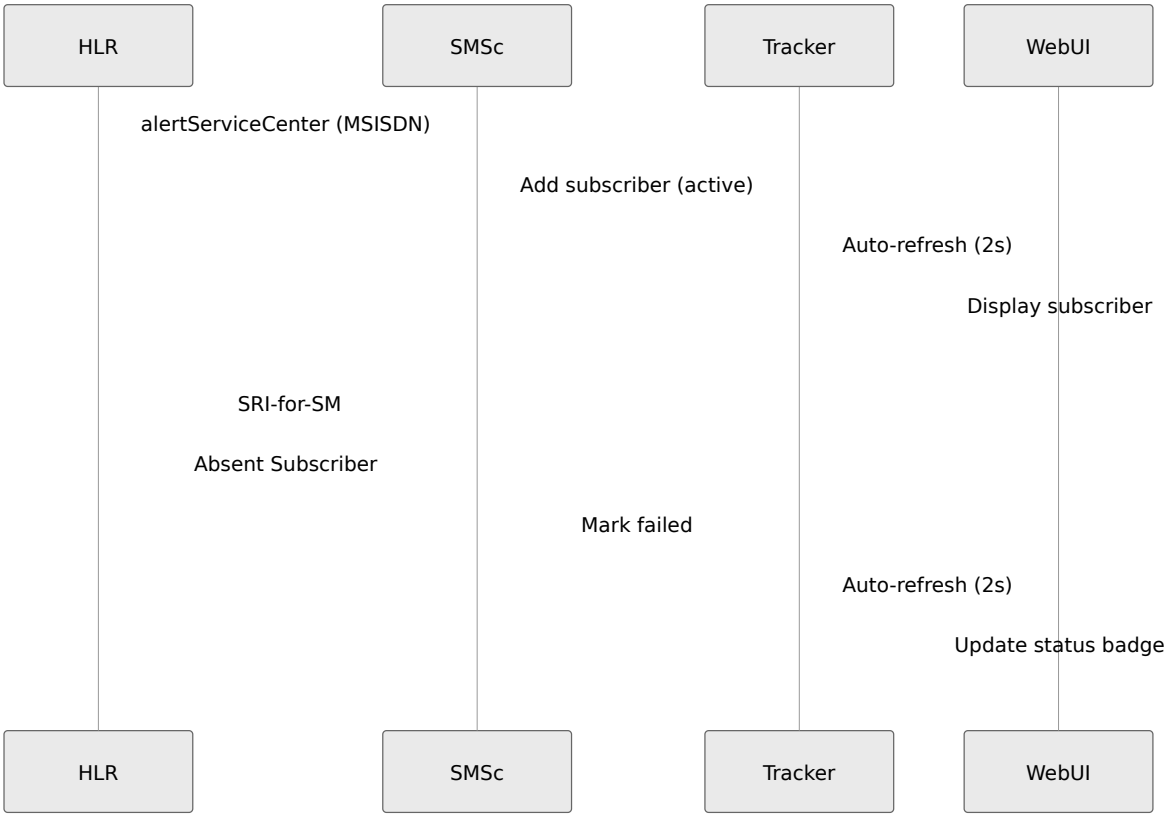
La página se actualiza automáticamente cada **2 segundos** para mostrar actualizaciones en tiempo real de los suscriptores.

Suscriptores SMSc

Página: `/smsc_subscribers`
Modo: Solo SMSc
Auto-Actualización: Cada 2 segundos

Muestra el seguimiento en tiempo real de los suscriptores basado en mensajes alertServiceCenter recibidos de HLRs, estado de entrega de mensajes y seguimiento de fallos.

Características de la Página



Columnas de la Tabla de Suscriptores

Columna	Descripción	Ejemplo
MSISDN	Número de teléfono del suscriptor	`"15551234567"`
IMSI	IMSI del suscriptor	`"001010123456789"`
HLR GT	HLR GT que envió alertServiceCenter	`"15551111111"`
Msgs Enviados	Conteo de mensajes MT-FSM enviados	`5`
Msgs Recibidos	Conteo de mensajes MO-FSM recibidos	`2`
Estado	`Active` o `Failed` (codificado por color)	`● Active`
Última Actualización	Marca de tiempo de la última actualización	`"2025-10-30 14:23:45 UTC"`
Duración	Tiempo desde la última actualización	`"15m 34s"`

Indicadores de Estado

- ● **Active** (Verde): El suscriptor es alcanzable, última alertServiceCenter recibida con éxito
- ○ **Failed** (Rojo): El último intento de entrega falló (error SRI-for-SM o suscriptor ausente)

Resumen de Estadísticas

Cuando hay suscriptores presentes, se muestra una tarjeta de resumen:

- **Total Rastreado**: Número total de suscriptores rastreados
- **Activos**: Número de suscriptores con estado activo
- **Fallidos**: Número de suscriptores con estado fallido
- **HLRs Únicos**: Número de HLRs distintos que envían alertas

Gestión de Suscriptores

Botón Eliminar: Elimina a un suscriptor individual del seguimiento.

Botón Limpiar Todo: Elimina todos los suscriptores rastreados.

Confirmación: Limpiar Todo requiere confirmación antes de limpiar (no se puede deshacer).

Caso de Uso:

- Eliminar entradas obsoletas después de problemas en la red
- Limpiar datos de prueba después del desarrollo
- Monitorear qué HLRs están enviando alertas

Contadores de Mensajes

El rastreador incrementa automáticamente los contadores:

- **Mensajes Enviados**: Incrementado cuando SRI-for-SM tiene éxito y se envía MT-FSM
- **Mensajes Recibidos**: Incrementado cuando se recibe MO-FSM del suscriptor

Auto-Actualización

La página se actualiza automáticamente cada **2 segundos** para mostrar actualizaciones en tiempo real de suscriptores y estados.

Operaciones Comunes

Búsqueda y Filtrado

Actualmente, la Interfaz Web no incluye funcionalidad de búsqueda/filtrado integrada. Para encontrar rutas específicas:

1. Use la función de búsqueda de su navegador (Ctrl+F / Cmd+F)
2. Busque nombres de pares, códigos de punto o prefijos GT

Operaciones Masivas

Para realizar cambios masivos en rutas:

1. **Opción 1**: Use la API REST para acceso programático
2. **Opción 2**: Edite `config/runtime.exs` y reinicie la aplicación
3. **Opción 3**: Use la Interfaz Web para cambios individuales en rutas

Exportar/Importar

Nota: La Interfaz Web actualmente no admite la exportación o importación de tablas de enrutamiento. Las rutas son:

- Almacenadas en archivos de base de datos Mnesia
- Configuradas en `config/runtime.exs`

Para respaldar rutas:

1. **Mnesia**: Respaldar el directorio `Mnesia.{node_name}/`
2. **Configuración**: Control de versiones de `config/runtime.exs`

Comportamiento de Auto-Actualización

Diferentes páginas tienen diferentes intervalos de actualización:

Página	Intervalo de Actualización	Razón
Gestión de Enrutamiento	5 segundos	Los cambios de ruta son infrecuentes
Suscriptores Activos	2 segundos	El estado de los suscriptores cambia con frecuencia
Estado de M3UA	Varía según la página	Monitoreo del estado de conexión

Conexión WebSocket: Todas las páginas utilizan conexiones WebSocket de Phoenix LiveView para actualizaciones en tiempo real.

Interrupción de Red: Si se pierde la conexión WebSocket, la página intentará reconectarse automáticamente.

Solución de Problemas

Página No Cargando

1. **Verifique el Certificado HTTPS**: Asegúrese de que `priv/cert/omnitouch.crt` y `.pem` estén presentes
2. **Verifique el Puerto 443**: Compruebe que las reglas del cortafuegos permitan tráfico HTTPS
3. **Aplicación en Ejecución**: Confirme que la aplicación esté en ejecución con `iex -S mix`

4. **Consola del Navegador**: Verifique si hay errores de certificado SSL (advertencias de certificado autofirmado)

Rutas No Persistentes

1. **Verifique el Almacenamiento de Mnesia**: Verifique `mnesia_storage_type: :disc_copies` en la configuración
2. **Directorio de Mnesia**: Asegúrese de que el directorio de Mnesia sea escribible
3. **Verifique los Registros**: Busque errores de Mnesia en los registros de la aplicación

Auto-Actualización No Funciona

1. **Conexión WebSocket**: Verifique la consola del navegador en busca de errores de WebSocket
2. **Red**: Verifique la conexión de red estable
3. **Recargar Página**: Intente actualizar la página (F5)

Documentación Relacionada

- **Guía STP** - Configuración de enrutamiento detallada
- **Guía HLR** - Gestión de suscriptores
- **Guía API** - API REST para acceso programático
- **Referencia de Configuración** - Todos los parámetros de configuración

---

Resumen

La Interfaz Web de OmniSS7 proporciona gestión intuitiva y en tiempo real de tablas de enrutamiento y seguimiento de suscriptores:

 Actualizaciones en Tiempo Real - La auto-actualización mantiene los datos actualizados

 Almacenamiento Persistente - Mnesia asegura que las rutas sobrevivan a los reinicios
 Interfaz Basada en Roles - Las páginas se adaptan al modo operativo (STP/HLR/SMSc)
 Gestión Interactiva - Agregar, editar, eliminar rutas sin reinicio
 Monitoreo de Estado - Estado de conexión y de pares en vivo

Para operaciones avanzadas o automatización, consulte la Guía API.