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

1. Descripción General
2. Configuración del Servidor HTTP
3. Swagger UI
4. Puntos Finales de la API
5. Ejemplos Trabajados
6. Autenticación
7. Formatos de Respuesta
8. Manejo de Errores
9. Métricas (Prometheus)
10. Ejemplos de Solicitudes

Descripción General

OmniSS7 expone una API REST en **el puerto 9090** para realizar operaciones de MAP (Mobile Application Part) y CAP (CAMEL Application Part) sobre la red SS7, además de inyección de SCCP/M3UA en bruto para pruebas. La API define **47 puntos finales de operación** `POST /api/*` junto con los puntos finales de métricas y Swagger. Permite:

- Activar solicitudes MAP (movilidad, SMS, autenticación, datos del suscriptor, servicios suplementarios, servicios de localización, GPRS, equipos)
- Realizar diálogos de control de llamadas CAP/CAMEL

- Inyectar cargas útiles SCCP/M3UA en bruto
- Monitorear métricas del sistema a través de Prometheus

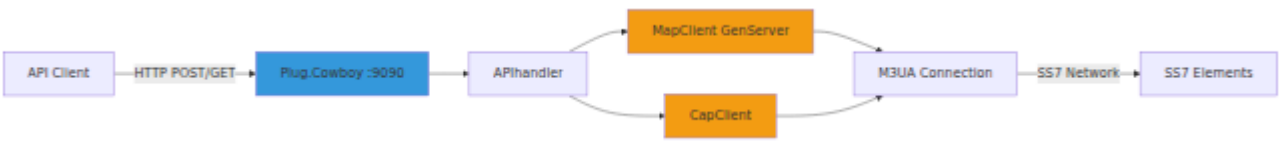
Para esquemas de solicitud/respuesta por operación y códigos de operación MAP, esta guía hace referencia cruzada a la **Guía del Cliente MAP**, que documenta cada operación en detalle.

Puertos y Transportes

OmniSS7 ejecuta **tres servidores HTTP distintos**. Esta guía cubre el primero (:9090). Los otros están documentados en la Referencia de Configuración.

Puerto	Transporte	Servidor	Alcance de esta guía
9090	HTTP (`Plug.Cowboy`)	`APIhandler`	**Sí.** MAP/CAP REST `/api/*`, `/metrics`, `/swagger`.
8087	HTTPS (`Bandit`, Phoenix)	`ControlPanelWeb.Endpoint`	No: interfaz de usuario del panel d control web. Ver Guía de UI Web.
8445	HTTPS / TLS	`:api_ex` controladores REST	No: estado/ruteo/enlac REST. Ver Referenc de Configuración.

Arquitectura de la API



Configuración del Servidor HTTP

Detalles del Servidor

Los puntos finales `/api/*`, `/metrics`, y `/swagger` son servidos por un oyente HTTP `Plug.Cowboy`.

Parámetro	Valor	Configurable
Protocolo	HTTP	No
Dirección IP	0.0.0.0 (todas las interfaces)	Solo a través de código
Puerto	9090	Sí: `config :omniss7, api_port: <port>` (por defecto `9090`)
Transporte	`Plug.Cowboy`	No

URL de Acceso: `http://[server-ip]:9090`

> El panel de control (:8087, Bandit) y los controladores REST `:api_ex` (:8445) son servidores separados. Ver Puertos y Transportes.

Habilitar/Deshabilitar el Servidor HTTP

Controla si el servidor HTTP :9090 se inicia:

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

Por Defecto: `true` (habilitado)

Cuando Está Deshabilitado: El servidor HTTP :9090 no se iniciará, y la API REST / Swagger UI / punto final de métricas no estará disponible. Los servidores

del panel de control (:8087) y `:api_ex` (:8445) no se ven afectados (están controlados por `start_web_interfaces`).

Swagger UI

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

Accediendo a Swagger UI

URL: `http://[server-ip]:9090/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]:9090/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

Cada punto final de operación sigue el patrón `POST /api/{operation}` y acepta un cuerpo de solicitud JSON. Las solicitudes son **sincrónicas**: el controlador bloquea hasta que llega la respuesta SS7 o se agota el tiempo de operación (por defecto **10 segundos** para operaciones MAP esperadas; ver Manejo de Errores).

Las tablas a continuación enumeran todos los **47** puntos finales de operación agrupados por categoría. Para cada campo de solicitud de operación, esquema de respuesta y código de operación MAP, consulte la sección correspondiente de la **Guía del Cliente MAP**, que fue actualizada junto con esta guía y documenta cada operación en su totalidad.

> **Códigos de respuesta (todos los puntos finales):** `200` = éxito, `400` = solicitud incorrecta/inválida (campos faltantes o de tipo incorrecto), `504` = tiempo de espera (sin respuesta SS7 dentro del tiempo de espera de la operación), `500` = error de operación (no por tiempo de espera), `503` = función no habilitada (USSD), `404` = ruta desconocida. Ver Manejo de Errores.

Gestión de Movilidad

Operación	Punto Final	Descripción
Actualizar Ubicación	`POST /api/updateLocation`	Registrar el VLR que sirve a un suscriptor con el HLR. En modo HLR, esto activa la secuencia InsertSubscriberData (ISD).
Cancelar Ubicación	`POST /api/cancelLocation`	Instruir a un VLR para eliminar un registro de suscriptor.
Proporcionar Número de Roaming	`POST /api/prn`	Obtener un MSRN del MSC que sirve.
Purgar MS	`POST /api/purgeMS`	Marcar a un suscriptor como purgado en el HLR.
Enviar Identificación	`POST /api/sendIdentification`	Recuperar datos IMSI/autenticación del VLR anterior (TMSI proporcionado como hex).
Reiniciar	`POST /api/reset`	Reinicio iniciado por el HLR hacia un VLR (no confirmado: devuelve `{"status":"sent"}`).
Restaurar Datos	`POST /api/restoreData`	El VLR solicita la restauración de datos del suscriptor desde el HLR.

SMS

Operación	Punto Final	Descripción
Enviar Información de Enrutamiento para SM	`POST /api/sri-for-sm`	Consultar al HLR sobre el nodo que sirve para la entrega de SMS.
MT-Forward SM	`POST /api/MT-forwardSM`	Entregar un SM terminado en móvil al MSC/SGSN que sirve (`smsPDU` como hex).
MO-Forward SM	`POST /api/forwardSM`	Enviar un SM originado en móvil (`smsPDU` como hex).
Enviar SM	`POST /api/sendSM`	Construir y entregar un SMS-DELIVER desde texto GSM-7.
Construir PDU de Entrega	`POST /api/deliverPDU`	Construir un PDU SMS-DELIVER y devolver el hex (sin envío SS7).
Reportar Estado de Entrega de SM	`POST /api/reportSM-DeliveryStatus`	Reportar el resultado de entrega de SM al HLR.
Listo para SM	`POST /api/readyForSM`	Notificar al HLR que un MS es alcanzable para SMS.
Alertar Centro de Servicio	`POST /api/alertServiceCentre`	Alertar a un SMSC que un suscriptor está disponible.

Autenticación

Operación	Punto Final	Descripción
Enviar Información de Autenticación	`POST /api/send-auth-info`	Recuperar vectores de autenticación del HLR.
Enviar IMSI	`POST /api/sendIMSI`	Resolver un MSISDN a su IMSI en el HLR.

Datos del Suscriptor

Operación	Punto Final	Descripci
Enviar Información de Enrutamiento	`POST /api/sri`	Consultar al HLR información de enrutamiento de llamadas de voz
Proporcionar Información del Suscriptor	`POST /api/provideSubscriberInfo`	Solicitar estado/localizaci suscriptor desde
Eliminar Datos del Suscriptor	`POST /api/deleteSubscriberData`	Eliminar datos d suscriptor en el (banderas `witho` opcionales).
Interrogación en Cualquier Momento	`POST /api/anyTimeInterrogation`	gsmSCF consulta información/loca del suscriptor en HLR.
Interrogación de Suscripción en Cualquier Momento	`POST /api/anyTimeSubscriptionInterrogation`	Consultar datos suscripción en e
Modificación en Cualquier Momento	`POST /api/anyTimeModification`	Modificar datos suscripci��n en HLR.
Notar Datos del	`POST /api/noteSubscriberDataModified`	El HLR notifica a gsmSCF sobre d cambiados.

Operación	Punto Final	Descripci
Suscriptor Modificados		
Notar Evento MM	`POST /api/noteMM-Event`	Reportar un eve gestión de movi gsmSCF.

Servicios Suplementarios

Operación	Punto Final	Descripción
Interrogar SS	`POST /api/interrogateSS`	Interrogar el estado de un servicio suplementario.
Registrar SS	`POST /api/registerSS`	Registrar un servicio suplementario (por ejemplo, desvío de llamadas / CFU).
Borrar SS	`POST /api/eraseSS`	Borrar un servicio suplementario.
Activar SS	`POST /api/activateSS`	Activar un servicio suplementario.
Desactivar SS	`POST /api/deactivateSS`	Desactivar un servicio suplementario.
Registrar Contraseña	`POST /api/registerPassword`	Registrar una contraseña de SS (`ss_code` entero).
Obtener Contraseña	`POST /api/getPassword`	Recuperar/verificar una contraseña de SS.
Indicación de Comprobación de SS Reenviada	`POST /api/forwardCheckSS-Indication`	Comprobación no confirmada hacia un VLR (devuelve `{"status":"sent"}`).

Las cinco operaciones centrales de SS (`interrogateSS`, `registerSS`, `eraseSS`, `activateSS`, `deactivateSS`) comparten una forma de solicitud: `ss_code` (entero) y `hlr_gt`, con `forwarded_to`, `no_reply_time`, y `basic_service` opcionales.

Servicios de Localización (LCS)

Operación	Punto Final	Descripción
Enviar Información de Enrutamiento para LCS	`POST /api/sendRoutingInfoForLCS`	GMLC consulta al HLR sobre enrutamiento LCS.
Proporcionar Ubicación del Suscriptor	`POST /api/provideSubscriberLocation`	Solicitar la ubicación de un suscriptor desde el nodo que sirve.
Informe de Ubicación del Suscriptor	`POST /api/subscriberLocationReport`	Reportar la ubicación de un suscriptor al GMLC.

GPRS

Operación	Punto Final	Descripción
Actualizar Ubicación GPRS	`POST /api/updateGprsLocation`	Registrar el SGSN que sirve a un suscriptor con el HLR.
Enviar Información de Enrutamiento para GPRS	`POST /api/sendRoutingInfoForGprs`	Consultar al HLR sobre enrutamiento GPRS (selección de GGSN).

Equipos

Operación	Punto Final	Descripción
Comprobar IMEI	`POST /api/checkIMEI`	Consultar el estado del equipo en el EIR.

USSD

Operación	Punto Final	Descripción
Enviar USSD (originado en la red)	`POST /api/ussd/send`	Enviar un mensaje USSD a un suscriptor. Requiere `ussd_gateway_enabled: true` (de lo contrario `503`). Ver la Guía del Gateway USSD.

CAP / CAMEL

Operación	Punto Final	Descripción
CAP InitialDP	`POST /api/cap/initialDP`	Activar un diálogo CAMEL hacia un gsmSCF. Devuelve un `otid` para operaciones de seguimiento.
CAP Connect	`POST /api/cap/connect`	Enrutar la llamada a un destino (`destination_number` + `otid`).
CAP Continue	`POST /api/cap/continue`	Continuar el procesamiento de la llamada (`otid`).
CAP Release Call	`POST /api/cap/releaseCall`	Liberar la llamada con una `cause` (`otid`).
CAP Apply Charging	`POST /api/cap/applyCharging`	Aplicar una ventana de cobro (`duration` + `otid`).

Las operaciones de seguimiento de CAP llevan el diálogo `otid` (hex, devuelto por `initialDP`) y responden con `{"status":"sent","otid":"<hex>"}`. Ver la Guía del Gateway CAMEL.

Inyección en Bruto (pruebas / diagnósticos)

Operación	Punto Final	Descripción
SCCP en Bruto	`POST /api/raw/sccp`	Inyectar una carga útil SCCP construida a mano (hex). Sobrescrituras SCCP opcionales (parte llamada/llamante, SSNs, OPC/DPC).
M3UA en Bruto	`POST /api/raw/m3ua`	Inyectar una carga útil M3UA construida a mano (hex). Sobrescrituras OPC/DPC/contexto de enrutamiento opcionales.

Puntos Finales de Utilidad

Punto Final	Método	Propósito
`/metrics`	GET	Métricas de Prometheus (formato de texto).
`/swagger`	GET	Swagger UI.
`/swagger.json`	GET	Especificación OpenAPI.

Ejemplos Trabajados

Los siguientes ejemplos ilustran la forma de solicitud/respuesta para operaciones representativas. Para la referencia completa de campos de cada operación, consulte la Guía del Cliente MAP.

SendRoutingInfo (SRI)

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

Punto Final: `POST /api/sri`

Cuerpo de Solicitud:

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

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

Respuesta (200 OK):

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

Ejemplo de cURL:

```
curl -X POST http://localhost:9090/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 Solicitud:

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

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

Ejemplo de cURL:

```
curl -X POST http://localhost:9090/api/sri-for-sm \
  -H "Content-Type: application/json" \
  -d '{"msisdn": "1234567890", "service_center": "5551234567"}'
```

UpdateLocation

Notificar al HLR de un cambio de ubicación de un suscriptor (registro en VLR). En modo HLR, esto activa la secuencia InsertSubscriberData (ISD).

Punto Final: `POST /api/updateLocation`

Cuerpo de Solicitud:

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

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

USSD Send (Originado en la Red)

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

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

Cuerpo de Solicitud:

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

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

Respuesta (200 OK):

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

Error (503, gateway deshabilitado):

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

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 - Campos de solicitud inválidos o faltantes
- 500 Internal Server Error - La operación falló (error no por tiempo de espera devuelto por la pila SS7)
- 503 Service Unavailable - Función no habilitada (gateway USSD)
- 504 Gateway Timeout - La operación superó su tiempo de espera (10 segundos por defecto)
- 404 Not Found - Punto final inválido

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/inválidos	400	Campos requeridos faltantes o de tipo incorrecto (las respuestas a menudo incluyen una lista de `"required"`)	Incluya todos los parámetros requeridos con tipos correctos
Error de operación	500	La pila SS7 devolvió un error no por tiempo de espera	Inspeccione el mensaje de `"error"`; verifique los registros de eventos SS7
Función deshabilitada	503	Gateway USSD no habilitado	Establezca `ussd_gateway_enabled: true`
Tiempo de espera	504	La operación excedió su tiempo de espera (10s por defecto)	Verifique la conectividad M3UA, disponibilidad HLR/VLR

Error	Código HTTP	Descripción	Solución
No Encontrado	404	Punto final inválido	Verifique la URL del punto final

Comportamiento de Tiempo de Espera

Las operaciones esperadas utilizan un **tiempo de espera de 10 segundos** por defecto:

1. Solicitud enviada al MapClient GenServer
2. Espera la 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 la conexión M3UA (UI Web → página M3UA)
- Verifique que el elemento de red (HLR/VLR/MSC) sea alcanzable
- 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]:9090/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 TID
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

Las métricas están agrupadas por capa de protocolo / subsistema. Todas se declaran al inicio y se emiten con ayudantes protegidos, por lo que una métrica no registrada nunca interrumpe un camino caliente de protocolo.

Aplicación (MAP / CAP / USSD / licencia)

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

SCCP

Métrica	Tipo	Etiquetas	
`sccp_messages_total`	Contador	`message_type`, `direction`	Mensajes (`udt`/`xud` y direcció
`sccp_decode_errors_total`	Contador	-	Mensajes decodific
`sccp_segments_received_total`	Contador	-	Segment reensaml
`sccp_reassembly_completed_total`	Contador	-	Mensajes completa
`sccp_reassembly_timeouts_total`	Contador	-	Buffers d temporiz

TCAP

Métrica	Tipo	Etiquetas	
`tcap_dialogues_total`	Contador	`type`	Diálogos TCAP por tip
`tcap_aborts_total`	Contador	`reason`	Mensajes de abortos `unrecognized_messag` (o `p_abort_other`); U
`tcap_decode_errors_total`	Contador	-	Mensajes TCAP de un
`tcap_active_dialogues`	Gauge	-	Transacciones MAP ac

SCTP

Métrica	Tipo	Etiquetas	
`sctp_association_state`	Gauge	`remote_ip`, `remote_port`	Estac inacti
`sctp_association_state_changes_total`	Contador	`event`	Trans (`comm`
`sctp_send_errors_total`	Contador	`reason`	Envío
`sctp_heartbeat_sent_total`	Contador	`type`	Latid
`sctp_heartbeat_received_total`	Contador	`type`	Latid

SMS / SMSc

Métrica	Tipo	Etiquetas	Descripción
`sms_messages_total`	Contador	`direction`, `operation`	SMS por direcció (`mo`/`mt`) y operación (`submit`/`deliver`
`sms_delivery_failures_total`	Contador	`reason`	Entregas de SMS fallidas por razón
`sms_tracked_subscribers`	Gauge	-	Suscriptores actualmente rastreados por el SubscriberTracke

M2PA (Adaptación Peer-to-Peer de MTP2)

Métrica	Tipo	Etiquetas	Descripción
`m2pa_messages_sent_total`	Contador	`link_name`, `point_code`, `adjacent_pc`	Mensajes MTP3 enviados a través de M2PA
`m2pa_messages_received_total`	Contador	`link_name`, `point_code`, `adjacent_pc`	Mensajes MTP3 recibidos a través de M2PA
`m2pa_bytes_sent_total`	Contador	`link_name`, `point_code`, `adjacent_pc`	Bytes enviados a través de M2PA
`m2pa_bytes_received_total`	Contador	`link_name`, `point_code`, `adjacent_pc`	Bytes recibidos a través de M2PA
`m2pa_link_state_changes_total`	Contador	`link_name`, `from_state`, `to_state`	Transiciones de estado del enlace M2PA
`m2pa_errors_total`	Contador	`link_name`, `error_type`	Errores de M2PA por tipo

M3UA STP

Métrica	Tipo	Etiquetas	Desc
`m3ua_stp_messages_received_total`	Contador	`peer_name`, `point_code`	Mensajes M recibidos p
`m3ua_stp_messages_sent_total`	Contador	`peer_name`, `point_code`	Mensajes M enviados p
`m3ua_stp_routing_failures_total`	Contador	`reason`	Fallos de en STP (`no_route`/

Configuración de Prometheus

Agregue a su `prometheus.yml`:

```yaml
scrape_configs:
  - job_name: 'omniss7'
    static_configs:
      - targets: ['server-ip:9090']
    metrics_path: '/metrics'
    scrape_interval: 15s
```

Ejemplos de Solicitudes

Ejemplo en Python

```python
import requests
import json

# Solicitud SRI-for-SM
url = "http://localhost:9090/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

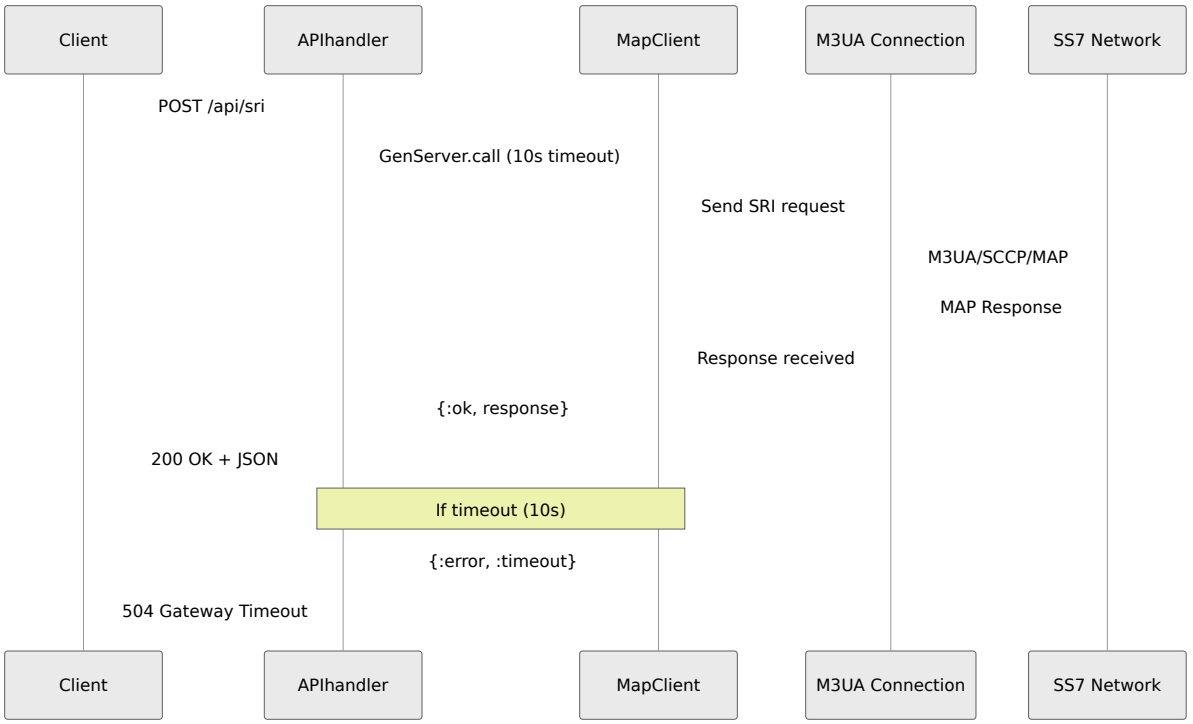
```
const axios = require('axios');

async function sendSRI() {
  try {
    const response = await
axios.post('http://localhost:9090/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:9090/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 Solicitud de API



Resumen

La API REST de OmniSS7 proporciona:

- **47 puntos finales de operación** en Gestión de Movilidad, SMS, Autenticación, Datos del Suscriptor, Servicios Suplementarios, Servicios de Localización, GPRS, Equipos, USSD, CAP/CAMEL, e Inyección en Bruto.
- **Swagger UI** en `/swagger` para documentación interactiva y pruebas.
- **Métricas de Prometheus** en `/metrics` para monitoreo y observabilidad.
- **Tiempo de espera por defecto de 10 segundos** para operaciones esperadas (`504` en caso de tiempo de espera).
- Un servidor HTTP **Plug.Cowboy en el puerto 9090**, controlado por `start_http_server`. Este servidor es separado del panel de control (:8087) y de los controladores REST `:api_ex` (:8445).

Para esquemas de solicitud/respuesta por operación y códigos de operación MAP, consulte la Guía del Cliente MAP.

Para acceso a la UI Web, consulte la Guía de UI 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

Los opcodes a continuación son los códigos de operación `local` de MAP según 3GPP TS 29.002. Esta tabla cubre las operaciones que OmniSS7 genera y/o maneja; para el conjunto completo de operaciones que pueden ser activadas por el cliente y sus parámetros de solicitud, consulte la **Guía del Cliente MAP**.

Operación	Opcode	Propósito
updateLocation	2	Registrar la ubicación del suscriptor (CS)
cancelLocation	3	Dar de baja al suscriptor del VLR/SGSN
provideRoamingNumber	4	Solicitar MSRN
noteSubscriberDataModified	5	Notificar a gsmSCF sobre el cambio de datos del suscriptor
insertSubscriberData	7	Enviar perfil del suscriptor al VLR/SGSN
deleteSubscriberData	8	Retirar datos del suscriptor del VLR/SGSN
registerSS	10	Registrar un servicio suplementario (p. ej. CFU)
eraseSS	11	Borrar un servicio suplementario
activateSS	12	Activar un servicio suplementario
deactivateSS	13	Desactivar un servicio suplementario
interrogateSS	14	Interrogar el estado del servicio suplementario
registerPassword	17	Registrar una contraseña de SS

Operación	Opcode	Propósito
getPassword	18	Solicitar una contraseña de SS al suscriptor
sendRoutingInfo	22	Consultar el enrutamiento de llamadas (SRI)
updateGprsLocation	23	Registrar la ubicación de conmutación de paquetes (SGSN)
sendRoutingInfoForGprs	24	Consultar el enrutamiento GPRS hacia el GGSN
reset	37	Reinicio del HLR hacia el VLR/SGSN
forwardCheckSS-Indication	38	Indicar que los datos de SS deben ser verificados nuevamente
checkIMEI	43	Consultar el EIR sobre el estado del equipo
mt-forwardSM	44	Entregar SMS al suscriptor (MT)
sendRoutingInfoForSM	45	Consultar el enrutamiento de SMS (SRI-for-SM)
mo-forwardSM	46	Reenviar SMS del suscriptor (MO)
reportSM-DeliveryStatus	47	Informar el resultado de la entrega de SMS al HLR

Operación	Opcode	Propósito
sendIdentification	55	Recuperar IMSI/auth del VLR anterior
sendAuthenticationInfo	56	Solicitar vectores de autenticación
restoreData	57	Restaurar datos del suscriptor después de una falla del VLR
sendIMSI	58	Resolver MSISDN a IMSI en el HLR
processUnstructuredSS-Request	59	Solicitud USSD de origen móvil (entrante)
unstructuredSS-Request	60	Solicitud USSD de origen en la red (saliente)
unstructuredSS-Notify	61	Notificación USSD de origen en la red
anyTimeSubscriptionInterrogation	62	Consulta de datos de suscripción por gsmSCF
alertServiceCentre	64	Alertar al SMSc que un suscriptor es alcanzable
anyTimeModification	65	Modificación de datos del suscriptor por gsmSCF
readyForSM	66	Notificar al HLR que el suscriptor está listo para SMS

Operación	Opcode	Propósito
purgeMS	67	Purgar el registro del suscriptor del VLR/SGSN
provideSubscriberInfo	70	Solicitar estado/ubicación del suscriptor
anyTimeInterrogation	71	Consulta de estado/ubicación del suscriptor por gsmSCF
provideSubscriberLocation	83	Solicitar estimación de ubicación del nodo de servicio (LCS)
sendRoutingInfoForLCS	85	Consulta GMLC para el nodo de servicio del suscriptor (LCS)
subscriberLocationReport	86	El nodo de servicio informa una ubicación al GMLC (LCS)
noteMM-Event	89	Informar un evento de gestión de movilidad a gsmSCF

Tipos de Mensajes TCAP

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

Direccionamiento SCCP

Formatos de Título Global

- **E.164** - Número de teléfono internacional (p. ej., 447712345678)
- **E.212** - Formato IMSI (p. ej., 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

SMS TPDU

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 de 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 Com��n 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 del subsistema
- **3** - Fallo del subsistema
- **4** - Usuario no equipado
- **5** - Fallo de MTP
- **6** - Congestión de la red
- **7** - No calificado
- **8** - Error en el transporte del mensaje

Códigos de Error MAP

Código	Error	Descripción
1	unknownSubscriber	Suscriptor no en el 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 SMS
- Guía HLR
- Características Comunes

Guía de Configuración del Gateway CAMEL

Descripción General

El modo **Gateway CAMEL (CAMELGW)** convierte a OmniSS7 en una plataforma de Red Inteligente (IN) que proporciona control de llamadas y facturación en tiempo real utilizando el protocolo CAMEL Application Part (CAP). OmniSS7 actúa como el **gsmSCF** (Función de Control de Servicio), recibiendo disparadores del **gsmSSF** en el MSC/VLR/GMSC y tomando decisiones de facturación a través de CGrateS.

¿Qué es CAMEL?

CAMEL (Aplicaciones Personalizadas para Lógica Mejorada de Redes Móviles) es un conjunto de estándares que funcionan 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 prepago** - Verificación y facturación de saldo en tiempo real
- **Servicios de tarifa premium** - Facturación especial para servicios de valor agregado
- **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

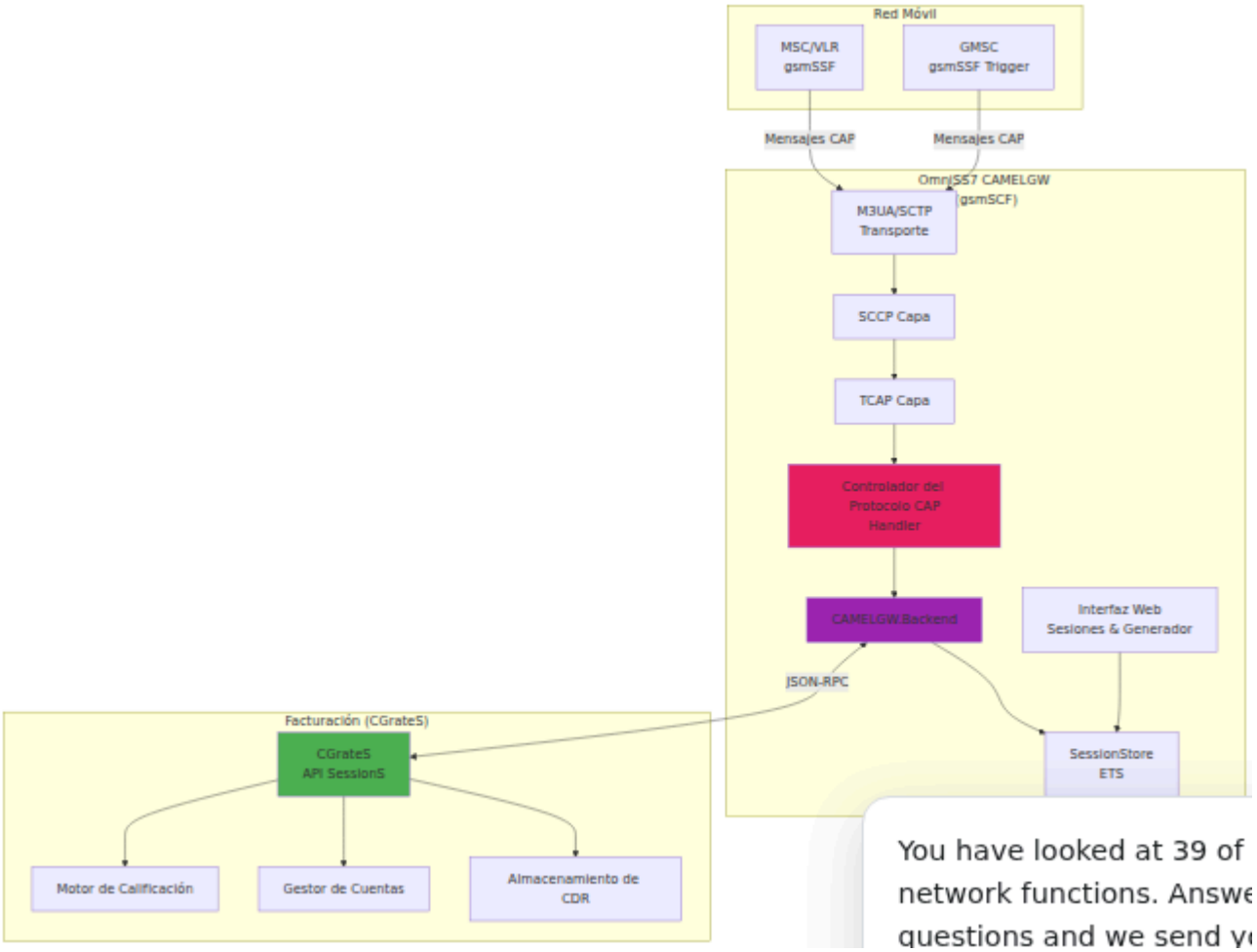
Versiones del Protocolo CAP

OmniSS7 CAMELGW admite múltiples versiones de CAP. La versión controla el OID de contexto de aplicación que se transporta en el diálogo TCAP:

Versión	Fase	OID de Contexto de Aplicación
CAP v1	Fase CAMEL 1	`0.4.0.0.1.0.50.0`
CAP v2	Fase CAMEL 2	`0.4.0.0.1.0.50.1`
CAP v3	Fase CAMEL 3	`0.4.0.0.1.21.3.4`
CAP v4	Fase CAMEL 4	`0.4.0.0.1.23.3.4`

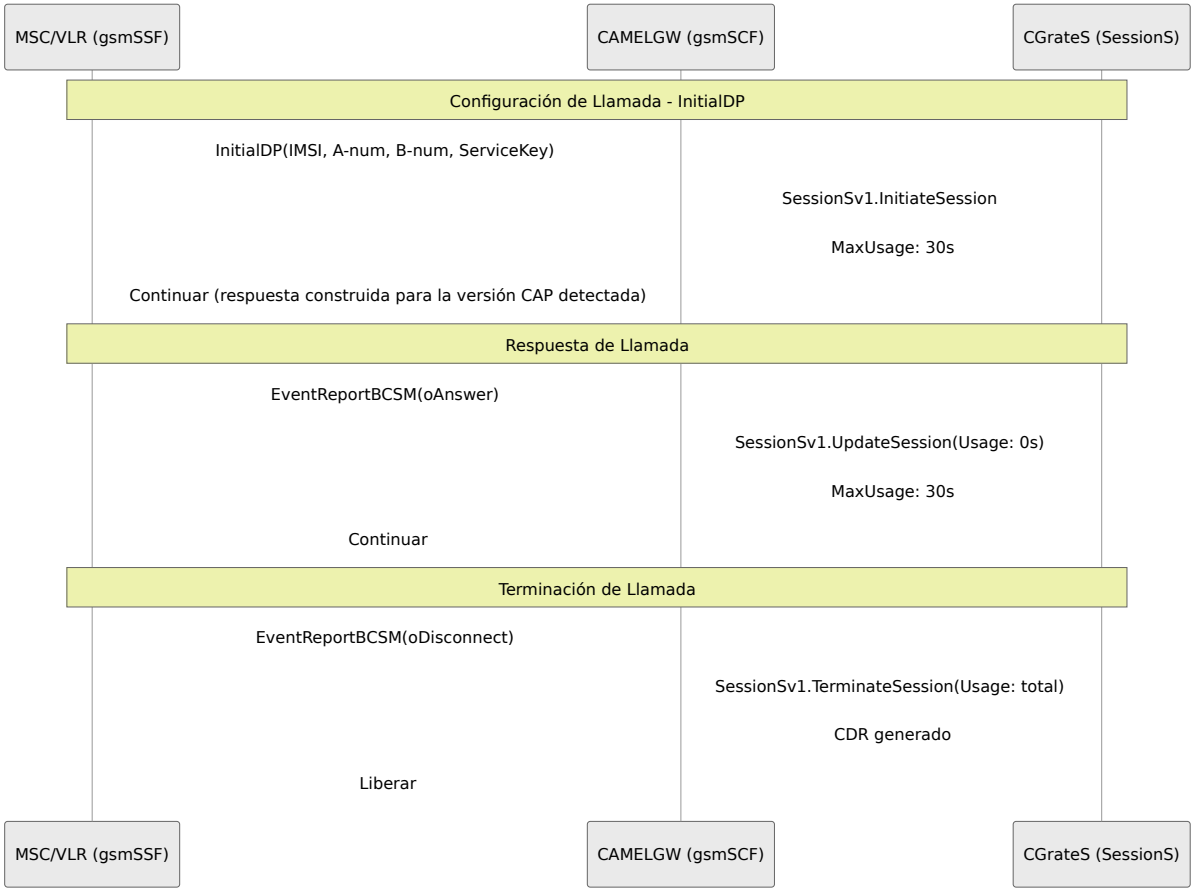
Predeterminado: CAP v2 (el más ampliamente implementado). La versión entrante se detecta automáticamente a partir de la parte del diálogo InitialDP y se reutiliza al generar la respuesta, de modo que las respuestas coincidan con la versión CAP de cada MSC.

Arquitectura



You have looked at 39 of
network functions. Answe
questions and we send y

Ejemplo de Flujo de Llamadas



Configuración

Requisitos Previos

- OmniSS7 instalado y en funcionamiento
- Conectividad M3UA al MSC/GMSC (gsmSSF)
- Una instancia de CGrateS accesible a través de JSON-RPC (opcional, para facturación en tiempo real)

Habilitar el Modo Gateway CAMEL

El procesamiento CAMEL se controla mediante dos banderas independientes. `cap_client_enabled` activa el cliente CAP/M3UA y los componentes de sesión

CAMEL; `cgrates_enabled` activa la integración de facturación CGrateS detrás del backend. Configure estos en `config/runtime.exs`:

```
config :omniss7,
  # Activa el cliente CAP/CAMEL (M3UA ASP, SessionStore, manejo de
solicitudes)
  cap_client_enabled: true,

  # Otros modos independientes (configúrelos según sea necesario
para su implementación)
  map_client_enabled: false,
  smsc_mode_enabled: false,

  # Versión CAP utilizada para solicitudes salientes / diálogo
(:v1 | :v2 | :v3 | :v4)
  cap_version: :v2,

  # Integración de facturación CGrateS
  cgrates_enabled: true,

  # Conexión M3UA para CAP (conectar como ASP hacia el MSC/GMSC
gsmSSF)
  cap_client_m3ua: %{
    routing_context: 1,
    opc: 5013,
    dpc: 5011
  }
```

> **Nota sobre `cap_client_m3ua`:** el cliente CAMEL solo lee `:routing_context`, `:opc` y `:dpc` de este mapa. Cualquier clave adicional es ignorada por el cliente CAP. El nombre del proceso M3UA ASP se fija internamente como `:camelgw_client_asp`.

> No hay bandera `camelgw_mode_enabled`, `hlr_mode_enabled` o `smsc_mode_enabled`-for-CAMEL. Las únicas banderas que controlan el comportamiento CAMEL son `cap_client_enabled` y `cgrates_enabled`.

Banderas de Modo

Clave	Predeterminado	Propósito
`cap_client_enabled`	`false`	Inicia el cliente CAP/CAMEL (M3UA ASP, SessionStore, manejo de solicitudes CAP)
`cgrates_enabled`	`false`	Habilita la integración de facturación CGrateS en `CAMELGW.Backend`
`cap_version`	`:v2`	Versión CAP predeterminada / OID de diálogo para solicitudes salientes
`map_client_enabled`	`false`	Modo de cliente MAP independiente
`smsc_mode_enabled`	`false`	Modo SMSc independiente (requiere cliente MAP)

Parámetros de Facturación CGrateS

Clave	Predeterminado	Pr
`cgrates_enabled`	`false`	Interrup para la CGrate
`cgrates_url`	`http://localhost:2080/jsonrpc`	Endpoi de CGra
`cgrates_tenant`	`cgrates.org`	Inquilin cada ev Sessior
`cgrates_request_type`	`*prepaid`	Tipo de CGrate ejemplo `*postp`
`cgrates_timeout`	`5000`	Tiempo solicitu milisegu
`cgrates_voice_usage_seconds`	*Requerido*	Tamañ fragme segund InitialD ApplyC reserva cantida ApplyC otorga. valor qu interva de TAS

Configurar Páginas de la Interfaz Web

La Interfaz Web expone las páginas CAMEL a través del panel de control. Estas se configuran en `config/config.exs`:

```
config :control_panel,
  use_additional_pages: [
    {SS7.Web.EventsLive, "/events", "Eventos SS7"},
    {SS7.Web.M3UAStatusLive, "/m3ua", "Pares"},
    {SS7.Web.CAMELSessionsLive, "/camel_sessions", "Sesiones
CAMEL"},
    {SS7.Web.CAMELRequestLive, "/camel_request", "Generador de
Solicitudes CAMEL"}
  ],
  page_order: [
    "/events", "/m3ua", "/camel_sessions", "/camel_request",
    "/application", "/configuration"
  ]
```

Etiqueta de Página	Ruta	Módulo
Pares	`/m3ua`	`SS7.Web.M3UAStatusLive`
Sesiones CAMEL	`/camel_sessions`	`SS7.Web.CAMELSessionsLive`
Generador de Solicitudes CAMEL	`/camel_request`	`SS7.Web.CAMELRequestLive`

Operaciones CAP Soportadas

Las operaciones CAP se identifican por nombre y dirección. Tenga en cuenta que algunos números de opcodes CAP son internamente inconsistentes (el mapa de opcodes del registro de eventos y el generador de solicitudes no coinciden en algunas operaciones), por lo que las operaciones a continuación se describen por **nombre y endpoint**. Donde se cita un opcode, es el que el generador de solicitudes realmente emite.

Operaciones Entrantes (gsmSSF → gsmSCF)

Dispatchadas en el opcode del componente TCAP decodificado:

Operación	Notas
InitialDP	Notificación de configuración de llamada; activa CGrateS `InitiateSession`. Aceptado en opcode 0, y también en opcode 60 (tratado como InitialDP/InitialDPSMS)
AssistRequestInstructions	Reconocido; dispatchado en opcode 1 en un TCAP Begin
EventReportBCSM	Eventos BCSM. oAnswer se registra (sin respuesta); oDisconnect / oAbandon / tDisconnect / tAbandon / ocupado / sin respuesta terminan la sesión CGrateS y finalizan el diálogo
ApplyChargingReport	Opcode 36. Llamada activa: informa el período a CGrateS y otorga el siguiente fragmento. Llamada finalizada: termina la sesión CGrateS con el uso total
Continue	Reconocido
ConnectToResource	Reconocido
RequestReportBCSMEvent	Reconocido
ApplyCharging	Parámetros decodificados en BER y reconocidos
Connect	Parámetros decodificados en BER y reconocidos
ReleaseCall	Reconocido

Los opcodes no manejados se responden con un error TCAP (Instalación No Soportada / Valor de Datos Inesperado).

Operaciones Salientes (gsmSCF → gsmSSF)

Enviadas hacia el gsmSSF. Los opcodes a continuación son los valores que OmniSS7 emite:

Operación	Opcode
InitialDP	0
Connect	20
ReleaseCall	22
RequestReportBCSMEvent	23
Continue	31
ApplyCharging	35
InitialDPSMS	60
NoteMM-Event	89 (MAP)

> **InitialDPSMS** (opcode 60) es para control CAMEL de MO-SMS: enviado por el smsSSF al gsmSCF cuando un SMS-CSI / mo-sms-CSI está armado, antes de que el mensaje corto sea retransmitido al SMSC. El argumento está codificado con el módulo CAP ASN.1 generado y el diálogo lleva el contexto de aplicación `id-ac-cap3-sms-AC`.
>
> **NoteMM-Event** es una operación MAP (opcode 89, no CAP) utilizada para notificaciones de movilidad CAMEL M-CSI del VLR al gsmSCF. Está codificado con el módulo MAP ASN.1 (`:TCAPMessages.encode(:MapSpecificPDUs, …)`) y lleva el contexto de aplicación `mm-EventReporting-v3`.

Todos los mensajes TCAP CAP están codificados con `:GenericTCAP.encode(:GenericPDUs, …)`. Solo el NoteMM-Event basado en MAP utiliza `:TCAPMessages`.

Características de la Interfaz Web

Página de Sesiones CAMEL

Título: Sesiones CAMEL · **Ruta:** `/camel_sessions` (HTTPS, puerto 8087)

Monitoreo en tiempo real de sesiones de llamadas CAMEL activas:

- **Lista de sesiones en vivo** con auto-refresco
- **Detalles de la sesión** - OTID, ID de Llamada, Estado, Duración
- **Versión CAP** detectada a partir de InitialDP y almacenada por sesión
- **Información de la llamada** - IMSI, Número A, Número B, Clave de Servicio
- **Seguimiento de estado** - `iniciada`, `respondida`, `terminada`

Columnas de la Tabla: ID de Llamada, Estado, Versión, IMSI, Número Llamante, Número Llamado, Clave de Servicio, Duración, Hora de Inicio, OTID

Los mismos datos de sesión están disponibles a través de la API REST en `/camel-sessions` (ver Monitoreo y Operaciones).

Generador de Solicitudes CAMEL

Título: Generador de Solicitudes CAMEL · **Ruta:** `/camel_request` (HTTPS, puerto 8087)

Herramienta interactiva para construir y enviar solicitudes CAP para pruebas.

Tipos de solicitud ofrecidos por el formulario: InitialDP, Connect, ReleaseCall, RequestReportBCSMEvent, ApplyCharging, Continue.

Selector de versión CAP: el formulario tiene un menú desplegable de versión CAP por solicitud (v1/v2/v3/v4 con sus OIDs). La selección anula temporalmente la predeterminada configurada para esa solicitud y persiste a través de cambios en el formulario; el historial de solicitudes muestra la versión utilizada en una columna **CAP Ver**.

Opciones avanzadas de SCCP/M3UA:

Campo	Predeterminado	Notas
GT de Parte Llamada	`99988415011`	Título Global de Destino
GT de Parte Llamante	(configurable)	Título Global de Origen
SSN Llamada	`146`	gsmSSF
SSN Llamante	`146`	gsmSSF
OPC	`5013`	Código de Punto de Origen
DPC	`5011`	Código de Punto de Destino

Integración con CGrateS

OmniSS7 factura llamadas a través de la API JSON-RPC **SessionS** de CGrateS. `CGrateS.SessionsClient` publica en el `cgrates_url` configurado y mapea las operaciones CAP a métodos de SessionS:

Disparador CAP	Método de SessionS
InitialDP	`SessionSv1.InitiateSession` (reservar un fragmento)
ApplyChargingReport, llamada activa	`SessionSv1.UpdateSession` (`LastUsed` = el período, reservar el siguiente fragmento)
ApplyChargingReport, llamada finalizada; EventReportBCSM desconectar / abandonar; TCAP Fin o Abort desde el MSC	`SessionSv1.TerminateSession` (uso total), luego `SessionSv1.ProcessCDR`

La facturación de voz sigue el modelo TAS Ro: el mismo inquilino (`cgrates_tenant`), `ToR *voice`, `RequestType` `cgrates_request_type`, `Category call`, un fragmento fijo (`cgrates_voice_usage_seconds`), y `GetAttributes` en cada solicitud para que los perfiles de AttributeS del inquilino mapeen al suscriptor en la cuenta de facturación. La `Account` es el **IMSI** del suscriptor servido desde el InitialDP (un InitialDP sin IMSI se libera). `Destination` es el número llamado. `OriginID` es `CAMEL-<hex OTID>`, `OriginHost` es `CAMELGW`.

Ciclo de Vida de la Llamada con Facturación

1. Inicio de Llamada (InitialDP)

Cuando el MSC envía InitialDP, CAMELGW:

1. **Detecta la versión CAP** a partir de la parte del diálogo
2. **Decodifica el InitialDP** (IMSI, número llamado/llamante ISUP, CalledPartyBCDNumber, clave de servicio, tipo de evento)
3. **Llama a CGrateS** `SessionSv1.InitiateSession` para un fragmento
4. **Responde al MSC** en un TCAP Continuar con RequestReportBCSMEvent (O-BCSM o T-BCSM respuesta, desconectar la pierna 1 y 2, abandonar; todos notifyAndContinue), ApplyCharging y Continuar

`MaxUsage` dimensiona la `maxCallPeriodDuration` de ApplyCharging. Se otorga un fragmento completo sin `releaseIfDurationExceeded`; un resto más corto se otorga con él, de modo que el MSC libera la llamada cuando se utiliza el último crédito. Sin crédito, o un error de CGrateS, se responde con **ReleaseCall** (causa 31) en un TCAP Fin.

2. Respuesta de Llamada (EventReportBCSM - oAnswer / tAnswer)

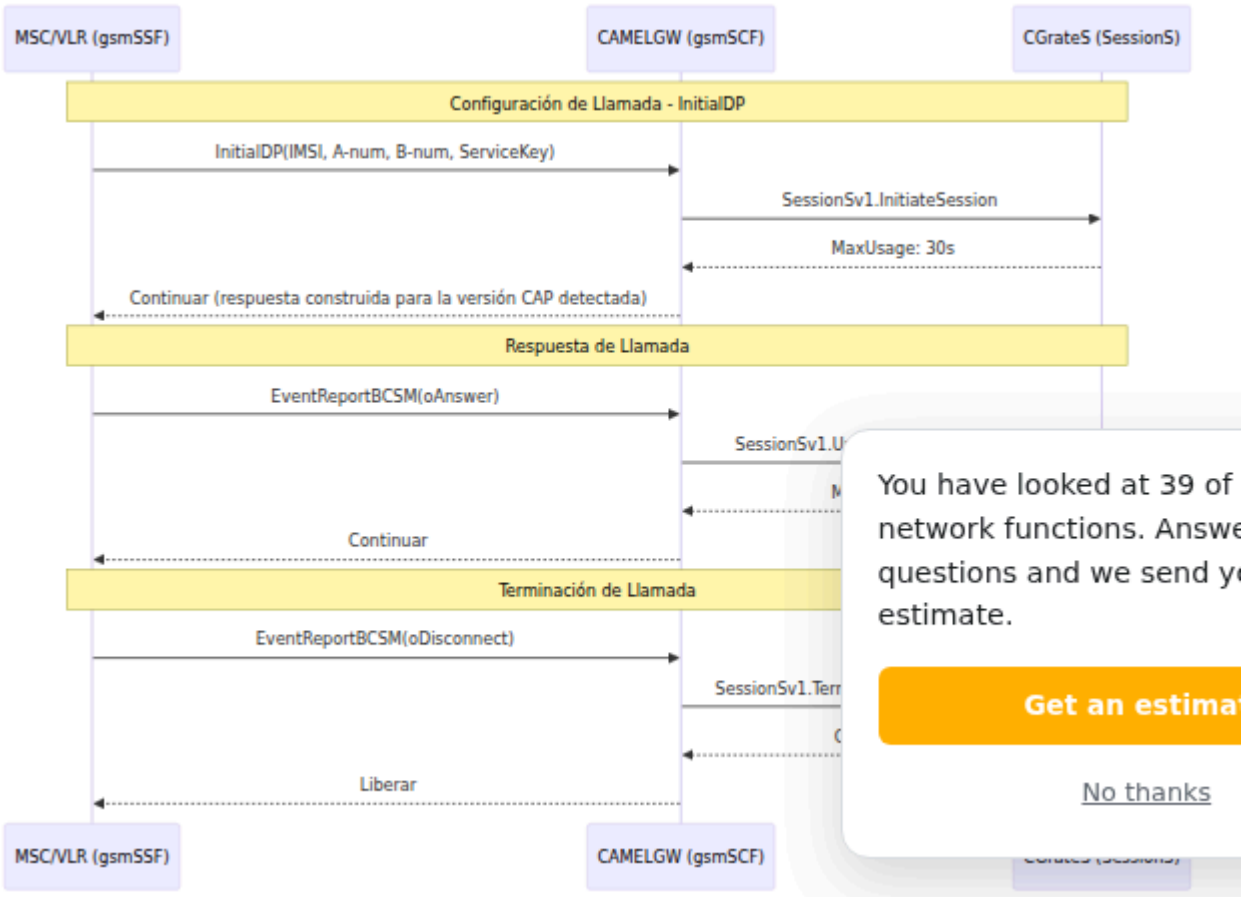
La sesión se marca como respondida. No se envía nada de vuelta; el diálogo permanece abierto.

3. Expiración del Período (ApplyChargingReport)

Cada vez que finaliza el período otorgado, el MSC informa el tiempo utilizado. El gateway lo informa a CGrateS y otorga el siguiente fragmento, o envía ReleaseCall cuando se agota el crédito.

4. Terminación de Llamada

El último ApplyChargingReport (llamada ya no activa), un EventReportBCSM de desconexión / abandono, o el MSC cerrando el diálogo termina la sesión CGrateS con el uso total, escribe el CDR (`ProcessCDR`) y elimina la sesión.



Campos CDR desde CAMEL

Los CDR son producidos por CGrateS e incluyen típicamente:

- `Account` - IMSI (o número de llamada como respaldo)
- `Destination` - Número de la parte llamada
- `OriginID` - Identificador de llamada (`CAMEL-<hex OTID>`)
- `Usage` - Duración total de la llamada (segundos)
- `IMSI`, `CallingPartyNumber`, `CalledPartyNumber`
- `MSCAddress` - MSC que sirve
- `ServiceKey` - Clave de servicio CAMEL

Pruebas

Pruebas Manuales con el Generador de Solicitudes

1. Abra `https://su-servidor:8087/camel_request`
2. Seleccione **InitialDP**, establezca la Clave de Servicio / Números Llamantes / Llamados, opcionalmente elija una versión CAP, y envíe. Tenga en cuenta el OTID generado.
3. Abra `https://su-servidor:8087/camel_sessions` para ver la sesión en estado `iniciada`.
4. Envíe **EventReportBCSM** con `oAnswer` para mover la sesión a `respondida`.
5. Envíe **ReleaseCall** con causa `16` (Normal) para terminar.

Pruebas con un MSC Real

Configure el servicio CAMEL en su MSC/VLR (Clave de Servicio, dirección Global gsmSCF) para que active InitialDP hacia OmniSS7. Ejemplo para un 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

OmniSS7 emite registros JSON estructurados (un objeto por línea a través de `OmniLogger.JsonFormatter`), por lo que puede filtrar en campos de metadatos:

```
# Seguir y filtrar eventos CAP por el campo map_event
tail -f /var/log/omniss7/omniss7.log | jq 'select(.map_event |
tostring | startswith("CAP:"))'
```

Monitoreo y Operaciones

Métricas de Prometheus

CAMELGW expone métricas en `http://localhost:9090/metrics`.

Métricas CAP:

- `cap_requests_total{operation}` - Total de solicitudes CAP por operación. Los valores de etiqueta son snake_case, por ejemplo, `initial_dp`, `connect`, `release_call`, `continue`, `apply_charging`.
- `cap_request_errors_total{operation}` - Errores de generación/envío de solicitudes CAP por operación.

Métricas MAP / API:

- `map_requests_total{operation}` - Total de solicitudes MAP por operación
- `map_pending_requests` - Número de esperas de TID MAP pendientes

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

- `m3ua_stp_routing_failures_total{reason}` - Fallos de enrutamiento por razón

```
# Todos los contadores de solicitudes CAP
curl http://localhost:9090/metrics | grep cap_requests_total

# Solo InitialDP (nota la etiqueta snake_case)
curl http://localhost:9090/metrics | grep
'cap_requests_total{operation="initial_dp"}'
```

Comprobaciones de Salud

```
# Estado del par M3UA (API TLS en el puerto 8445)
curl -k https://localhost:8445/m3ua-status

# Sesiones CAMEL activas (devuelve session_count + detalle por
sesión)
curl -k https://localhost:8445/camel-sessions
```

No hay un endpoint `/api/ocs-status`, `/api/events`, o `/api/camel/sessions/count`. Los conteos de sesiones provienen de `/camel-sessions`.

Configuración de Registros

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

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

El formateador se establece en JSON estructurado en `config/config.exs`:

```
config :logger, :default_formatter,
  format: {OmniLogger.JsonFormatter, :format},
  metadata: :all,
  truncate: :infinity
```

Solución de Problemas

Problema: No se reciben mensajes CAP

Síntomas: El Generador de Solicitudes funciona, pero el MSC nunca envía InitialDP.

Verifique:

1. Estado del par M3UA: `curl -k https://localhost:8445/m3ua-status`
2. Configuración del servicio CAMEL del MSC (Clave de Servicio, dirección gsmSCF)
3. Enrutamiento SCCP (el Título Global gsmSCF debe enrutarse a CAMELGW)
4. Reglas de firewall (permitir el puerto SCTP 2905)

Problema: Errores de CGrateS

Síntomas: `not_configured`, `http_error`, o `cgrates_error` en los registros.

Verifique:

1. `cgrates_enabled: true` y `cgrates_url` apuntan a un endpoint JSON-RPC accesible
2. La cuenta (IMSI o número de llamada) existe y tiene saldo en CGrateS
3. `cgrates_tenant` y `cgrates_request_type` coinciden con su configuración de calificación CGrateS
4. `cgrates_timeout` es lo suficientemente grande para su latencia de CGrateS

En un fallo de facturación durante una llamada, CAMELGW continúa la llamada en lugar de finalizarla; el error se registra.

Problema: Sesión no encontrada

Síntomas: EventReportBCSM registra "Sesión no encontrada".

Causa: Desajuste de OTID o la sesión ya fue terminada.

Verifique: Verifique el OTID en los registros y asegúrese de que el DTID/OTID correlacionen a través del diálogo.

Problema: Errores de decodificación

Síntomas: "Error al decodificar InitialDP" en los registros.

Causa: Desajuste de versión CAP o mensaje mal formado.

Verifique: Confirme que la configuración de la versión CAP coincida con el MSC y capture un PCAP para análisis:

```
tcpdump -i eth0 -w cap_trace.pcap sctp port 2905
```

Resumen

El modo Gateway CAMEL permite que OmniSS7 actúe como un gsmSCF con:

- **Soporte para el protocolo CAP** (v1/v2/v3/v4) con anulación de versión por solicitud
- **Facturación en tiempo real** a través de la API JSON-RPC SessionS de CGrateS
- **Operaciones de control de llamadas** (Connect, Release, Continue, ApplyCharging)
- **Gestión de sesiones** en ETS, indexadas por OTID
- **Pruebas interactivas** a través del Generador de Solicitudes de la Interfaz Web
- **Monitoreo en vivo** de sesiones activas (Interfaz Web y API `/camel-sessions`)
- **Generación de CDR** por CGrateS para facturación y análisis

Para más detalles:

- Referencia Técnica - Operaciones CAP

Producto: OmniSS7 Gateway CAMEL
Versión de Documentación: 1.1

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

1. Descripción General de la Interfaz Web
2. Documentación de la API
3. Monitoreo y Métricas
4. Mejores Prácticas
5. Multihoming SCTP para Redundancia de Red

Descripción General de la Interfaz Web

La Interfaz Web se sirve a través de HTTPS en el puerto `8087` (TLS se aplica a través de `force_ssl`), por ejemplo, `https://localhost:8087/`.

Navegación Principal

El menú se construye a partir de las páginas habilitadas en `config/config.exs` y ordenadas por `page_order`. Entradas comunes (etiqueta del menú en negrita):

- **Recursos** (`/application`) - Estado de la aplicación e información de tiempo de ejecución
- **Configuración** (`/configuration`) - Visor de configuración del sistema
- **Eventos SS7** (`/events`) - Eventos de señalización SS7 en tiempo real y registros de mensajes
- **Registros del Sistema** (`/logs`) - Visor de registros de la aplicación
- **Cliente SS7** (`/client`) - Cliente de prueba interactivo de solicitudes MAP
- **Pares** (`/m3ua`) - Conexiones de pares M3UA/M2PA
- **Enrutamiento** (`/routing`) - Códigos de punto y rutas de Títulos Globales
- **Prueba de Enrutamiento** (`/routing_test`) - Herramienta de prueba de resolución de rutas
- **Enlaces SMSc** (`/smsc_links`) - Enlaces de backend de SMS (SMSc)

- **Suscriptores SMSc** (`/smsc_subscribers`) - Suscriptores de SMS rastreados (SMSc)
- **Enlaces HLR** (`/hlr_links`) - Enlaces de backend de HLR (HLR)
- **Suscriptores Activos** (`/subscribers`) - Suscriptores de UpdateLocation rastreados
- **Sesiones CAMEL** (`/camel_sessions`) - Sesiones de llamada CAMEL en vivo (CAMEL GW)
- **Constructor de Solicitudes CAMEL** (`/camel_request`) - Constructor de solicitudes CAP interactivo (CAMEL GW)
- **Prueba MSISDN/IMSI** (`/msisdn_imsi_test`) - Herramienta de prueba de traducción de números

Accediendo a la Interfaz Web

1. Abre tu navegador web
2. Navega al host configurado en el puerto 8087 (por ejemplo, `https://localhost:8087/`)
3. Visualiza el panel de estado del sistema

Documentación de la API Swagger

La documentación interactiva de la API se sirve a través de la API de operación MAP en el puerto HTTP 9090:

```
http://your-server:9090/swagger
```

Configuración de la Interfaz Web

El punto final de la interfaz web se configura bajo `config :control_panel, ControlPanelWeb.Endpoint` en `config/config.exs`, y el menú de navegación bajo `config :control_panel`:

```
config :control_panel, ControlPanelWeb.Endpoint,
  force_ssl: [rewrite_on: [:x_forwarded_proto]],
  server: true,
  url: [host: "0.0.0.0", path: "/"],
  https: [
    port: 8087,
    keyfile: "priv/cert/omnitouch.pem",
    certfile: "priv/cert/omnitouch.crt"
  ]

config :control_panel,
  page_order: [
    "/application", "/configuration", "/events", "/logs",
"/client",
    "/m3ua", "/routing", "/routing_test", "/smsc_links",
"/smsc_subscribers",
    "/hlr_links", "/subscribers", "/camel_sessions",
"/camel_request",
    "/msisdn_imsi_test"
  ]
```

Parámetros de Configuración:

Parámetro	Tipo	Predetermi
`ControlPanelWeb.Endpoint.url.host`	String	`"0.0.0.0"`
`ControlPanelWeb.Endpoint.https.port`	Integer	`8087`
`ControlPanelWeb.Endpoint.https.keyfile`	String	`"priv/cert/omnit`
`ControlPanelWeb.Endpoint.https.certfile`	String	`"priv/cert/omnit`
`ControlPanelWeb.Endpoint.force_ssl`	Keyword	`[rewrite_on: [:x_forwarded_pr`
`page_order`	Lista de Strings	ver ejemplo arriba

> Consulta la Referencia de Configuración para el esquema completo del Panel de Control.

Configuración del Registrador

Los registros se emiten como JSON estructurado (un objeto por línea) por `OmniLogger.JsonFormatter`, que incluye toda la `Logger.metadata` para que el contexto por proceso (OPC/DPC/GTs de enrutamiento STP, puerta de enlace SMS desde/hacia MSISDNs, otid/operación de transacción) se lleve automáticamente. Este formato se configura en `config/config.exs`:

```
config :logger, :default_formatter,
  format: {OmniLogger.JsonFormatter, :format},
  metadata: :all,
  truncate: :infinity
```

El nivel de registro se establece por separado:

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

Parámetros de Configuración:

Parámetro	Tipo	Predetermin
`:logger.level`	Atom	`:debug`
`:logger.default_formatter.format`	Tuple	`{OmniLogger.JsonF` `:format}`
`:logger.default_formatter.metadata`	Atom/List	`:all`
`:logger.default_formatter.truncate`	Integer/Atom	`:infinity`

Documentación de la API

OmniSS7 expone dos superficies HTTP:

Superficie	Puerto	Protocolo	Contenidos
API de operación MAP	`9090`	HTTP	Operaciones MAP `/api/*`, `/swagger`, `/swagger.json`, `/metrics`
API de estado JSON (`api_ex`)	`8445`	HTTPS	Estado de solo lectura (`/api/status`, `/api/m3ua-status`, `/api/routing`, `/api/subscribers`, `/api/camel-sessions`, ...)

URL Base de la API MAP

```
http://your-server:9090/api
```

Códigos de Respuesta

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

Especificación OpenAPI

```
http://your-server:9090/swagger.json
```

Monitoreo y Métricas

Punto Final de Métricas de Prometheus

```
http://your-server:9090/metrics
```

Categorías Clave de Métricas

Métricas M3UA/SCTP:

- Cambios de estado de 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 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 API o OAuth
    - Registrar todas las solicitudes para auditoría

Ajuste de Rendimiento

1. **Límites de TPS**
    - Configurar TPS apropiado
    - Monitorear carga del sistema
    - Ajustar buffers SCTP
2. **Ajuste del Backend de SMS**

- Mantener la API del backend de SMS receptiva (las solicitudes expiran después de 5 segundos)
- Ajustar `auto_flush_tps` y `auto_flush_interval` para que coincidan con la capacidad del backend
- Monitorear la profundidad de la cola y las tasas de entrega

3. **Ajuste de M3UA**

- Ajustar los 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 sola 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
- **Mejor Confiabilidad**: Sobrevive a fallos de red, fallos de conmutadores o fallos de NIC
- **Balanceo de Carga**: SCTP puede distribuir tráfico a través de múltiples caminos (dependiente de la implementación)

Cómo Funciona

Cuando configuras 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 a través de 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 nuevamente como disponible

Configuración

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

Una Sola IP (Tradicional)

```
# Una sola 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), debes especificar `remote_ip` como una **única tupla** - la dirección IP que el par remoto usa para iniciar la conexión SCTP (SCTP INIT). NO uses una lista.
>
> **¿Por qué?** El STP coincide las conexiones entrantes basándose en 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 - 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 Entradas de Par Multihomed

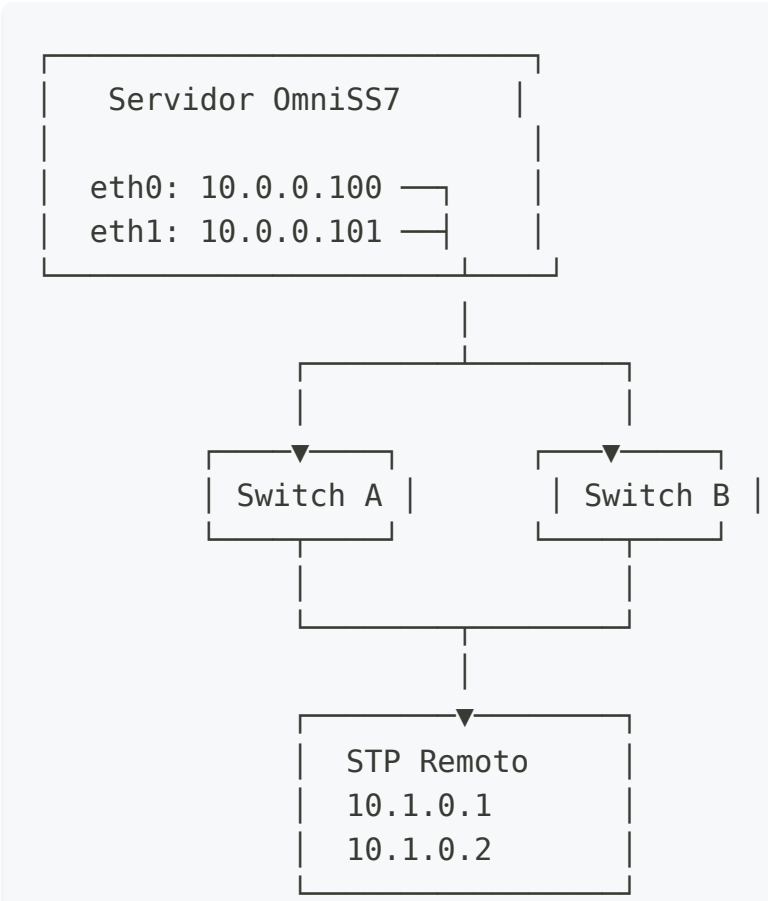
```
# Aceptando conexión entrante de un HLR multihomed
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 1,
      name: "HLR",
      role: :server,  # ENTRANTE - HLR se conecta a nosotros
      # Multihoming: nuestras IPs locales (la lista está bien)
      local_ip: [{10, 0, 1, 10}, {10, 0, 1, 11}],
      local_port: 2905,
      # IMPORTANTE: Solo tupla única - la IP que el HLR usa 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 desde 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 IP
      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: Doble NIC (Despliegue Común)



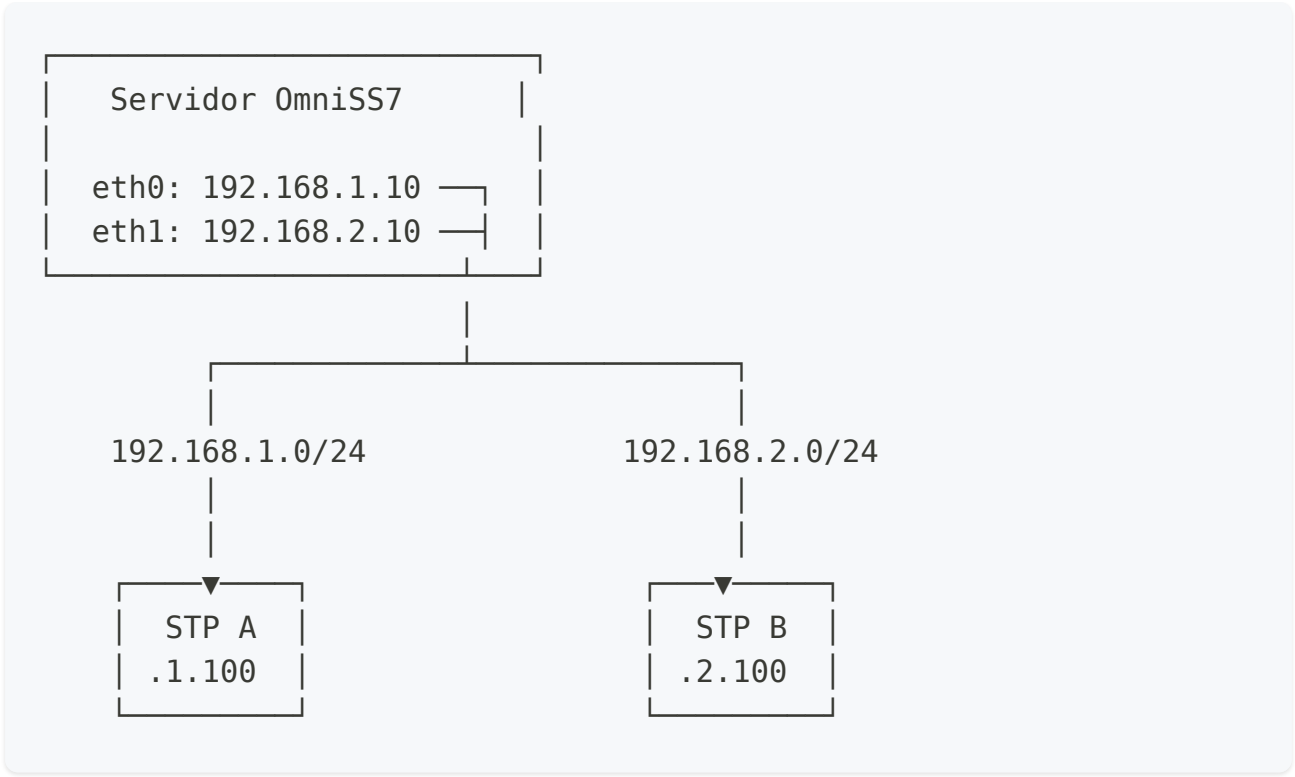
Configuración:

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

Beneficios:

- Sobrevive a la falla de una 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 una subred
- Posibilidad de redundancia geográfica
- Rutas de enrutamiento independientes

Monitoreo y Registro

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

Multihoming Exitoso

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

Eventos de Conmutación por Error 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 fallas de conmutadores
- **Diferentes subredes** si es posible para diversidad de enrutamiento
- **Mismo centro de datos inicialmente** - prueba antes de separación geográfica

2. Planificación de Direcciones IP

- **La primera IP es primaria** - asegúrate de que esté en el camino más confiable
- **El orden importa** - lista las IPs en orden de preferencia
- **Direcciones consistentes** - usa esquemas de direccionamiento similares para solucionar 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, no hay conmutación por error

Verificaciones:

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

Problema: El Camino No Cambia

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

Verificaciones:

1. Verifica la configuración de latidos SCTP
2. Verifica que la tabla de enrutamiento tenga rutas para todos los caminos
3. Verifica que el firewall permita SCTP en todas las IPs
4. Revisa 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 - verifica 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 para 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 abajo > 1 minuto
- Tasa de tiempo de espera de MAP > 10%
- Profundidad de cola > 1000
- Tasa de error de 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, :default_formatter,
  format: {OmniLogger.JsonFormatter, :format},
  metadata: :all,
  truncate: :infinity

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

Parámetro	Tipo	Requerido	
`:logger.level`	Atom	No	`:debu`
`:logger.default_formatter.format`	Tuple	No	`{Omn` `:form`
`:logger.default_formatter.metadata`	Atom/List	No	`:all`
`:logger.default_formatter.truncate`	Integer/Atom	No	`:inf`

Configuración de la Interfaz Web (`:control_panel`)

```
config :control_panel, ControlPanelWeb.Endpoint,
  force_ssl: [rewrite_on: [:x_forwarded_proto]],
  server: true,
  url: [host: "0.0.0.0", path: "/"],
  https: [
    port: 8087,
    keyfile: "priv/cert/omnitouch.pem",
    certfile: "priv/cert/omnitouch.crt"
  ]

config :control_panel,
  page_order: [
    "/application", "/configuration", "/events", "/logs",
"/client",
    "/m3ua", "/routing", "/routing_test", "/smsc_links",
"/smsc_subscribers",
    "/hlr_links", "/subscribers", "/camel_sessions",
"/camel_request",
    "/msisdn_imsi_test"
  ]
```

Parámetro	Tipo	Requerido	
`ControlPanelWeb.Endpoint.url.host`	String	No	`"0.`
`ControlPanelWeb.Endpoint.https.port`	Integer	No	`808`
`ControlPanelWeb.Endpoint.https.keyfile`	String	No	`"pr`
`ControlPanelWeb.Endpoint.https.certfile`	String	No	`"pr`
`ControlPanelWeb.Endpoint.force_ssl`	Keyword	No	`[re` `[:x_`
`page_order`	Lista de Strings	No	ver

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	D
`sctp_handler.enabled`	Boolean	Sí	`false`	H m ir
`sctp_handler.local_ip`	Tuple	Sí	`{127, 0, 0, 1}`	IP vi M e
`sctp_handler.local_port`	Integer	Sí	`2905`	Pu p
`enable_gt_routing`	Boolean	No	`false`	H e d G

Parámetros de Par M3UA:

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

Parámetros de Ruta M3UA:

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

Parámetros de Ruta GT M3UA:

Parámetro	Tipo	Requerido	Descripción
`gt_prefix`	String	Sí	Prefijo de Título Global a coincidir
`peer_id`	Integer	Sí	Par de destino
`priority`	Integer	Sí	Prioridad de la ruta
`description`	String	No	Descripción de la ruta para registro
`source_ssn`	Integer	No	Coincidir solo si el SSN de origen coincide
`dest_ssn`	Integer	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	Predetermin
`map_client_enabled`	Boolean	Sí	`false`
`map_client_m3ua.mode`	String	Sí	`"ASP"`
`map_client_m3ua.callback`	Tuple	Sí	`{MapClient, :handle_payl []}`
`map_client_m3ua.process_name`	Atom	Sí	`:map_client_`
`map_client_m3ua.local_ip`	Tuple	Sí	-
`map_client_m3ua.local_port`	Integer	Sí	`2905`
`map_client_m3ua.remote_ip`	Tuple	Sí	-
`map_client_m3ua.remote_port`	Integer	Sí	`2905`
`map_client_m3ua.routing_context`	Integer	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	De
`auto_flush_enabled`	Boolean	No	`false`	Hab vaci auto la co
`auto_flush_interval`	Integer	No	`10000`	Inte sond cola (mil
`auto_flush_dest_smsc`	String/nil	No	`nil`	Filtr SMS dest todo
`auto_flush_tps`	Integer	No	`10`	Máx tran por

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://omnimessage.example.com:8443",
  smsc_name: "omni-smsc01"  # Opcional: predeterminado al nombre
derivado del hostname
```

Parámetros de la API:

Parámetro	Tipo	Requerido	Predeterminado
`smsc_api_base_url`	String	Sí	`"https://omnimessage.example`
`smsc_name`	String	No	nombre derivado del hostname

Puntos Finales de la 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` - Punto final de verificación de salud

Notas de Configuración:

- La verificación SSL está deshabilitada por defecto para certificados autofirmados
- Las solicitudes HTTP expiran después de 5 segundos
- Todas las marcas de tiempo están en formato ISO 8601
- La API utiliza JSON para 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

1. Descripción General
2. Dónde Vive la Configuración
3. Banderas de Modo Operativo
4. Parámetros de Modo HLR
5. Parámetros de Modo SMSc
6. Parámetros de Modo STP
7. Parámetros de Modo CAMEL Gateway
8. Parámetros de NAT de Título Global
9. Parámetros de Conexión M3UA
10. Estado del Subsistema SCCP
11. Filtrado de Gateway
12. Parámetros de Gateway USSD
13. Banderas de Tiempo de Ejecución / Supervisión
14. Parámetros de Infraestructura
15. Parámetros de Base de Datos
16. Valores Codificados

Descripción General

OmniSS7 soporta cuatro modos operativos, seleccionados completamente a través de la configuración:

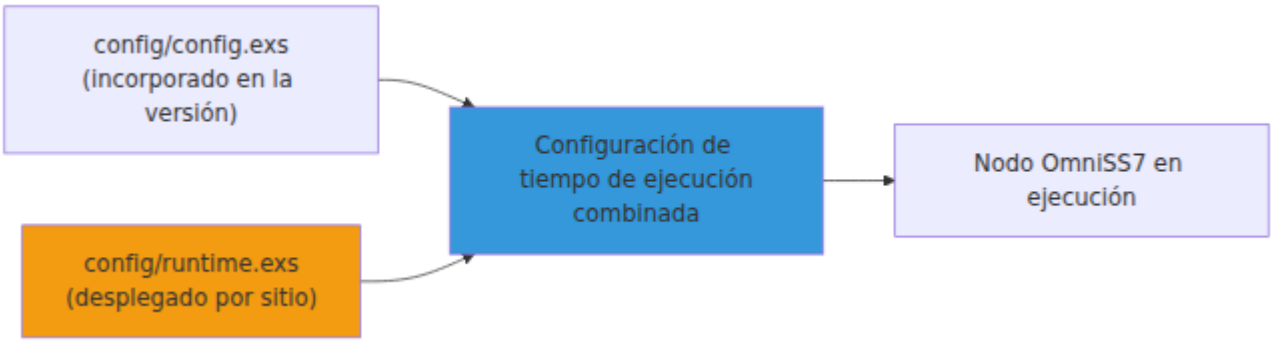
- **Modo STP** - Punto de Transferencia de Señales para enrutamiento
- **Modo HLR** - Registro de Ubicación del Hogar para gestión de suscriptores
- **Modo SMSc** - Centro de SMS para entrega de mensajes
- **Modo CAMEL GW** - Gateway CAMEL para control inteligente de llamadas

Un solo nodo puede combinar roles (por ejemplo HLR + SMSc) habilitando las Bandera de Modo Operativo relevantes.

Dónde Vive la Configuración

OmniSS7 lee su configuración de dos capas. Entender qué archivo posee un valor determina dónde cambiarlo en un sistema desplegado.

Archivo	Rol	¿Editar en despliegue?
`config/config.exs`	**Configuración por defecto / en tiempo de compilación** enviada con la versión. Define las páginas del panel de control, los controladores REST `:api_ex`, el formateador de registro y todos los valores predeterminados básicos.	No: parte de la construcción.
`config/runtime.exs` (en el repositorio)	Un **stub de prueba de 4 líneas** solamente. En el árbol de origen no configura nada más allá del entorno de prueba.	No.
`config/runtime.exs` (desplegado)	La **configuración de tiempo de ejecución por despliegue**, desplegada por separado en cada servidor (típicamente renderizada desde una plantilla de Ansible/Jinja). Aquí es donde se configuran las banderas de modo, GTs, códigos de punto, pares M3UA y backends de API para un sitio dado.	**Sí. Este es el archivo orientado al operador.**



> **En resumen:** los ejemplos en este documento utilizan `config :omniss7, ...`. En un nodo en vivo, estos van en el **desplegado**

> `config/runtime.exs`. El `config/runtime.exs` comprometido en el repositorio es intencionalmente un stub utilizado solo por la suite de pruebas automatizadas. No lo confunda con la configuración de producción.

Todas las claves a continuación se leen a través de `Application.get_env(:omniss7, ...)` (o la aplicación relevante) en tiempo de ejecución, por lo que los valores predeterminados mostrados son los valores utilizados cuando se omite una clave.

Banderas de Modo Operativo

Controla qué características están habilitadas.

Parámetro	Tipo	Predeterminado	Descripción	I
`map_client_enabled`	Booleano	`false`	Habilitar conectividad de cliente MAP y 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 del Gateway CAMEL	C C
`ussd_gateway_enabled`	Booleano	`false`	Habilitar Gateway USSD (puente HTTP/JSON)	U C

Ejemplo:

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

Parámetros de Modo HLR

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

Configuración de API HLR

Parámetro	Tipo	Predeterminado	Requeri
`hlr_api_base_url`	Cadena	-	**Sí**
`hlr_api_key`	Cadena	-	**Sí**
`hlr_api_status_path`	Cadena	`"/api/status"`	No

Parámetro	Tipo	Predeterminado	Requeri
`hlr_service_center_gt_address`	Cadena	-	**Sí**
`smsc_service_center_gt_address`	Cadena	-	**Sí**

> **Verificación SSL:** No hay clave `hlr_api_verify_ssl`. No se lee en ningún lugar del código. La verificación del certificado TLS para llamadas a la API de backend está gobernada por el campo `verify_ssl` del mapa genérico `:api` (ver Verificación SSL de Backend de API).

Ejemplo:

```
config :omniss7,
  hlr_api_base_url: "https://omnihss.example.com:8443",
  hlr_api_key: "<Clave de API de OmniHSS con el rol
hlr_interworking>",
  hlr_api_status_path: "/api/status",
  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 MSISDNs. Para una explicación técnica detallada del algoritmo de mapeo, ver 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 dígitos):

```
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: Establecer `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 VLRs durante UpdateLocation. Para una explicación detallada de la secuencia ISD y el flujo de mensajes, ver 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 inteligente de llamadas basado en CAMEL. Para una explicación detallada de la integración CAMEL y las claves de servicio, ver 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

| `hlr_camel_subscriptions` | Lista | `[]` | No | O-CSI / T-CSI por prefijo IMSI enviados al VLR en InsertSubscriberData (ver Suscripciones CAMEL en ISD) |

Ejemplo:

```
config :omniss7,
  camel_service_key: 11_110,
  camel_trigger_detection_point: :termAttemptAuthorized,
  hlr_camel_subscriptions: [
    %{
      imsi_prefix: "3133800000009",
      o_csi: %{service_key: 100, gsmscf_address: "14155550300",
default_call_handling: :release_call},
      t_csi: %{service_key: 101, gsmscf_address: "14155550300",
default_call_handling: :continue_call}
    }
  ]
```

Prefijos VLR de Origen

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

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

Ejemplo:

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

Parámetros de Modo SMSc

Configuración para el modo Centro de SMS.

Configuración de API SMSc

Parámetro	Tipo	Predeterminad
`smsc_api_base_url`	Cadena	`"https://omnimessage.examp`
`smsc_api_status_path`	Cadena	`"/api/status"`
`smsc_name`	Cadena	`"{hostname}_SMSc"`
`smsc_service_center_gt_address`	Cadena	-

> **Verificación SSL:** No hay clave `smsc_api_verify_ssl`. No se lee en ningún lugar del código. La verificación del certificado TLS para llamadas a la API de backend está gobernada por el campo `verify_ssl` del mapa genérico `:api` (ver Verificación SSL de Backend de API).

Ejemplo:

```
config :omniss7,
  smsc_api_base_url: "https://omnimessage.example.com:8443",
  smsc_api_status_path: "/api/status",
  smsc_name: "ipsmgw",
  smsc_service_center_gt_address: "55512341112"
```

Nota: El registro en frontend ocurre cada **5 minutos** (codificado) 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**	Nom 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 de Modo STP

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

Servidor STP Independiente

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, ver 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 los protocolos M3UA como M2PA). Para ejemplos completos de configuración, ver 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 enla
`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 Socket:

M2PA utiliza automáticamente SCTP.SocketHandler para la gestión de socket compartido. Todos los pares M2PA utilizan el socket compartido, lo que permite que múltiples pares compartan eficientemente el mismo puerto SCTP. Para una configuración detallada, ver 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 código de punto. Si no se especifica, las rutas se generan automáticament 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)
- `route_on_ssn`: Invertir el indicador de enrutamiento de la Parte Llamada para enrutamiento por SSN (Booleano, predeterminado `false`). Dejar `false` para enrutamiento GT de tránsito para que el nodo descendente realice GTT (ITU-T Q.714 §2.4); establecer `true` **solo** cuando este STP sea el traductor final que entrega a un punto final solo SSN. Ver Guía STP → Enrutamiento por SSN Después de GTT.

Valores Comunes:

- **Tipo de Traducción (TT):** 0=Desconocido, 1=Internacional, 2=Nacional, 3=Específico de Red
- **Plan de Numeración (NPI):** 0=Desconocido, 1=ISDN(E.164), 6=Móvil(E.212)
- **Naturaleza de la 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 NPI
    %{
      gt_prefix: "49",
      peer_id: 1,
      priority: 1,
      description: "Números alemanes: conversión NPI Móvil→ISDN",
      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 por defecto/alternativa
    %{
      gt_prefix: "",
      peer_id: 1,
      priority: 99,
      description: "Ruta de respaldo por defecto"
    }
  ]
```

Parámetros del Modo de Puerta de Enlace CAMEL

Configuración para el modo de puerta de enlace CAMEL (protocolo CAP).

Banderas de Modo CAMEL

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

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: "99988411553"
```

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 JSO 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 CAP M3UA

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 CAP M3UA (misma estructura qu `map_client_m` 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 GTs 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 de GT 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 llamado (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, gana la longitud de prefijo combinado más largo
3. Tanto `calling_prefix` como `called_prefix` pueden usarse juntos para una coincidencia precisa

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:
"99988411553"},

    # Prioridad media: Coincidir solo llamado a "555"
    %{called_prefix: "555", weight: 10, response_gt:
"99988411554"},

    # Coincidir solo llamando desde "8773"
    %{calling_prefix: "8773", weight: 10, response_gt:
"99988411554"},

    # Regla de respaldo comodín (coincide con todo, mayor peso)
    %{weight: 100, response_gt: "99988411555"}
  ]
```

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

Parámetros de Conexión M3UA

Configuración de conexión M3UA para el modo cliente MAP. Para un uso detallado y ejemplos, 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 del 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.

Watchdog de Recepción

El **watchdog de recepción** monitorea las conexiones SCTP en busca de sockets zombis. Un socket zombi es una asociación que permanece en un estado ESTABLECIDO a nivel del sistema operativo, pero donde el extremo remoto ha dejado de enviar datos silenciosamente. Sin el watchdog, el nodo puede no detectar una conexión muerta 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. Cuando el SG remoto no envía tráfico de capa de aplicación periódicamente (y no envía M3UA BEATs), deshabilite el watchdog para evitar 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 operando 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: deshabilitar 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: personalizar 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)
  }
```

Estado del Subsistema SCCP

OmniSS7 responde a las Pruebas de Estado de Subsistema (SST) de gestión SCCP desde el estado real del subsistema local consultado (ITU-T Q.714 §5.3.4). La clave `local_subsystems` declara qué Números de Subsistema (SSNs) alberga realmente este nodo. Un SST para un SSN listado se responde como `Subsystem-Allowed` (SSA); cualquier otro SSN se responde como `Subsystem-Prohibited` (SSP).

Parámetro	Tipo	Predeterminado	Requerido	Descrip
`local_subsystems`	Lista de Enteros o `:all`	`:all`	No	SSNs SCC locales en servicio utilizados responder `:all` (el predeterm trata cada subsistem como disponible se adapta relé puro alberga subsistem Los opera que termi SSNs espe deben list para que refleje la realidad.

SSNs Comunes: 6 = HLR, 7 = VLR, 8 = MSC, 9 = EIR, 146 = gsmSCF/CAMEL.

Ejemplo:

```
config :omniss7,
  local_subsystems: [6, 7, 8, 146]
```

Filtrado de Puerta de Enlace

El filtrado de puerta de enlace (el firewall SCCP) permite o deniega el tráfico de tránsito antes de que sea enrutado. Las reglas coinciden en el par remoto, OPC, DPC, Indicador de Servicio, prefijo GTA de la parte llamada/llamante SCCP y SSN; la primera regla habilitada que coincida (menor prioridad) decide `allow`/`deny`. Cuando ninguna regla coincide, se aplica la acción predeterminada a continuación. Los mensajes denegados se eliminan y se cuentan como `m3ua_stp_routing_failures_total{reason="screened"}`. Consulte la Guía STP para el modelo completo.

Parámetro	Tipo	Predeterminado	Requerido	
`screening_default_action`	`:allow` o `:deny`	`:allow`	No	A u t r f ` : ` (e c p ` : ` c p (p E h f a r c

Ejemplo:

```
config :omniss7,
  # Denegación predeterminada: solo el tráfico que coincide con
una regla de permiso explícita se reenvía.
  screening_default_action: :deny
```

Las reglas en sí se mantienen en la tabla Mnesia `ss7_screen_rule` y se gestionan programáticamente; solo la acción predeterminada se establece a través de la configuración.

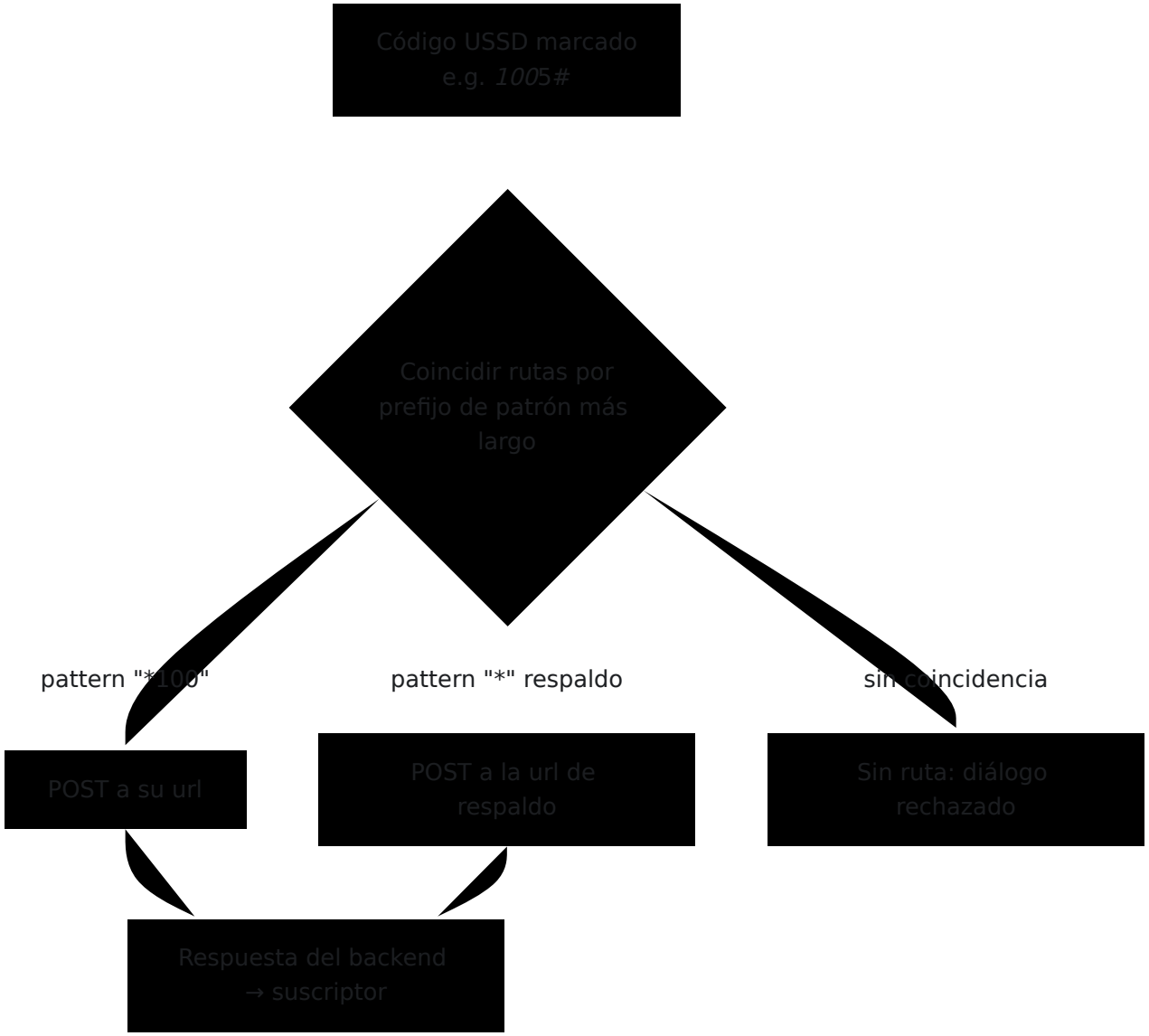
Parámetros de Puerta de Enlace USSD

La Puerta de Enlace USSD conecta USSD iniciados por la red y por el móvil a un backend HTTP/JSON. Habilítelo con `ussd_gateway_enabled: true` (ver Banderas de Modo Operacional), luego configure el mapa `ussd_gateway`. Para el protocolo completo de callback y el ciclo de vida de la sesión, consulte la Guía de Puerta de Enlace USSD.

Parámetro	Tipo	Predeterminado	Requeri
`ussd_gateway.routes`	Lista de Mapas	`[]`	**Sí** (si est habilitad
`ussd_gateway.session_timeout_ms`	Entero	`180_000`	No
`ussd_gateway.turn_timeout_ms`	Entero	`30_000`	No
`ussd_gateway.http_timeout_ms`	Entero	`5_000`	No

Parámetro	Tipo	Predeterminado	Requeri
`ussd_gateway.max_text_length`	Entero	`182`	No

Enrutamiento: Cuando llega un código USSD, la puerta de enlace ordena las rutas configuradas por longitud de `pattern` en orden descendente. Luego, la puerta de enlace selecciona la primera ruta cuyo `pattern` es un prefijo del código marcado. Esto proporciona una coincidencia determinística de prefijo más largo.



Ejemplo:

```
config :omniss7,
  ussd_gateway_enabled: true,
  ussd_gateway: %{
    routes: [
      %{pattern: "*100", url: "http://balance-app:9000/ussd"},
      %{pattern: "*200", url: "http://topup-app:9000/ussd"},
      %{pattern: "*",    url: "http://default-app:9000/ussd"}
    ],
    session_timeout_ms: 180_000,
    turn_timeout_ms: 30_000,
    http_timeout_ms: 5_000,
    max_text_length: 182
  }
```

Banderas de Ejecución / Supervisión

Estas banderas controlan cuánto del árbol de supervisión se inicia. Son útiles para ejecutar OmniSS7 sin cabeza (como una dependencia de biblioteca) o para eliminar la capa web en implementaciones restringidas.

Parámetro	Tipo	Predeterminado	Requerido	

`start_web_interfaces`	Booleano	`true`	No	Ini
				us
				co
				`:a`
				reg
				SS
				so
				reg
				(`S`
				se
				po
				en
				`EN`
				(`t`

`start_http_server`	Booleano	`true`	No	Ini
				Plu
				pu
				`/a`
				`/s`
				AP

`api_port`	Entero	`9090`	No	Pu
				se
				Plu
				`/a`
				`/s`
				ob
				Pro
				ca

`headless`	Booleano	`false`	No	Su de `tr` **so**

Parámetro	Tipo	Predeterminado	Requerido	
				có (re US dir we lic SC M3 inc On de
`vlr_cache_ttl_seconds`	Entero	`3600`	No	Tie (se bú ca en

Ejemplo (modo de biblioteca sin cabeza):

```
config :omniss7,
  headless: true
```

Ejemplo (sin capa web, solo API REST):

```
config :omniss7,
  start_web_interfaces: false,
  start_http_server: true
```

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"
  }
```

Puertos y transportes de un vistazo:

Puerto	Transporte	Servidor	Propósit
9090	HTTP (Plug.Cowboy)	`APIhandler`	MAP/CAP REST `/a` `/metrics`, `/swag` `/swagger.json`. A por `start_http_`
8087	HTTPS (Bandit/Phoenix)	`ControlPanelWeb.Endpoint`	Interfaz de usuar panel de control Parte de la capa (`start_web_inte`
8445	HTTPS / TLS	`:api_ex` controladores REST	Estado, CRUD de enrutamiento, en suscriptores, vist CAMEL/SCTP/M3U de la capa web.

Puntos finales de la API:

- API REST MAP/CAP + Swagger: `http://[server-ip]:9090/...` (ver Guía de API)
- Controladores REST `:api_ex`: `https://[server-ip]:8445/api/*`
- Panel de control web: `https://[server-ip]:8087/`
- Métricas de Prometheus: `http://[server-ip]:9090/metrics`

Verificación SSL del Backend de la API

Las llamadas salientes a las API HLR/SMSc del backend **no** utilizan claves `*_verify_ssl` por objetivo. La verificación de certificados está controlada por el mapa genérico `:api`:

Parámetro	Tipo	Predeterminado	Descripción
`api.base_url`	Cadena	retrocede a `smsc_api_base_url`	URL base para el cliente API genérico.
`api.timeout`	Entero	`5000`	Tiempo de espera de la solicitud en milisegundos.
`api.verify_ssl`	Booleano	`false`	Si verificar los certificados TLS del backend. **Este es el único interruptor de verificación SSL.** No hay `hlr_api_verify_ssl` o `smsc_api_verify_ssl`.

Ejemplo:

```
config :omniss7,
  api: %{
    base_url: "https://backend.example.com:8443",
    timeout: 5000,
    verify_ssl: false
  }
```

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 almace 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 en disco persistente (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 Codificados

Los siguientes valores están **codificados en el código fuente** y no se pueden cambiar 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: **9090**	El punto final de métricas/Swagger se ejecuta en el puerto 9090	Establecer `config :omniss7, api_port: <port>` (predeterminado `9090`)

Intervalos de Registro

Valor	Impacto	Solución Alternativa
Registro en el 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://omnihss.example.com:8443",
  hlr_api_key: "<Clave API de OmniHSS con el rol de
hlr_interworking>",
  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://omnimessage.example.com: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 Independiente

```
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: 85+

Por Categoría:

- Modo Operacional: 6 parámetros
- Modo HLR: 18 parámetros
- Modo SMSc: 8 parámetros

- Modo STP: 5+ parámetros (más listas de m3ua_peers, m3ua_routes, m3ua_gt_routes)
- Modo CAMEL GW: 14 parámetros
- NAT de Título Global: 2 parámetros
- Conexión M3UA: 8 parámetros
- Estado del Subsistema SCCP: 1 parámetro
- Puerta de Enlace USSD: 5 parámetros
- Tiempo de Ejecución / Supervisión: 4 parámetros
- Infraestructura (Licencia, Web, API, Registro): 23 parámetros
- Base de Datos: 1 parámetro

Parámetros Requeridos por Modo:

Modo HLR:

- `hlr_api_base_url`
- `hlr_api_key`
- `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 la limpieza automática está habilitada)
- Todos los parámetros `map_client_m3ua.*` (8)

Modo STP:

- `sctp_handler.point_code` (si el controlador SCTP está habilitado)
- `sctp_handler.local_ip`
- `sctp_handler.local_port`

Modo CAMEL GW:

- `cgrates_url` (si CGrateS está habilitado)

- Todos los parámetros `cap_client_m3ua.*` (8)

Infraestructura:

- `license_client.license_server_api_urls`
- `license_client.licensee`

Documentación Relacionada

- **Guía de HLR** - Configuración específica de HLR
- **Guía de SMSc** - Configuración específica de SMSc
- **Guía de STP** - Configuración de enrutamiento STP
- **Guía de API** - Referencia de la 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 característica 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 al operar con múltiples Títulos Globales y necesitar asegurar que las respuestas utilicen el GT correcto según qué red o par está llamando y/o qué GT llamaron.

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/de respaldo)

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 de respaldo - menor prioridad
  %{weight: 100, response_gt: "999999"}
]
```

Casos de uso

Operación Multired

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:

- Los suscriptores de la **red de origen** utilizan GT `555000`
- El **socio de roaming 1** utiliza GT `555001`
- El **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 GT 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
- Agradecimientos 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 predeterminada (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 prefijo

Formato de Regla

Cada regla es un mapa con las siguientes claves:

```
%{
  calling_prefix: "8772",      # (Opcional) Prefijo para coincidir
con GT de llamada
  called_prefix: "555",        # (Opcional) Prefijo para coincidir
con 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 como una coincidencia de prefijo de cadena
  - 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 como una coincidencia de prefijo de cadena
  - 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
  - Peso menor = mayor prioridad (procesado primero)
  - Debe ser >= 0
  - Se utiliza como criterio de ordenamiento 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 GT 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/de respaldo.

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 de prefijo combinada (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 de respaldo
  %{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 predeterminado de SMSc (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"},

    # Predeterminado: usar GT estándar de SMSc (comodín de
respaldo)
    %{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 HLR
predeterminado

  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"},

    # Predeterminado 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
(predeterminado)

  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 aún 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 deseas 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"},

    # Respaldo predeterminado
    %{weight: 100, response_gt: "555000"}
  ]
```

Flujo de Tráfico:

```
Solicitud entrante a GT Llamada: 555100 (tu GT de SMS)
GT Llamante: 441234567 (cualquier llamador)

Búsqueda de GT NAT:
  GT Llamada "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 Prefijos de Llamada + Llamada (AVANZADO)

Escenario: Diferentes socios llaman a diferentes GTs y deseas 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"},

    # Respaldo predeterminado - baja prioridad (peso 100)
    %{weight: 100, response_gt: "555000"}
  ]
```

Ejemplos de Coincidencia:

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

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

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

# Llamador desconocido llama a GT de Voz
Llamada: "999999999", Llamada: "555200"
→ Coincide con peso 100 comodín → "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 operaciones HLR, consulta la Guía de Configuración de HLR.

Configuración:

```
config :omniss7,
  hlr_mode_enabled: true,
  hlr_service_center_gt_address: "5551234567",  # GT HLR
predeterminado

  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
predeterminado
  ]
```

Modo SMSc

GT NAT afecta:

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

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

Configuración:

```
config :omniss7,
  smsc_mode_enabled: true,
  smsc_service_center_gt_address: "5559999",  # GT SMSc
predeterminado

  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
predeterminado
  ]
```

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.)
- Agradecimientos de RequestReportBCSMEvent
- Respuestas de ApplyCharging
- Respuestas de Continue

Configuración:

```
config :omniss7,
  camelgw_mode_enabled: true,
  camel_gsmscf_gt_address: "55512341112",  # GT gsmSCF
predeterminado

  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
predeterminado
  ]
```

Caso de uso: Al operar como un gsmSCF (Función de Control de Servicio) para múltiples redes, cada gsmSSF de la red puede esperar respuestas de un GT específico de gsmSCF. GT NAT asegura que se utilice el GT correcto según qué gsmSSF esté 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
Respuesta 55512341112
[info] GT NAT [UpdateLocation ISD]: GT Llamante 331234567 -> GT
Respuesta 55512341111
[info] GT NAT [respuesta MAP BEGIN]: GT Llamante 441234567 -> GT
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"` - Agradecimiento de ISD
- `"respuesta de error HLR"` - Respuesta de error del 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 GT de llamada solamente (called_gt es nil)
iex> GtNat.translate_response_gt("877234567", nil, "default_gt")
"55512341112"

# Probar traducción con ambos GT de llamada y llamada
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 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 Respuesta 55512341112
"55512341112"

# Probar sin coincidencia (devuelve predeterminado)
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 predeterminada 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, peso bajo
  %{weight: 100, response_gt: "111111"}  # Comodín, peso alto
(respaldo)
]
```

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
- ✅ Agradecimientos de ISD
- ✅ Respuestas MAP BEGIN

Si un tipo de mensaje específico no utiliza GT NAT, puede que aún no esté implementado. Revisa el código fuente o contacta al 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 predeterminada (prefijo vacío) debe ser la última

Benchmark (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 Respaldo Comodín

```
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 peso alto
]
```

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 GT: 4412xxxxxxx)
  # Peso 10: Prioridad estándar del socio
  %{calling_prefix: "4412", weight: 10, response_gt: "5551001"},

  # Socio ABC - red de Francia (rango 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 a nivel INFO para ver todas las traducciones de GT NAT en producción.

Integración con Otras Características

Modo STP

GT NAT funciona independientemente del enrutamiento STP. STP enruta basado en 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 basado en el GT de gsmSSF entrante

Configuración:

- `camel_gsmscf_gt_address` - GT gsmSCF predeterminado (opcional)
- Si no está configurado, utiliza el GT de Parte Llamada de la solicitud entrante
- Las reglas de GT NAT anulan el predeterminado basado en 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 se puede combinar 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 de
respaldo
  ]
```

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 retroceso listo (configurar `gt_nat_enabled: false`)

Soporte

Para problemas o preguntas:

- Revisa los registros en busca de mensajes "GT NAT"
- Valida la configuración con `GtNat.validate_config()`
- Revisa la sección de solución de problemas de esta guía
- Contacta al 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 en el 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 en el hogar (HSS) completo. Esta arquitectura separa las preocupaciones:

- **OmniSS7 (Frontend HLR)**: Maneja toda la señalización del protocolo SS7/MAP, enrutamiento SCCP y comunicación de red
- **OmniHSS (Backend HSS)**: Gestiona datos de suscriptores, autenticación, aprovisionamiento 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 a un solo MSISDN para roaming internacional, cambio de red y aprovisionamiento de eSIM
- **Autenticación Flexible**: Soporte para algoritmos de autenticación Milenage (3G/4G/5G) y COMP128 (2G)
- **Seguimiento de Sesiones de Circuito y Paquete**: Seguimiento independiente de registros de red CS (circuito) y PS (paquete)
- **Aprovisionamiento Avanzado**: Perfiles de servicio personalizables, servicios suplementarios y datos de suscripción CAMEL

- **Diseño API-First**: API HTTP RESTful para integración con sistemas de facturación, CRM y aprovisionamiento
- **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 únicamente**. Toda la lógica de gestión de suscriptores, algoritmos de autenticación, reglas de aprovisionamiento y operaciones de base de datos son manejadas por OmniHSS. Esta guía cubre la configuración del protocolo SS7/MAP en OmniSS7. Para información sobre aprovisionamiento 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 SIMs 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 aprovisionamiento adecuados para cada IMSI.

Tabla de Contenidos

1. Integración de OmniHSS
2. Soporte Multi-IMSI
3. ¿Qué es el Modo HLR?
4. Habilitando el Modo HLR
5. Base de Datos de Suscriptores
6. Vectores de Autenticación
7. Actualizaciones de Ubicación
8. Integración CAMEL
9. Manejo de Suscriptores en Roaming
10. Operaciones HLR

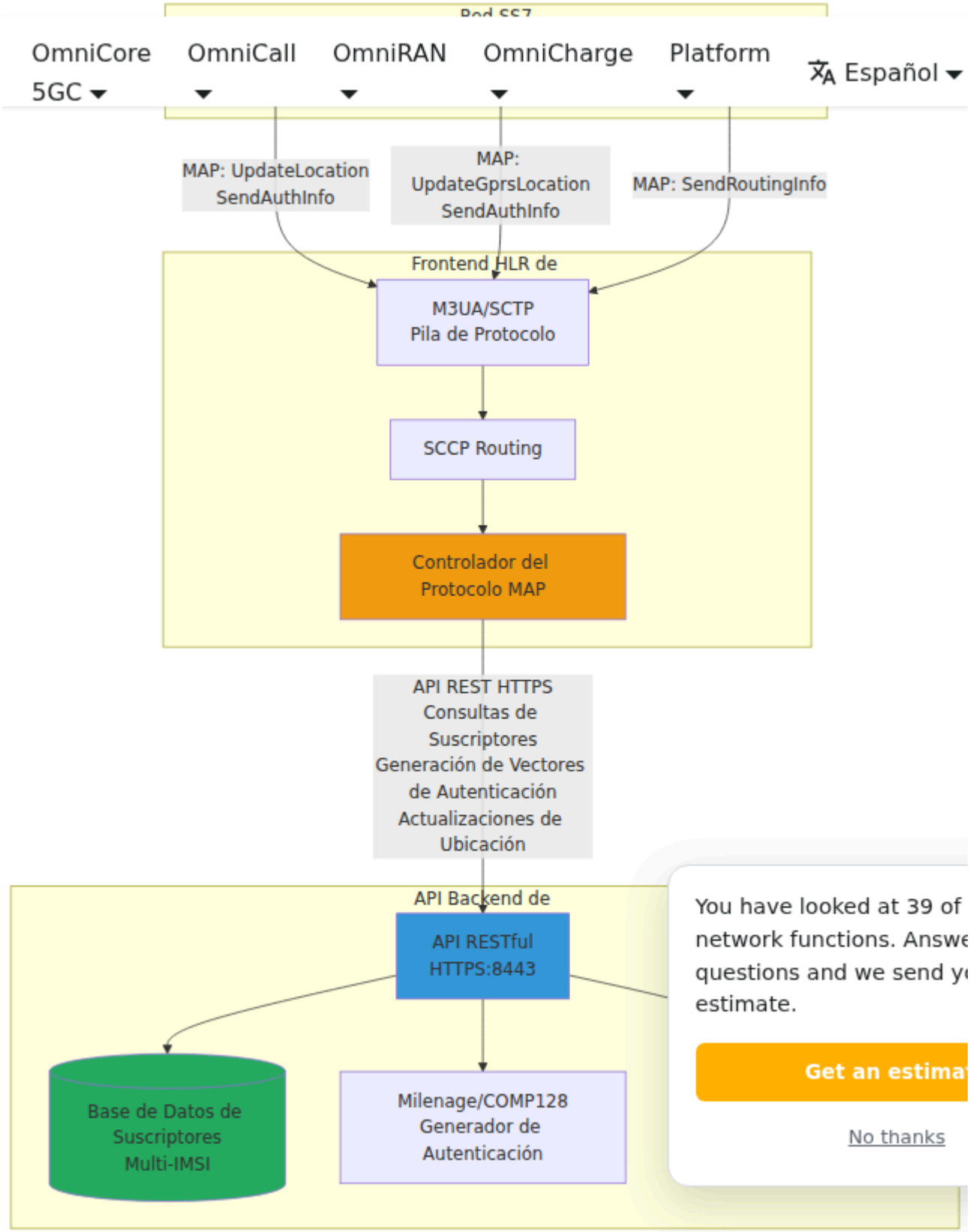
- Mapeo de Campos de Respuesta
  - SendRoutingInfo (SRI)
  - UpdateLocation / ISD
  - SendRoutingInfoForSM
- Resumen de Fuentes de Campo

¿Qué es el Modo HLR?

El 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 VLRs
- **Información de Enrutamiento**: Proporcionar información de enrutamiento para llamadas y SMS

Arquitectura HLR



Habilitando el Modo HLR

OmniSS7 puede operar en diferentes modos (STP, HLR, SMSc). El modo se selecciona a través de un conjunto de banderas de modo y la configuración asociada en **`config/config.exs`**, el archivo de configuración predeterminado de la aplicación.

Cambiando a Modo HLR

Para ejecutar OmniSS7 como un HLR, configure las banderas de modo para que las características de HLR estén habilitadas y las características de SMSc estén deshabilitadas, luego agregue la configuración específica de HLR:

1. **Abra** `config/config.exs`
2. **Configure** las banderas de modo bajo `config :omniss7`:
   - `map_client_enabled: true`: requerido para enviar y recibir operaciones MAP
   - `hlr_mode_enabled: true`: habilita el manejo específico de HLR
   - `smsc_mode_enabled: false`: deshabilita las características de SMSc
3. **Agregue** los parámetros de configuración de HLR (punto final de API, direcciones GT, configuraciones ISD/CAMEL, conexión M3UA) que se muestran a continuación
4. **Personalice** los parámetros para su implementación
5. **Reinicie** la aplicación para que los cambios surtan efecto

> **Nota**: `config/runtime.exs` es un stub mínimo utilizado solo para pruebas automatizadas. No contiene definiciones de modo operativo. Toda la configuración de modo y HLR vive en `config/config.exs` (con sobrescrituras específicas del entorno en archivos como `config/test.exs`). Las implementaciones en producción proporcionan su propio `config/config.exs`.

Configuración del Modo HLR

La configuración completa de HLR (en `config/config.exs`) se ve así:

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

  # Configuración de la API Backend de OmniHSS
  hlr_api_base_url: "https://omnihss.example.com:8443",
  hlr_api_key: "<Clave API de OmniHSS con el rol
hlr_interworking>",

  # 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 Barrado de Llamadas)
  isd_send_call_barring: true,

  # Configuración CAMEL (para respuestas SendRoutingInfo)
  # Clave de Servicio para la iniciación del servicio 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 Origen
  # Lista de prefijos de dirección de VLR que se consideran red
"local"
```

```
  # Si el VLR del suscriptor comienza con uno de estos prefijos,
use la respuesta SRI estándar
  # De lo contrario, el suscriptor está en roaming y necesitamos
enviar PRN para obtener MSRN
  # Por defecto es [] (sin prefijos = todos los VLRs tratados como
roaming) si se omite
  home_vlr_prefixes: ["555123"],

  # 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 para Personalizar

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

Parámetro	Tipo	Predeterminado
`hlr_api_base_url`	String	*Requerido*
`hlr_api_key`	String	*Requerido*
`hlr_service_center_gt_address`	String	*Requerido*
`smsc_service_center_gt_address`	String	*Requerido*
`hlr_smsc_alert_gts`	List	`[]`
`hlr_alert_location_expiry_seconds`	Integer	`172800`

Parámetro	Tipo	Predeterminado
`hlr_imsi_plmn_prefix`	String	`"50557"`
`hlr_msisdn_country_code`	String	`"61"`
`hlr_msisdn_nsn_offset`	Integer	`0`
`hlr_msisdn_nsn_length`	Integer	`9`
`isd_network_access_mode`	Atom	`:packetAndCircuit`
`isd_send_ss_data`	Boolean	`true`

Parámetro	Tipo	Predeterminado
`isd_send_call_barring`	Boolean	`true`
`camel_service_key`	Integer	`11_110`
`camel_trigger_detection_point`	Atom	`:termAttemptAuthorized`
`hlr_camel_subscriptions`	List	`[]`
`home_vlr_prefixes`	List	`[]`
`local_ip`	Tuple	*Requerido*
`local_port`	Integer	`2905`

Parámetro	Tipo	Predeterminado
`remote_ip`	Tuple	*Requerido*
`remote_port`	Integer	`2905`
`routing_context`	Integer	`1`

Qué Sucede Cuando el Modo HLR Está Habilitado

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

- **Eventos SS7** - Registro de eventos
- **Cliente SS7** - Pruebas de operaciones MAP
- **Pares** - Estado de conexión (pares M3UA/SCTP)
- **Enlaces HLR** - Estado de la API HLR + gestión de suscriptores ← *específico de HLR*
- **Recursos** - Monitoreo del sistema
- **Configuración** - Visor de configuración

Las pestañas de **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

- **Clave API de OmniHSS**: `hlr_api_key` es obligatoria. OmniSS7 la envía en el encabezado `x-omnihss-key` en cada solicitud a OmniHSS, y también en la verificación de estado de Enlaces HLR. Si falta o está vacía, cada solicitud a OmniHSS falla con un error que menciona `hlr_api_key`. OmniSS7 no envía una solicitud sin la clave. Si la clave es incorrecta, OmniHSS devuelve HTTP 401.
- **Tiempo de Espera de Solicitud API**: Todas las solicitudes a la API de OmniHSS tienen un **tiempo de espera de 5 segundos codificado**
- **Tiempo de Espera de Solicitud MAP**: Todas las solicitudes MAP (SRI, UpdateLocation, SendAuthInfo, etc.) tienen un **tiempo de espera de 10 segundos codificado**
- **Tiempo de Espera ISD**: Cada mensaje InsertSubscriberData (ISD) en una secuencia UpdateLocation tiene un **tiempo de espera de 10 segundos codificado**
- Se requiere conexión M3UA a STP para recibir operaciones MAP
- Después de cambiar de modo, debe 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 a la API**: Consulte la Guía de API para la 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 e 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 disparo y direcciones gsmSCF
- **Servicios suplementarios**: Desvío de llamadas, barrado, 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 para 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 del suscriptor, actualizar datos de ubicación y generar vectores de autenticación:

```
config :omniss7,
  hlr_api_base_url: "https://omnihss-server:8443",
  hlr_api_key: "<Clave API de OmniHSS con el rol
hlr_interworking>"
```

Cada solicitud a OmniHSS contiene el encabezado `x-omnihss-key` con el valor de `hlr_api_key`. OmniHSS utiliza la clave para encontrar el rol del cliente. La clave HLR debe tener el rol `hlr_interworking`. Este rol da acceso a la búsqueda de suscriptores, conjunto de claves, generación de vectores de autenticación y puntos finales de sesión de circuito que utiliza el HLR.

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

- **Recuperar datos del suscriptor** por IMSI o MSISDN
- **Generar vectores de autenticación** utilizando las credenciales Ki/OPc almacenadas
- **Actualizar la ubicación de la sesión de circuito** cuando los suscriptores realizan UpdateLocation
- **Verificar el estado del suscriptor** y los 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 exista y esté habilitado
3. **Actualizar la sesión de circuito** en OmniHSS con la nueva ubicación VLR/MSC
4. **Enviar mensajes InsertSubscriberData (ISD)** para aprovisionar al suscriptor en el nuevo VLR
5. **Devolver respuesta de UpdateLocation** al VLR (incluye GT de HLR de `hlr_service_center_gt_address`)
6. **Enviar alertServiceCenter** a los GTs de 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 de UpdateLocation. Esto permite que el VLR/MSC identifique y enrute mensajes de regreso a este HLR.

Integración del Centro de Servicio de Alerta

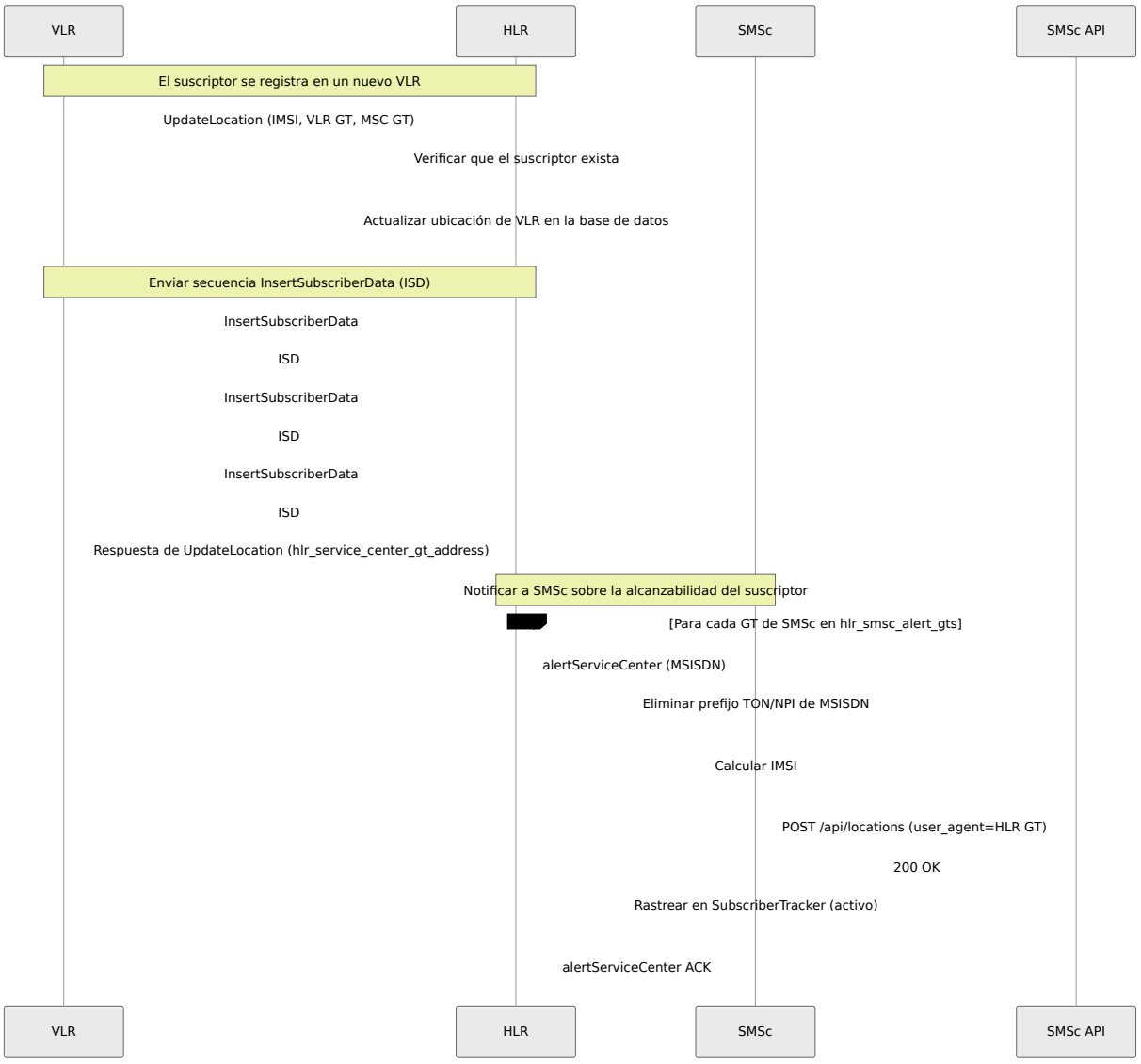
Después de una actualización de ubicación exitosa, el HLR puede notificar automáticamente a los sistemas SMSc que un suscriptor ahora es alcanzable enviando mensajes **alertServiceCenter** (código de operación MAP 64). Para información sobre cómo el SMSc maneja estas alertas, consulte Manejo del Centro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

Configuración

Configure la lista de Títulos Globales de SMSc a notificar:

```
config :omniss7,
  # Lista de GTs de 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 de 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 del llamante
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/locations)
- **Establece el campo `user_agent`** al GT de 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`
- Rastrear al suscriptor en el Tracker de Suscriptores de SMSc para monitoreo

Pruebas

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

1. Navega a la pestaña "Suscriptores Activos"
2. Encuentra la sección "Probar Alert Service Center"
3. Ingresa MSISDN, SMSc GT y HLR GT (los valores predeterminados se completan automáticamente desde la configuración)
   - SMSc GT predeterminado es la primera entrada en `hlr_smsc_alert_gts`
   - HLR GT predeterminado es `hlr_service_center_gt_address`
4. Haz clic en "Enviar alertServiceCenter"

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

Configuración de InsertSubscriberData (ISD)

Después de un UpdateLocation exitoso, el HLR envía datos de aprovisionamiento del suscriptor al VLR utilizando mensajes de **InsertSubscriberData** (ISD). La configuración de ISD te permite personalizar qué datos se envían y cómo.

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

Secuencia de ISD

El HLR puede enviar hasta 3 mensajes ISD secuenciales:

1. **ISD #1** (Siempre enviado) - Datos básicos del suscriptor:
    - IMSI
    - MSISDN
    - Categoría del 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 llamada
    - Retención de llamada
    - Servicio de múltiples partes
    - Estado y características del servicio suplementario
3. **ISD #3** (Opcional) - Datos de Prohibición de Llamadas:
    - Prohibición de todas las llamadas salientes (BAOC)
    - Prohibición 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 Prohibición 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 paquetes (GPRS/LTE) como conmutado por circuitos (voz)	Predeterminado - Suscriptores de servicio completo
`:packetOnly`	Solo conmutado por paquetes (datos/LTE)	Tarjetas SIM solo de datos, dispositivos IoT
`:circuitOnly`	Solo conmutado por circuitos (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 (Aprovisionamiento mínimo):

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

Enviar básico + servicios suplementarios (Sin prohibición 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 de SS [si
está habilitado]
VLR → HLR: ISD #2 ACK (CONTINUAR)
HLR → VLR: InsertSubscriberData #3 (CONTINUAR) - Prohibición de
llamadas [si está habilitado]
VLR → HLR: ISD #3 ACK (CONTINUAR)
HLR → VLR: Respuesta de UpdateLocation (FIN)
```

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

Mejores Prácticas

- **Configuración Predeterminada**: Usa `:packetAndCircuit` y habilita todos los ISDs para máxima compatibilidad
- **IoT/M2M**: Usa `:packetOnly` y desactiva datos de SS/prohibición de llamadas para dispositivos solo de datos

- **Interoperabilidad**: Algunos VLR más antiguos pueden no soportar todos los servicios suplementarios - desactiva `isd_send_ss_data` si encuentras 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 de llamadas inteligente y control de servicios.

Para referencia de parámetros de configuración, consulta 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 inteligentes en redes GSM/UMTS. Permite a los operadores de red implementar servicios de valor agregado como:

- Facturación prepago
- Filtrado y prohibición 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 de SendRoutingInfo)
# Clave de servicio para la iniciación del servicio 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 Servicios). Este es un identificador numérico configurado en tu 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 la configuración 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 (disparar antes del enrutamiento):

```
camel_trigger_detection_point: :termAttemptAuthorized,
```

Manejo personalizado de ocupado (disparar cuando está ocupado):

```
camel_trigger_detection_point: :tBusy,
```

Facturación basada en respuesta (disparar al responder):

```
camel_trigger_detection_point: :tAnswer,
```

Respuesta SRI con CAMEL

Cuando está configurado, las respuestas de 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 Predeterminado: continueCall

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

Mejores Prácticas

- **Redes de Producción**: Usa claves de servicio estandarizadas acordadas con tu proveedor de gsmSCF
- **Pruebas**: Usa `: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 gsmSCF no está disponible

Suscripciones CAMEL en InsertSubscriberData

Durante UpdateLocation, el HLR envía un InsertSubscriberData más que lleva `vlrCamelSubscriptionInfo` cuando el IMSI del suscriptor coincide con una regla en `hlr_camel_subscriptions`. La primera regla coincidente gana. El MSC utiliza el O-CSI para llamadas MO (DP collectedInfo) y el VT-CSI para llamadas MT (DP termAttemptAuthorized), y abre el diálogo CAP hacia el título global `gsmscf_address`, por lo que el gsmSCF es elegido por el HLR, no por la configuración del MSC.

```
config :omniss7,
  hlr_camel_subscriptions: [
    %{
      imsi_prefix: "3133800000009",
      o_csi: %{service_key: 100, gsmscf_address: "14155550300",
default_call_handling: :release_call},
      t_csi: %{service_key: 101, gsmscf_address: "14155550300",
default_call_handling: :continue_call}
    }
  ]
```

Clave	Descripción
`imsi_prefix`	Prefijo IMSI al que se aplica la regla
`o_csi` / `t_csi`	Al menos uno. `service_key` (InitialDP serviceKey), `gsmscf_address` (gsmSCF E.164 GT), `default_call_handling` (`:release_call` o `:continue_call`, aplicado por el MSC cuando el gsmSCF no responde)

Los CSIs se envían con manejo de capacidad CAMEL 2 (CAP v2). Una regla con una clave faltante genera un error nombrando la clave.

Manejo de Suscriptores en Roaming

Detección de VLR Local vs VLR en Roaming

Cuando el HLR recibe una solicitud de **SendRoutingInfo** (SRI), necesita determinar si el suscriptor está en un VLR "local" (dentro de tu red) o en un VLR en roaming (visitando otra red). El comportamiento difiere según esta determinación:

Para referencia de parámetros de configuración, consulta Prefijos de VLR Local en la Referencia de Configuración.

- **VLR Local**: Devuelve respuesta SRI estándar con información de enrutamiento CAMEL
- **VLR en Roaming**: Envía una solicitud de Proveer Número de Roaming (PRN) para obtener un MSRN, luego lo devuelve en la respuesta SRI

Configuración

```
# Prefijos de VLR Local
# Lista de prefijos de dirección VLR que se consideran red "local"
# Si la dirección VLR del suscriptor comienza con uno de estos
prefijos, usa la 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:

```
# Red local única
home_vlr_prefixes: ["555123"],

# Múltiples redes locales (por ejemplo, diferentes regiones o
subsidiarias)
home_vlr_prefixes: ["555123", "555124", "555125"],
```

Cómo Funciona

1. Flujo de Suscriptor Local (Estándar)

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

```
GMSC → HLR: SendRoutingInfo (MSISDN: "1234567890")
HLR consulta la API de backend para datos del suscriptor
HLR verifica la dirección VLR: "5551234567"
HLR determina: VLR comienza con "555123" → Red local
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 local:

```
GMSC → HLR: SendRoutingInfo (MSISDN: "1234567890")
HLR consulta la API de 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 de Roaming (MSRN): "49170999888777"
```

Diferencias en la Estructura de Respuesta

Respuesta SRI de Suscriptor Local

```
%{
  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 de 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 Roaming de Estación Móvil): 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 Local Única

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

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

Operador Multi-Región

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

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

Configuración de Pruebas

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

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

Mejores Prácticas

- **Selección de Prefijos**: Usa el prefijo único más corto que identifique los VLR de tu red (por ejemplo, código de país + código de red)

- **Múltiples Prefijos**: Incluye todos los prefijos de VLR en tu red, incluidos diferentes regiones y subsidiarias
- **Acuerdos de Roaming**: Asegúrate de que PRN esté debidamente soportado por las redes de socios de roaming
- **Pruebas**: Prueba a fondo tanto escenarios locales como en roaming antes del despliegue en producción
- **Monitoreo**: Monitorea 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**: Verifica las direcciones VLR en tu base de datos y actualiza los prefijos en consecuencia

Síntoma: Solicitudes PRN que se agotan

- **Causa**: Problemas de conectividad de red con el MSC/VLR de roaming asociado
- **Solución**: Verifica 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**: Revisa los registros de respuesta PRN y ajusta `extract_msrn_from_prn/1` si es necesario

---

Operaciones HLR

Operaciones MAP Soportadas

OmniSS7 despacha operaciones MAP entrantes por opcode (independientemente de la fase TCAP). Las siguientes operaciones entrantes son manejadas:

Opcode	Operación	Propósito
2	`updateLocation`	Registrar ubicación VLR (dispara secuencia ISD + alertServiceCenter opcional)
23	`updateGprsLocation`	Registrar ubicación SGSN (conmutado por paquetes)
3	`cancelLocation`	Desregistrar del antiguo VLR
7	`insertSubscriberData`	Recibir perfil del suscriptor (por ejemplo, al actuar como lado VLR)
22	`sendRoutingInfo` (SRI)	Proveer MSRN/enrutamiento CAMEL para llamadas (local vs roaming)
45	`sendRoutingInfoForSM` (SRI-for-SM)	Proveer IMSI + SMSC GT para enrutamiento SMS
44	`MT-Forward-SM`	Entrega de SMS móvil-terminado
46	`MO-Forward-SM`	Recepción de SMS móvil-origen
47	`reportSM-DeliveryStatus`	Informe de estado de entrega de SMS-GMSC
64	`alertServiceCenter`	Alerta de suscriptor alcanzable al SMSc
66	`readyForSM`	MSC/SGSN indica que el suscriptor está listo para SMS
56	`sendAuthenticationInfo`	Generar vectores de autenticación

Opcode	Operación	Propósito
57	`restoreData`	Restauración de datos del suscriptor en VLR
67	`purgeMS`	Purgar un registro de suscriptor en el HLR
59	`processUnstructuredSS-Request`	USSD (manejado a través de la puerta de enlace USSD cuando está habilitado)
71	`anyTimeInterrogation` (ATI)	Interrogación de ubicación/estado del suscriptor
10	`registerSS`	Registro de servicio suplementario (por ejemplo, Desvío de Llamadas Incondicional)

Los opcodes no manejados se responden con un error `facilityNotSupported` (21).

Mapeo de Campos de Respuesta

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

Respuesta SendRoutingInfo (SRI)

Propósito: Proveer 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 local o en roaming:

Respuesta de Suscriptor Local (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 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	config.exs	Identificador de servicio CAMEL (`camel_service_key`)
Punto de Detección de Disparador	config.exs	Cuándo activar CAMEL (`camel_trigger_detection_point`)
Manejo de Capacidad CAMEL	Estático	Nivel de soporte de fase CAMEL
Manejo de Llamadas Predeterminado	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 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 de Roaming (MSRN)	Respuesta PRN	MSRN obtenido de la solicitud ProvideRoamingNumber	'

Lógica de Decisión de Enrutamiento:

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

   Si VLR comienza con prefijo local:
     → Retornar información de enrutamiento CAMEL (flujo de
suscriptor local)

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

Flujo de Datos:

- OmniSS7 consulta a OmniHSS para información del suscriptor

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

Requisitos de Configuración:

```
# En config/config.exs
home_vlr_prefixes: ["555123"],  # Lista de prefijos de VLR locales
```

Respuestas de Error:

- Si `serving_vlr` y `serving_msc` son `null`: Devuelve error 27 (Suscriptor Ausente)
- Si no se encuentra al suscriptor: 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 de UpdateLocation con InsertSubscriberData

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

Respuesta FINAL de UpdateLocation

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

InsertSubscriberData #1 (Datos Básicos del Suscriptor)

Campo	Fuente	Descripción	Ejemplo
IMSI	Solicitud	Desde la solicitud de UpdateLocation	`"9999998765432`
MSISDN	API de OmniHSS	Número de teléfono del suscriptor desde OmniHSS	`"555123456"`
Categoría	Estático	Categoría del suscriptor	`"\n"` (0x0A)
Estado del Suscriptor	Estático	Estado del servicio	`:serviceGrante`
Lista de Servicios de Portadora	Estático	Servicios de portadora soportados	`[<<31>>]`
Lista de Teleservicios	Estático	Teleservicios soportados	`[<<17>>, "!", "\""]`
Modo de Acceso a la Red	config.exs	Acceso por paquete/circuito (`isd_network_access_mode`)	`:packetAndCirc`

InsertSubscriberData #2 (Servicios Suplementarios) - Opcional

Campo	Fuente	Descripción	Controlado Por
SS Provisionado	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 Participantes	Estático	Soporte para llamadas en conferencia	Config habilitado

ISD #2 incluye:

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

InsertSubscriberData #3 (Prohibición de Llamadas) - Opcional

Campo	Fuente	Descripción	Controlado Por
Información de Prohibición de Llamadas	Estático	Configuraciones de prohibición de llamadas	`isd_send_call_barring: true`
BAOC	Estático	Prohibición de Todas las Llamadas Salientes (código SS 146)	Config habilitado
BOIC	Estático	Prohibición 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 config/config.exs
- ISD #3: Enviado solo si `isd_send_call_barring: true` en config/config.exs

Respuesta de 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.

El HLR responde en uno de dos modos, decidido por el **GT de la parte que llama SCCP** de la solicitud:

1. Enrutamiento en casa (predeterminado: cada originador externo). Este modo devuelve un IMSI sintético más el GT propio del IP-SM-GW como el número de nodo de la red. El HLR nunca expone IMSIs reales y nodos de servicio a operadores externos. El MT-ForwardSM regresa al IP-SM-GW, que luego maneja la entrega. Este es el comportamiento correcto para tráfico de interconexión y roaming.

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

2. Enrutamiento directo en la red (solo cuando la solicitud proviene de uno de nuestros propios nodos SMSc / IP-SM-GW). Cuando el GT que llama coincide con la lista configurada de `home_smsc_gts`, el HLR busca al suscriptor en su backend y devuelve el **IMSI real** y el **MSC de servicio real** como el número de nodo de la red, para que nuestro SMSc pueda enviar el MT-ForwardSM directamente al MSC. Si no hay un MSC de servicio disponible para ese suscriptor, se vuelve al enrutamiento en casa.

Campo	Fuente	Descripción	Ejemplo
IMSI	Backend del HLR	IMSI real del suscriptor	`"505570123456789"`
Número de Nodo de la Red	Backend del HLR	`serving_msc` real de la última actualización de UpdateLocation	`"5559876543"`

Parámetros de Configuración (de `config/config.exs`):

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

# Enrutamiento directo en la red: GTs de llamada SCCP de nuestros propios nodos SMSc / IP-SM-GW.
# Un SRI-for-SM de cualquiera de estos obtiene el MSC de servicio REAL en lugar de enrutamiento en casa.
# Todos los demás son enrutados en casa. Deje vacío ([]) para enrutarlos a todos en casa.
home_smsc_gts: ["5551234567"],                # Opcional (predeterminado [])

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

> **Nota:** El enrutamiento directo en la red solo funciona si el HLR realmente tiene el `serving_msc` del suscriptor. Es decir, el MSC de servicio realizó un MAP `updateLocation` a este HLR. Si el MSC se registra a través de otro mecanismo (por ejemplo, GSUP a un HLR separado), el backend no tiene la ubicación y la consulta se enruta a casa.

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 a utilizar al construir IMSIs sintéticos (por ejemplo, "50557" para MCC=505, MNC=57)
- `hlr_msisdn_country_code`: Código de país a anteponer al hacer mapeo inverso IMSI→MSISDN (por ejemplo, "61" para Australia, "1" para EE. UU./Canadá)
- `hlr_msisdn_nsn_offset`: Posición del carácter donde comienza el Número Nacional de Suscriptor (NSN) dentro del MSISDN (típicamente 0 si el MSISDN no incluye el código de país, o la 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 de 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** del suscriptor (número de teléfono).

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 de MSISDN a IMSI, OmniSS7 utiliza un esquema de codificación simple para **calcular** IMSIs sintéticos directamente desde el MSISDN. Este enfoque proporciona dos beneficios clave:

1. **Privacidad**: Los IMSIs reales de los suscriptores almacenados en la base de datos del 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 del 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 a la izquierda MSISDN con ceros 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 conservan los dígitos más a la derecha, se truncan los dígitos más a la izquierda (por ejemplo, `"91123456789"` → `"555123456"`)
- **Longitud de IMSI**: Siempre exactamente 15 dígitos

Mapeo Inverso (IMSI → MSISDN):

El SMSc puede invertir este mapeo para convertir IMSIs de vuelta a MSISDNs:

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

# Paso 1: Quitar el 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 nuevo de IMSI a MSISDN
- **Configuración Mínima**: Solo requiere `hlr_imsi_plmn_prefix`
- **Preservación de Privacidad**: Los IMSIs reales nunca se exponen
- **Sin Búsqueda en Base de Datos**: Cálculo rápido, no se necesitan llamadas a la API
- **Siempre 15 Dígitos**: El IMSI siempre es exactamente de 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**: Conservar 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 a la izquierda con ceros
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 IMSIs reales almacenados en la base de datos de suscriptores del HLR. Esto proporciona una capa adicional de protección de privacidad, ya que los IMSIs reales de los suscriptores solo se exponen cuando es absolutamente necesario (por ejemplo, durante operaciones de UpdateLocation o SendAuthenticationInfo que requieren vectores de autenticación reales).

Flujo de Respuesta:

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

2. Procesamiento del 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 de SMSC de la configuración: "5551234567"

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

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

Configuración:

Los siguientes parámetros se utilizan en `config/config.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",      # Usado para mapeo inverso
(IMSI→MSISDN)
hlr_msisdn_nsn_offset: 1,          # Omitir el código de país de 1
dígito
hlr_msisdn_nsn_length: 10          # Extraer NSN de 10 dígitos
```

Configuración de Extracción de NSN:

Si sus MSISDN incluyen el código de país (por ejemplo, `"99988000088"`" 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 de 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 el prefijo PLMN

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

2. **IMSI → MSISDN**: Quitar el prefijo PLMN, eliminar ceros a la izquierda, anteponer el 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 únicamente. No se

requieren llamadas a la API del backend.

Resumen de Fuentes de Campo

Tipo de Fuente	Descripción	Ejemplos
API de OmniHSS	Datos dinámicos de la base de datos de suscriptores de OmniHSS	IMSI, MSISDN, VLR/MSC de servicio desde circuit_session
config.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 config/config.exs:

- `hlr_service_center_gt_address` - Usado en respuestas de UpdateLocation

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

Opcional en config/config.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 de estado del suscriptor y perfil 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, aprovisionamiento, configuración de autenticación y operaciones

administrativas, consulte la **documentación del producto OmniHSS**. OmniHSS contiene toda la lógica de la base de datos de suscriptores, algoritmos de autenticación, reglas de aprovisionamiento de servicios y capacidades de gestión de Multi-IMSI.

OmniSS7 por Omnitouch Network Services

Guía de Configuración del Cliente MAP

← Volver a la Documentación Principal

Esta guía cubre cómo operar OmniSS7 como un **Cliente MAP**. En este rol, OmniSS7 se conecta a una red SS7 como un Proceso de Servidor de Aplicaciones (ASP). Emite operaciones MAP (Mobile Application Part) hacia HLRs, MSC/VLRs, SGSNs, GMLCs y otros elementos de la red central. El comportamiento del protocolo sigue 3GPP TS 29.002.

Tabla de Contenidos

1. ¿Qué es el Modo Cliente MAP?
2. Habilitando el Modo Cliente MAP
3. Referencia de Operaciones MAP
4. Ejemplos Comunes de Operaciones

5. Enviando Solicitudes a través de la API
6. Métricas y Monitoreo
7. Solución de Problemas

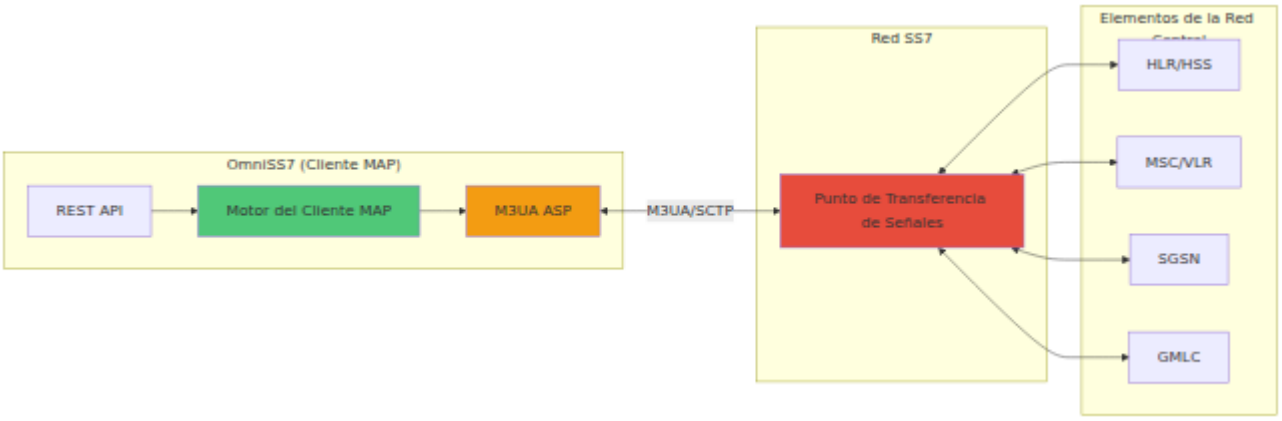
¿Qué es el Modo Cliente MAP?

El **Modo Cliente MAP** permite a OmniSS7 conectarse como un **Proceso de Servidor de Aplicaciones (ASP)** a un par M3UA (STP o SGP) a través de SCTP y intercambiar mensajes **MAP**. Cada operación se expone como un endpoint HTTP. La API codifica una solicitud como un diálogo TCAP/MAP y la transporta a través de SCCP y M3UA a la red SS7. OmniSS7 luego decodifica la respuesta de la red y la devuelve de manera sincrónica al llamador.

Los usos típicos incluyen:

- **Gestión de movilidad**: Actualizar ubicación, Cancelar ubicación, Purgar MS, Proveer número de roaming
- **Manejo de llamadas**: Enviar información de enrutamiento, Proveer información del suscriptor
- **SMS**: SRI-for-SM, MO/MT-ForwardSM, informes de estado de entrega
- **Autenticación**: Enviar información de autenticación
- **Datos del suscriptor**: Insertar/Eliminar datos del suscriptor, Interrogación en cualquier momento
- **Servicios suplementarios**: registrar/borrar/activar/interrogar SS, manejo de contraseñas
- **Servicios de localización (LCS)** y enrutamiento **GPRS**

Arquitectura de la Red



Habilitando el Modo Cliente MAP

El modo Cliente MAP se configura en `config/runtime.exs`. Para la referencia completa de conexión M3UA, consulte Parámetros de Conexión M3UA.

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

  # Conexión M3UA para el Cliente MAP (se conecta como un ASP a un
STP/SGP remoto)
  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
  }
```

Parámetros de `map_client_m3ua`

Parámetro	Tipo	Requerido	Predeterminado	Descrip
`mode`	Cadena	Sí	-	Rol M3UA. (para modo (se conecta STP/SGP).
`callback`	Tupla	Sí	-	`{Módulo, Función, A` invocado p cada carga M3UA entra Para el Clie MAP, esto e `{MapClient :handle_pa []}`.
`process_name`	Átomo	Sí	-	Nombre registrado proceso M3 ASP.
`local_ip`	Tupla	Sí	-	Dirección d enlace SCT como una t IP, por ejen `{10, 0, 0, 100}`.
`local_port`	Entero	No	2905	Puerto SCT 2905 es el M3UA regis por IANA.

Parámetro	Tipo	Requerido	Predeterminado	Descrip
`remote_ip`	Tupla	Sí	-	Dirección IP STP/SGP re como una t
`remote_port`	Entero	No	2905	Puerto SCT STP/SGP re
`routing_context`	Entero	No	-	Valor del Contexto d Enrutamier M3UA, cuar par requier

> Claves de nivel superior: `map_client_enabled` (Booleano, predeterminado `false`) activa el Cliente MAP; `map_client_m3ua` (Mapa) contiene la conexión anterior.

Se configura la dirección SCCP para operaciones salientes (Títulos Globales de Parte Llamante/Llamada, SSNs, códigos de punto) por separado. Consulte la Referencia de Configuración.

Referencia de Operaciones MAP

Cada operación a continuación es accesible como `POST /api/<endpoint>`. Los códigos de operación son los códigos de operación MAP de 3GPP TS 29.002. Las solicitudes son sincrónicas: la API se bloquea hasta que llega la respuesta SS7 o se agota el tiempo de espera de la solicitud (consulte Códigos de Respuesta).

Gestión de Movilidad

Operación	Endpoint	Opcode	Descripción
Actualizar Ubicación	`/api/updateLocation`	2	Registrar el VLR que sirve a un suscriptor con el HLR.
Cancelar Ubicación	`/api/cancelLocation`	3	Instruir a un VLR para eliminar un registro de suscriptor.
Proveer Número de Roaming	`/api/prn`	4	Obtener un MSRN del MSC que sirve.
Purgar MS	`/api/purgeMS`	67	Marcar un suscriptor como purgado en el HLR.
Enviar Identificación	`/api/sendIdentification`	55	Recuperar IMSI/datos de autenticación del VLR anterior.
Reiniciar	`/api/reset`	37	Reinicio iniciado por el HLR hacia un VLR.
Restaurar Datos	`/api/restoreData`	57	El VLR solicita la restauración de datos del suscriptor desde el HLR.

Manejo de Llamadas y Datos del Suscriptor

Operación	Endpoint	Opcode	
Enviar Información de Enrutamiento	`/api/sri`	22	Cons infor enru llam
Proveer Información del Suscriptor	`/api/provideSubscriberInfo`	70	Solic esta susc
Eliminar Datos del Suscriptor	`/api/deleteSubscriberData`	8	Elim susc
Interrogación en Cualquier Momento	`/api/anyTimeInterrogation`	71	gsm infor del s HLR.
Interrogación de Suscripción en Cualquier Momento	`/api/anyTimeSubscriptionInterrogation`	62	Cons susc
Modificación en Cualquier Momento	`/api/anyTimeModification`	65	Mod susc
Notificar Datos del Suscriptor Modificados	`/api/noteSubscriberDataModified`	5	El HI gsm cam

Operación	Endpoint	Opcode	
Notificar Evento MM	`/api/noteMM-Event`	89	Infor de g mov

SMS

Operación	Endpoint	Opcode	Descripción
Enviar Información de Enrutamiento para SM	`/api/sri-for-sm`	45	Consultar al HLR por el nodo que sirve para la entrega de SMS.
MT-Forward SM	`/api/MT-forwardSM`	44	Entregar un SM terminado en móvil al MSC/SGSN que sirve.
MO-Forward SM	`/api/forwardSM`	46	Enviar un SM originado en móvil al SMSC.
Enviar SM (PDU)	`/api/sendSM`, `/api/deliverPDU`	44	Construir y entregar un PDU SMS-DELIVER.
Reportar Estado de Entrega de SM	`/api/reportSM-DeliveryStatus`	47	Reportar el resultado de la entrega de SM al HLR.
Listo para SM	`/api/readyForSM`	66	Notificar al HLR que un MS es alcanzable para SMS.
Alertar Centro de Servicio	`/api/alertServiceCentre`	64	Alertar a un SMSC que un

Operación	Endpoint	Opcode	Descripción
			suscriptor está disponible.

Autenticación y Equipos

Operación	Endpoint	Opcode	Descripción
Enviar Información de Autenticación	`/api/send-auth-info`	56	Recuperar vectores de autenticación del HLR.
Enviar IMSI	`/api/sendIMSI`	58	Resolver un MSISDN a su IMSI en el HLR.
Verificar IMEI	`/api/checkIMEI`	43	Consultar estado del equipo en el EIR.

Servicios Suplementarios

Operación	Endpoint	Opcode	Descripción
Registrar SS	`/api/registerSS`	10	Registrar un servicio suplementario (por ejemplo, desvío de llamadas / CFU).
Borrar SS	`/api/eraseSS`	11	Borrar un servicio suplementario.
Activar SS	`/api/activateSS`	12	Activar un servicio suplementario.
Desactivar SS	`/api/deactivateSS`	13	Desactivar un servicio suplementario.
Interrogar SS	`/api/interrogateSS`	14	Interrogar el estado de un servicio suplementario.
Registrar Contraseña	`/api/registerPassword`	17	Registrar una contraseña de SS.
Obtener Contraseña	`/api/getPassword`	18	Recuperar/verificar una contraseña de SS.
Indicación de Comprobación de SS de Reenvío	`/api/forwardCheckSS-Indication`	38	Comprobación no confirmada hacia un VLR.

Servicios de Localización (LCS) y GPRS

Operación	Endpoint	Opcode	Descripció
Enviar Información de Enrutamiento para LCS	`/api/sendRoutingInfoForLCS`	85	GMLC consulta al HLR por enrutamient LCS.
Proveer Ubicación del Suscriptor	`/api/provideSubscriberLocation`	83	Solicitar la ubicación de un suscripto desde el nodo que sirve.
Informe de Ubicación del Suscriptor	`/api/subscriberLocationReport`	86	Informar la ubicación de un suscripto al GMLC.
Enviar Información de Enrutamiento para GPRS	`/api/sendRoutingInfoForGprs`	24	Consultar al HLR por enrutamient GPRS (selección d GGSN).
Actualizar Ubicación GPRS	`/api/updateGprsLocation`	23	Registrar el SGSN que sirve a un suscriptor con el HLR.

> **Alcance de LCS:** las operaciones de Servicios de Localización son **solo salientes (solicitante)**. El Cliente MAP las activa hacia un HLR, nodo que

> sirve o GMLC. El nodo no actúa como un *servidor* LCS (no responderá a un ProvideSubscriberLocation entrante ni procesará un SubscriberLocationReport entrante). Los codificadores llevan los elementos de información obligatorios más las opciones centrales mencionadas anteriormente. El conjunto de parámetros LCS más amplio (QoS, prioridad, formas GAD, palabra clave, número de referencia) aún no se expone.

CAP / CAMEL e Inyección Cruda

Operación	Endpoint	Descripción
CAP InitialDP	`/api/cap/initialDP`	Activar un diálogo CAMEL hacia un gsmSCF.
CAP Connect / Continue / ReleaseCall / ApplyCharging	`/api/cap/connect`, `/api/cap/continue`, `/api/cap/releaseCall`, `/api/cap/applyCharging`	Operaciones de control de llamadas CAMEL. Consulte la Guía del Gateway CAMEL.
Raw SCCP	`/api/raw/sccp`	Inyectar una carga útil SCCP construida a mano (pruebas/diagnósticos).
Raw M3UA	`/api/raw/m3ua`	Inyectar una carga útil M3UA construida a mano (pruebas/diagnósticos).

Ejemplos Comunes de Operaciones

Enviar Información de Enrutamiento para SM (SRI-for-SM)

Consulta al HLR por el nodo que sirve utilizado para entregar un SMS. Para cómo el lado del HLR procesa esto, consulte SRI-for-SM en la Guía del HLR.

Dirección SCCP (ruta de entrega de SMS de SMSc). Por defecto, el SRI-for-SM se envía **enrutado por GT dirigido al MSISDN de destino** (llamado SSN 6/HLR), lo que requiere que el STP mantenga una entrada GTT para ese rango de números. Para suscriptores **en la red**, el SMSc en su lugar dirige el SRI directamente al **HLR GT** configurado. Luego, enruta en la dirección de par (ya enrutable) del HLR, por lo que no se necesita GTT por rango, y el HLR devuelve el MSC que sirve real. Esto se selecciona por destino:

Configuración (`:omniss7`)	Propósito	Ejemplo
`hlr_gt_address`	GT del HLR local al que dirigir los SRI en la red	`"61455501111"`
`home_msisdn_prefixes`	Prefijos de MSISDN en la red; una coincidencia dirige al HLR GT	`["6777"]`

Un destino que coincide con una entrada de `home_msisdn_prefixes` (con `hlr_gt_address` configurado) se envía al HLR GT con SSN llamado 6; todo lo demás mantiene el predeterminado de enrutamiento por GT por MSISDN. La solicitud SRI-for-SM acepta `sccp_opts` (`:called_party`, `:called_ssn`) para anular la parte llamada SCCP para cualquier llamador.

Endpoint: `POST /api/sri-for-sm`

Solicitud:

```
{
  "msisdn": "447712345678",
  "serviceCenter": "447999123456"
}
```

Respuesta:

```
{
  "result": {
    "imsi": "234509876543210",
    "locationInfoWithLMSI": {
      "networkNode-Number": "447999555111"
    }
  }
}
```

cURL:

```
curl -X POST http://localhost/api/sri-for-sm \
  -H "Content-Type: application/json" \
  -d '{"msisdn": "447712345678", "serviceCenter": "447999123456"}'
```

Enviar Información de Enrutamiento (SRI)

Endpoint: `POST /api/sri`

Solicitud:

```
{
  "msisdn": "447712345678",
  "gmsc": "447999123456"
}
```

Proveer Número de Roaming (PRN)

Endpoint: `POST /api/prn`

Solicitud:

```
{
  "msisdn": "447712345678",
  "gmsc": "447999123456",
  "msc_number": "447999555111",
  "imsi": "234509876543210"
}
```

Enviar Información de Autenticación

Endpoint: `POST /api/send-auth-info`

Solicitud:

```
{
  "imsi": "234509876543210",
  "vectors": 5
}
```

Respuesta:

```
{
  "result": {
    "authenticationSetList": [
      {
        "rand": "0123456789ABCDEF0123456789ABCDEF",
        "xres": "ABCDEF0123456789",
        "ck": "0123456789ABCDEF0123456789ABCDEF",
        "ik": "FEDCBA9876543210FEDCBA9876543210",
        "autn": "0123456789ABCDEF0123456789ABCDEF"
      }
    ]
  }
}
```

Actualizar Ubicación

Registra el VLR que sirve a un suscriptor con el HLR, que responde con la secuencia de Insertar Datos del Suscriptor. Consulte Actualizaciones de

Ubicación en la Guía del HLR.

Endpoint: `POST /api/updateLocation`

Solicitud:

```
{
  "imsi": "234509876543210",
  "vlr": "447999555111"
}
```

Interrogación en Cualquier Momento (ATI)

Endpoint: `POST /api/anyTimeInterrogation`

Solicitud:

```
{
  "hlr_gt": "447999123456",
  "msisdn": "447712345678"
}
```

Categorías de Operaciones



Enviando Solicitudes a través de la API

Usando Swagger UI

La interfaz Swagger UI proporciona una interfaz interactiva para cada endpoint.

1. Navegue a `http://your-server/swagger`.
2. Seleccione un endpoint (por ejemplo, `/api/sri-for-sm`) y haga clic en **Probar**.
3. Complete el cuerpo de la solicitud y haga clic en **Ejecutar**.
4. La respuesta SS7 decodificada se muestra a continuación.

El documento OpenAPI en bruto está disponible en `http://your-server/swagger.json`.

Códigos de Respuesta de la API

Código	Significado
200	Éxito: el resultado decodificado está en el cuerpo de la respuesta.
400	Solicitud incorrecta: parámetros inválidos o faltantes.
504	Tiempo de espera de la puerta de enlace: no hay respuesta SS7 dentro del tiempo de espera de la solicitud (predeterminado 10 segundos).

Algunas operaciones son no confirmadas (fire-and-forget) y devuelven inmediatamente una vez que se envía el mensaje.

Métricas del Cliente MAP

Todas las métricas se exportan en el endpoint de Prometheus (`/metrics`).

Métrica: `map_requests_total` **Tipo:** Contador **Descripción:** Total de solicitudes MAP enviadas. **Etiquetas:** `operation`: por ejemplo `sri`, `sri_for_sm`, `prn`, `authentication_info`, `update_location`.

Métrica: `map_request_errors_total` **Tipo:** Contador **Descripción:** Total de solicitudes MAP que resultaron en un error o tiempo de espera. **Etiquetas:** `operation`.

Métrica: `map_request_duration_milliseconds` **Tipo:** Histograma **Descripción:** Distribución de los tiempos de ida y vuelta de las solicitudes MAP. **Etiquetas:** `operation`.

Métrica: `map_pending_requests` **Tipo:** Medidor **Descripción:** Solicitudes MAP que actualmente esperan una respuesta.

Ejemplos de Consultas Prometheus

```
# Solicitudes SRI-for-SM en la última hora
increase(map_requests_total{operation="sri_for_sm"}[1h])

# Tiempo de respuesta promedio de SRI
rate(map_request_duration_milliseconds_sum{operation="sri"}[5m])
  / rate(map_request_duration_milliseconds_count{operation="sri"}
[5m])

# Tasa de errores por operación
sum(rate(map_request_errors_total[5m])) by (operation)

# Solicitudes pendientes actuales
map_pending_requests
```

Solución de Problemas del Cliente MAP

Tiempo de Espera de Solicitudes (504)

Síntomas: La API devuelve 504; no hay respuesta del HLR/MSC.

Causas posibles:

- La asociación M3UA al STP/SGP no está ACTIVA.
- El STP no tiene ruta para el Título Global de destino.
- Dirección SCCP incorrecta (GT de Parte Llamante/Llamada, SSN) o Contexto de Enrutamiento.

Resolución:

1. Confirme el estado de la asociación M3UA en el panel de control o a través de las vistas de estado de la Guía del STP.
2. Verifique la conectividad de red (SCTP) al STP.
3. Confirme que el Título Global de destino sea enrutable y que el subsistema de destino (por ejemplo, HLR SSN 6) esté en servicio.
4. Revise los registros para mensajes de retorno SCCP (ver más abajo).

Mensajes de Retorno / Error SCCP

Síntomas: Los registros muestran mensajes de retorno del Servicio Unitdata SCCP (UDTS).

Causas comunes de retorno:

- **Sin Traducción para la Dirección**: el Título Global no está en la tabla de enrutamiento del STP.
- **Fallo del Subsistema**: el subsistema de destino (por ejemplo, HLR SSN 6) no está disponible.
- **Fallo de Red / Congestión**: problema de red transitorio.

Resolución:

- Confirme la configuración de enrutamiento/traducción del STP para el GT de destino.
- Verifique que el subsistema de destino esté operativo.
- Reintente después de que la congestión se despeje.

Documentación Relacionada

- ← Volver a la Documentación Principal
- Referencia de Configuración: todos los parámetros de configuración
- Guía de Características Comunes: Interfaz Web, API, monitoreo
- Guía del STP: enrutamiento y Traducción de Títulos Globales
- Guía del HLR: procesamiento de operaciones del lado del HLR
- Guía del Centro de SMS: entrega de SMS
- Guía del Gateway CAMEL: operaciones CAP/CAMEL

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)** 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 reintento, 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 reintento 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 solo 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 la congestión de la 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 reintento, 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

1. Integración de OmniMessage
2. ¿Qué es el Modo Centro de SMS?
3. Habilitando el Modo SMSc
4. Configuración de API HTTP
5. Flujos de Mensajes SMS
6. Manejo del Centro de Servicio de Alertas
7. Prevención de Bucles
8. Seguimiento de Suscriptores SMSc
9. Cacheo de Direcciones VLR
10. Configuración de Auto-Flush

11. Métricas y Monitoreo
12. Solución de Problemas

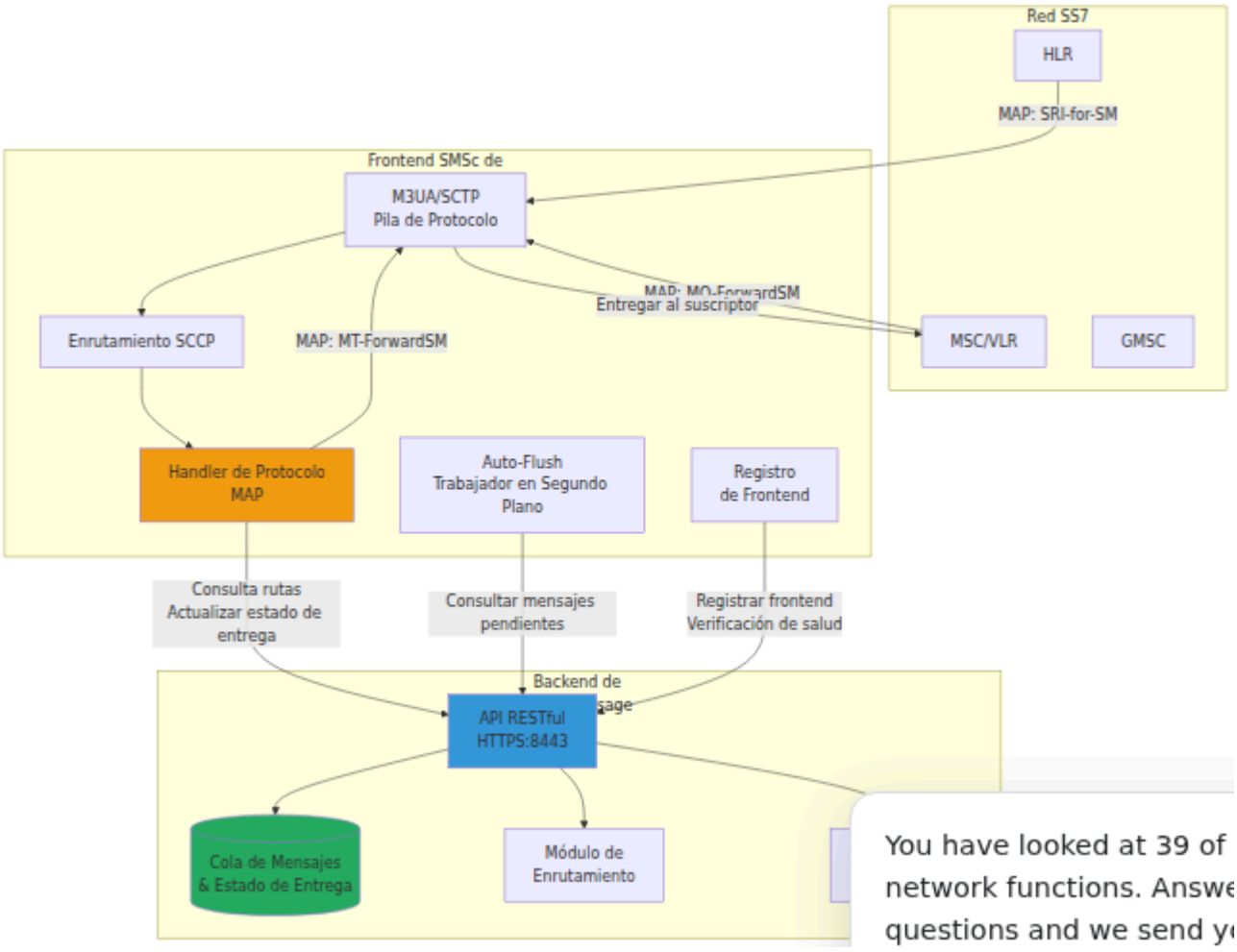
¿Qué es el Modo Centro de SMS?

Nota: Esta sección cubre solo 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 reintento
- **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 (STP, HLR, SMSc). El modo se selecciona a través de un conjunto de banderas de modo y la configuración asociada en `config/config.exs`, el archivo de configuración predeterminado de la aplicación.

Cambiando a Modo SMSc

Para ejecutar OmniSS7 como un SMSc, configure las banderas de modo para que las características de SMSc estén habilitadas, luego agregue la configuración específica de SMSc:

1. **Abra** `config/config.exs`
2. **Establezca** las banderas de modo bajo `config :omniss7`:
   - `map_client_enabled: true`: requerido para capacidades de enrutamiento/MAP
   - `smsc_mode_enabled: true`: habilita el manejo específico de SMSc
   - `hlr_mode_enabled: false`: desactiva las características de HLR
3. **Agregue** los parámetros de configuración de SMSc (punto final de API, nombre de SMSc, dirección GT, configuraciones de auto-flush, conexión M3UA) que se muestran a continuación
4. **Personalice** los parámetros para su implementación
5. **Reinicie** la aplicación para que los cambios surtan efecto

> **Nota**: `config/runtime.exs` es un stub mínimo utilizado solo para pruebas automatizadas. No contiene definiciones de modo operativo. Toda la configuración de modo y SMSc vive en `config/config.exs` (con sobreescrituras específicas del entorno en archivos como `config/test.exs`). Las implementaciones de producción proporcionan su propio `config/config.exs`.

Configuración del Modo SMSc

La configuración completa de SMSc (en `config/config.exs`) se ve así:

```
config :omniss7,
  # Banderas de modo - Habilitar características de STP + SMSc
  # Nota: map_client_enabled es true porque SMSc necesita
capacidades de enrutamiento
  map_client_enabled: true,
  hlr_mode_enabled: false,
  smsc_mode_enabled: true,

  # Configuración de API del Backend de OmniMessage
  smsc_api_base_url: "https://omnimessage.example.com:8443",
  # Identificación de SMSc para registro con el backend
  smsc_name: "ipsmgw",
  # Dirección GT del Centro de Servicio para operaciones de 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
  # Conectarse como ASP para enviar/recibir operaciones MAP SMS
  map_client_m3ua: %{
    mode: "ASP",
    callback: {MapClient, :handle_payload, []},
    process_name: :stp_client_asp,
    # Punto final local (sistema SMSc)
    local_ip: {10, 179, 4, 12},
    local_port: 2905,
    # Punto final remoto STP
    remote_ip: {10, 179, 4, 10},
    remote_port: 2905,
    routing_context: 1
  }
```

Las páginas de la interfaz web se registran por separado bajo `config :control_panel` en `config/config.exs`. La lista predeterminada `use_additional_pages` registra todas las páginas de OmniSS7 (~16 entradas); el tiempo de ejecución filtra las páginas que no se aplican al modo activo.

Tenga en cuenta que la página de estado de pares está etiquetada como **"Peers"**:

```
config :control_panel,
  use_additional_pages: [
    {SS7.Web.EventsLive, "/events", "Eventos SS7"},
    {SS7.Web.TestClientLive, "/client", "Cliente SS7"},
    {SS7.Web.M3UAStatusLive, "/m3ua", "Peers"},
    {SS7.Web.SCTPConnectionsLive, "/sctp", "Conexiones SCTP"},
    {SS7.Web.RoutingLive, "/routing", "Enrutamiento"},
    {SS7.Web.RoutingTestLive, "/routing_test", "Prueba de
Enrutamiento"},
    {SS7.Web.SmscLinksLive, "/smsc_links", "Enlaces SMSc"},
    {SS7.Web.SmscSubscribersLive, "/smsc_subscribers",
"Suscriptores SMSc"},
    {SS7.Web.HlrLinksLive, "/hlr_links", "Enlaces HLR"},
    # ... páginas adicionales (Suscriptores Activos, CAMEL,
Registros, Prueba MSISDN/IMSI, etc.)
  ],
  page_order: [
    "/application", "/configuration", "/events", "/logs",
"/client", "/m3ua",
    "/routing", "/routing_test", "/smsc_links",
"/smsc_subscribers", "/hlr_links",
    # ... rutas ordenadas restantes
  ]
```

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`	String	*Requerido*	**Punto f** backend OmniMe
`smsc_name`	String	`"{hostname}_SMSc"`	Su ident SMSc pa
`smsc_service_center_gt_address`	String	*Requerido*	Título G de Serv
`auto_flush_enabled`	Boolean	`true`	Habilita automá
`auto_flush_interval`	Integer	`10_000`	Interval procesa en milis
`auto_flush_dest_smsc`	String	`nil`	Filtro de SMSc de auto-flu procesa mensaje
`auto_flush_tps`	Integer	`10`	Tasa de de men (transac
`local_ip`	Tuple	*Requerido*	Direcció sistema
`local_port`	Integer	`2905`	Puerto S
`remote_ip`	Tuple	*Requerido*	Direcció conectiv

Parámetro	Tipo	Predeterminado	D
`remote_port`	Integer	`2905`	Puerto S
`routing_context`	Integer	`1`	ID de co enrutam

¿Qué Sucede 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 operación MAP
- **Peers** - Estado de conexión (pares M3UA/SCTP)
- **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 el `smsc_api_base_url` configurado
- **Registro de Frontend**: El sistema se registra automáticamente con OmniMessage cada **~85 segundos** (`@registration_interval` = 85,000 ms) a través del módulo `SMS.FrontendRegistry`

- **Tiempo de Espera de Solicitud de API**: Todas las solicitudes de 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
- Se requiere conexión M3UA a STP para enviar/recibir operaciones MAP SMS
- Después de cambiar de modo, debe 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 a API**: Consulte la Guía de API para la documentación de la API REST y acceso a Swagger UI

Configuración de 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 de suscriptores y registrarse como un frontend activo:

```
config :omniss7,
  # URL base de la API de OmniMessage
  smsc_api_base_url: "https://omnimessage.example.com:8443",
  # Identificador de nombre de SMSC para registro (predeterminado
a hostname_SMSc si está vacío)
  smsc_name: "omni-smsc01",
  # Dirección GT del Centro de Servicio para operaciones de SMS
  smsc_service_center_gt_address: "5551234567"
```

Parámetros de Configuración:

Parámetro	Tipo	Requerido	Predeterm
`smsc_api_base_url`	String	Sí	`"https://localh`
`smsc_name`	String	No	`""` (usa `"{hostname}_SMSc`
`smsc_service_center_gt_address`	String	No	`"5551234567"`

Registro de Frontend

El sistema se registra automáticamente con OmniMessage al iniciar y **vuelve a registrarse cada ~85 segundos** a través del módulo `SMS.FrontendRegistry`. Esto permite a OmniMessage:

- Rastrear frontends activos para balanceo de carga

- Monitorear el tiempo de actividad y el estado de salud
- Recopilar información de configuración
- Gestionar el enrutamiento de SMS distribuido a través de múltiples frontends

Detalles de Implementación:

- **Intervalo de Registro**: ~85 segundos (`@registration_interval 85_000`, 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 está configurado, se predetermina a `"{hostname}_SMSc"`.

Comunicación de 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** e informar el estado de salud
- **Enviar mensajes originados en móviles (MO)** recibidos de suscriptores
- **Recuperar mensajes terminados en móviles (MT)** de la cola para entrega
- **Actualizar el estado de entrega** con informes de éxito/fallo
- **Consultar información de enrutamiento** para el reenvío de mensajes

Endpoint	Método	Propósito	Cuerpo de
`/api/frontends`	POST	Registrar instancia de frontend	`{"frontend_nar` `"configuratior` `"frontend_type` `"hostname": ".` `"uptime_second`
`/api/messages_raw`	POST	Insertar nuevo mensaje SMS (TPDU en bruto)	`{"rp_originato` `"...",` `"rp_destinatio` `"...", "direct` `"message_body"` `"send_time": "` `"smsc_node_nam` `"source_smsc":` `"source_type":`
`/api/messages`	GET	Obtener cola de mensajes	Encabezado: `sm` `<smsc_name>`
`/api/messages/{id}`	PATCH	Marcar mensaje como entregado	`{"deliver_time` `"dest_smsc": "`
`/api/messages/{id}`	PUT	Actualizar estado del mensaje	`{"dest_smsc":`
`/api/locations`	POST	Insertar/actualizar ubicación del suscriptor	`{"msisdn": "."` `"...", "locati` `"ims_capable":` `"csfb": false,` `"...", "user_a` `"...", "ran_lo` `"...", "imei":` `"registered":`

Endpoint	Método	Propósito	Cuerpo de
`/api/events`	POST	Agregar seguimiento de eventos	`{"message_id"` `"name": "...",` `"description":`
`/api/status`	GET	Verificación de salud (la ruta es configurable a través de `smsc_api_status_path`, predeterminado `/api/status`; sondeado por la página de Enlaces SMSc, no por `SMSc.APIClient`)	-

Formato de Respuesta de API

Todas las respuestas de 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/1` - Insertar mensaje SMS en bruto (toma un solo mapa de mensaje; publica en `/api/messages_raw`)
- `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` - Agregar seguimiento de eventos para mensajes
- `flush_queue/2` - Procesar mensajes pendientes (SRI-for-SM + MT-forwardSM)
- `auto_flush/2` - Bucle de procesamiento continuo de colas

2. SMS.FrontendRegistry

Maneja el registro periódico del frontend con el backend:

- Se registra automáticamente al iniciar
- Vuelve a registrarse cada ~85 segundos
- Usa `smsc_name` de la configuración (vuelve a usar el nombre de host)
- Recopila información de configuración y tiempo de actividad del sistema

3. SMS.Utils

Funciones de utilidad para operaciones de SMS:

- `generate_tp_scts/0` - Generar marca de tiempo de SMS en formato TPDU

Flujos de Mensajes SMS

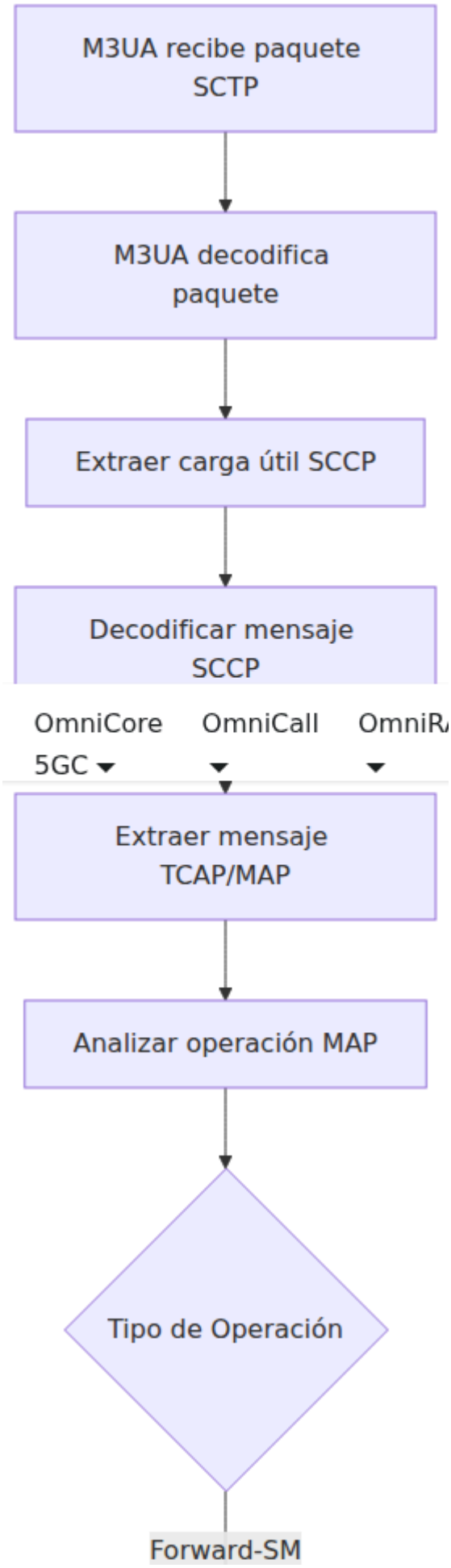
Flujo de SMS Entrante (Originado en Móvil)

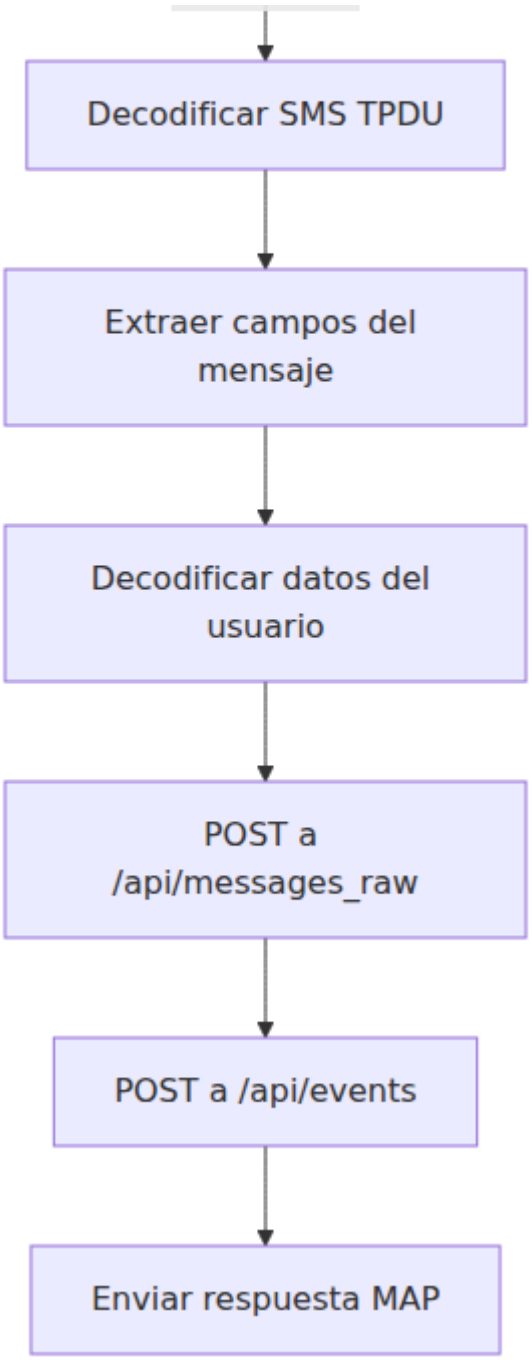

M3UA recibe paquete SCTP
M3UA decodifica paquete
Extraer carga útil SCCP
Decodificar mensaje SCCP
Extraer mensaje TCAP/MAP
Analizar operación MAP
Tipo de Operación
Forward-SM



Flujo de SMS Saliente (Terminado en 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 el IMSI real del suscriptor y sirviendo VLR. Un **cacheo de VLR** en el Rastreador 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:

- **Verificación de Cacheo de VLR**: Antes de consultar la API HSS, el sistema verifica el cacheo de VLR del Rastreador de Suscriptores para una entrada válida (IMSI + dirección VLR dentro de TTL). Un hit en el cacheo omite completamente la API HSS. Consulte Cacheo de Direcciones VLR para más detalles.

- **Entrega Directa de API HSS (en red)**: Cuando `hlr_api_base_url` está configurado, 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 ambas están disponibles, el IP-SM-GW envía el MT-ForwardSM directamente al VLR. Esto evita completamente SRI-for-SM.

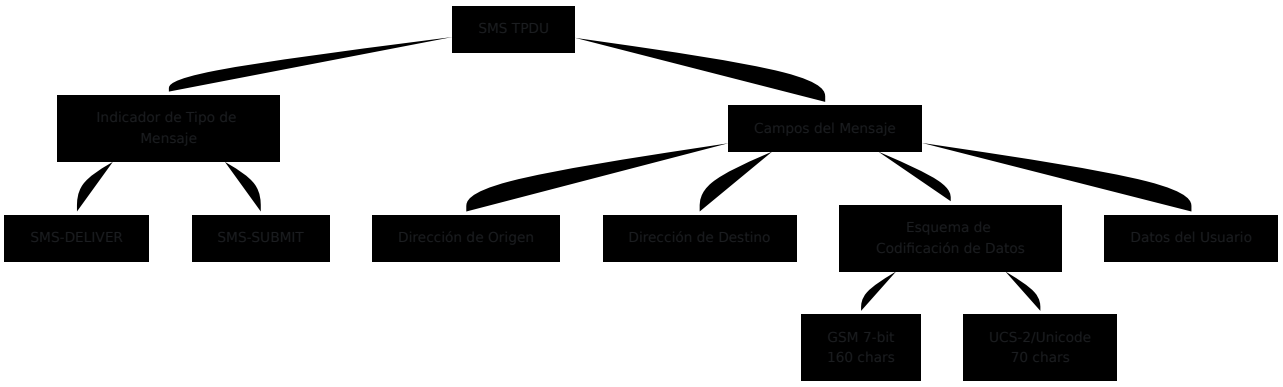
- **Suscriptor Ausente (en red)**: Si la API HSS encuentra al suscriptor pero no se le asigna un VLR de servicio, el mensaje se marca como fallido con una entrada de ubicación expirada. Se reintenta la entrega cuando el HLR envía un alertServiceCenter después de que el suscriptor se vuelve a registrar.

- **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 la Guía SRI-for-SM en HLR.

- **VLR Clear on Failure**: Cuando la entrega de MT-ForwardSM falla (tiempo de espera, error SCCP, etc.), la dirección VLR en caché se borra para que el próximo intento vuelva a consultar la API HSS para un VLR fresco.

- **MT-ForwardSM Timeout**: El diálogo MT-ForwardSM tiene un **tiempo de espera de 30 segundos** para acomodar respuestas más lentas de VLR 2G/CS.

Estructura del TPDU de SMS



Manejo del Centro de Servicio de Alerta

El SMSc puede recibir mensajes de **alertServiceCenter** del HLR para rastrear el estado de alcanzabilidad del suscriptor.

Para obtener información sobre cómo el HLR envía mensajes de alertServiceCenter, consulte Integración del Centro de Servicio de Alerta en la Guía HLR.

¿Qué es alertServiceCenter?

Cuando un suscriptor realiza una 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 de **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 el SMSc recibe
alertServiceCenter (predeterminado: 48 horas)
  hlr_alert_location_expiry_seconds: 172800
```

Comportamiento

Cuando el SMSc recibe un mensaje de 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, recurrir a generar un IMSI sintético utilizando el mismo mapeo que SRI-for-SM. El IMSI real es crítico para la entrega MT en la red a suscriptores 2G/CS. El MSC/VLR requiere el IMSI correcto en el MT-ForwardSM.
4. **POST a /api/locations**: 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 CS 2G/3G, no IMS/VoLTE)
   - `user_agent`: HLR GT que envió la alerta (para seguimiento)
   - `ran_location`: "SS7"
5. **Rastrear en el Rastreador de Suscriptores SMSc**: Registrar al suscriptor con HLR GT, 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/locations con `expires=0` para marcar al suscriptor como inalcanzable
3. **Agente de usuario**: Establecer en "SS7_AbsentSubscriber" para identificar la fuente
4. **Actualizar rastreador**: Marcar al suscriptor como `failed` 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



Endpoint de API

POST /api/locations

```
{
  "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 HLR GT que envió el alertServiceCenter, lo que permite 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 de enrutamiento de mensajes infinitos cuando los mensajes provienen de redes SS7, permitiendo al mismo tiempo la entrega legítima de MO→MT entre suscriptores en la misma red.

Por qué es Importante la Prevención de Bucles

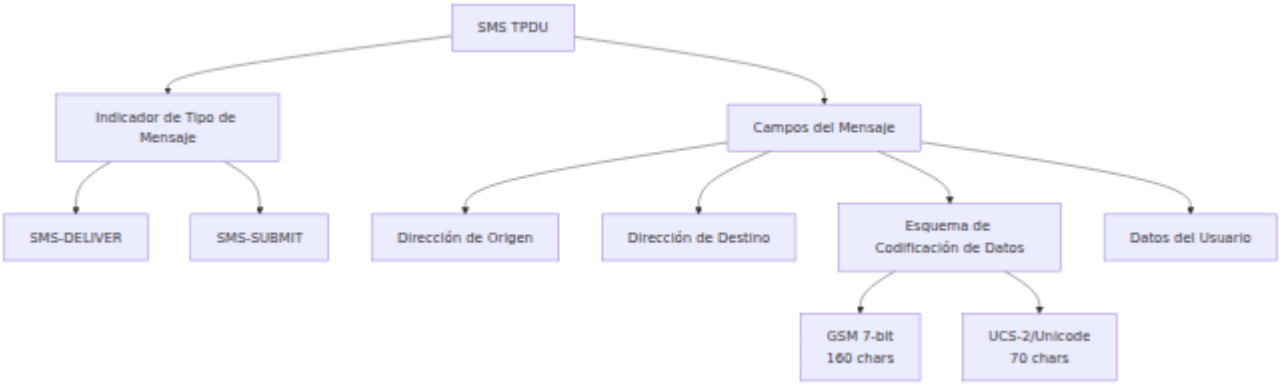
Cuando el SMSc recibe mensajes SMS de origen móvil (MO) de la red SS7, los inserta en la cola de mensajes con un campo `source_smsc` que identifica su origen (por ejemplo, `"SS7_MO_15551234567"`). Sin la prevención de bucles, estos mensajes podrían ser:

1. Recibidos de la red SS7 → Encolados con `source_smsc` que contiene "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 la red** y **fuera de la red**. El SMSc bloquea mensajes de SS7 solo cuando el suscriptor de destino **no está en la red**. Esto previene bucles de enrutamiento de tránsito. Aún permite la entrega legítima de MO→MT entre suscriptores en la misma red (por ejemplo, MSC a MSC a través del IP-SM-GW).



Detección en la Red

Cuando un mensaje proviene de SS7, el sistema verifica si el suscriptor de destino está en la red antes de aplicar la prevención de bucles:

1. **Caché VLR** (más rápido): Verificar si el suscriptor tiene una dirección VLR en caché con un 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 la red.

Si ninguna de las verificaciones encuentra al suscriptor, el destino se considera **fuera de la red** y el mensaje se bloquea.

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 la red**:

    - El mensaje se omite
    - Evento agregado: `"Prevención de Bucles"` con una descripción que indica que el destino está fuera de la red
    - Mensaje marcado como fallido a través de una solicitud PUT
    - Registrado con nivel de advertencia
- **Si `source_smsc` contiene "SS7" y el destino está en la red**:
    - El mensaje se procesa normalmente. Este es un flujo legítimo de MO→MT.
    - La entrega procede a través de la ruta de la API HSS / caché VLR
- **Si `source_smsc` no contiene "SS7"**:
    - El mensaje se procesa normalmente (por ejemplo, mensajes de la API web, SMPP u otras fuentes que no son SS7)

Valores de SMSC de Origen

Los mensajes pueden tener varios valores de `source_smsc`:

Fuente	Valor de Ejemplo	Acción
Red SS7 (MO-FSM), destino fuera de la red	`"SS7_MO_15551234567"`	**Omitido** - Prevención de bucles
Red SS7 (MO-FSM), destino en la red	`"SS7_MO_15551234567"`	**Permitido** - Entrega MO→MT en la red
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 la
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 se puede desactivar. Esta es una característica crítica de seguridad para prevenir la interrupción de la red por bucles de mensajes. El bypass en la red requiere que tanto `smsc_mode_enabled: true` como la caché VLR o `hlr_api_base_url` estén configurados.

Escenarios de Ejemplo

Escenario 1: MO-SMS de SS7 a suscriptor fuera de la red (bloqueado)

```
1. Teléfono móvil → MSC/VLR → IP-SM-GW (a través de MO-ForwardSM)
2. IP-SM-GW recibe MO-FSM, inserta en la cola: source_smsc =
"SS7_MO_9991900"
3. Auto-flush recupera mensaje de la cola
4. Destino no en caché VLR o HSS → fuera de la red
5. IP-SM-GW detecta "SS7" en source_smsc + destino fuera de la red
→ OMITIR
6. Evento registrado: "Prevención de Bucles"
7. No se envió SRI-for-SM ni MT-ForwardSM (bucle prevenido)
```

Escenario 2: MO-SMS de SS7 a suscriptor en la red (permitido)

```
1. Teléfono móvil → MSC/VLR → IP-SM-GW (a través de MO-ForwardSM)
2. IP-SM-GW recibe MO-FSM, inserta en la cola: source_smsc =
"SS7_MO_9991900"
3. Auto-flush recupera mensaje de la cola
4. Destino encontrado en API HSS → suscriptor en la 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
```

Rastreo de Suscriptores SMSc

El SMSc incluye un **Rastreador de Suscriptores** GenServer que mantiene el estado en tiempo real de los suscriptores basado en mensajes de alertServiceCenter e 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

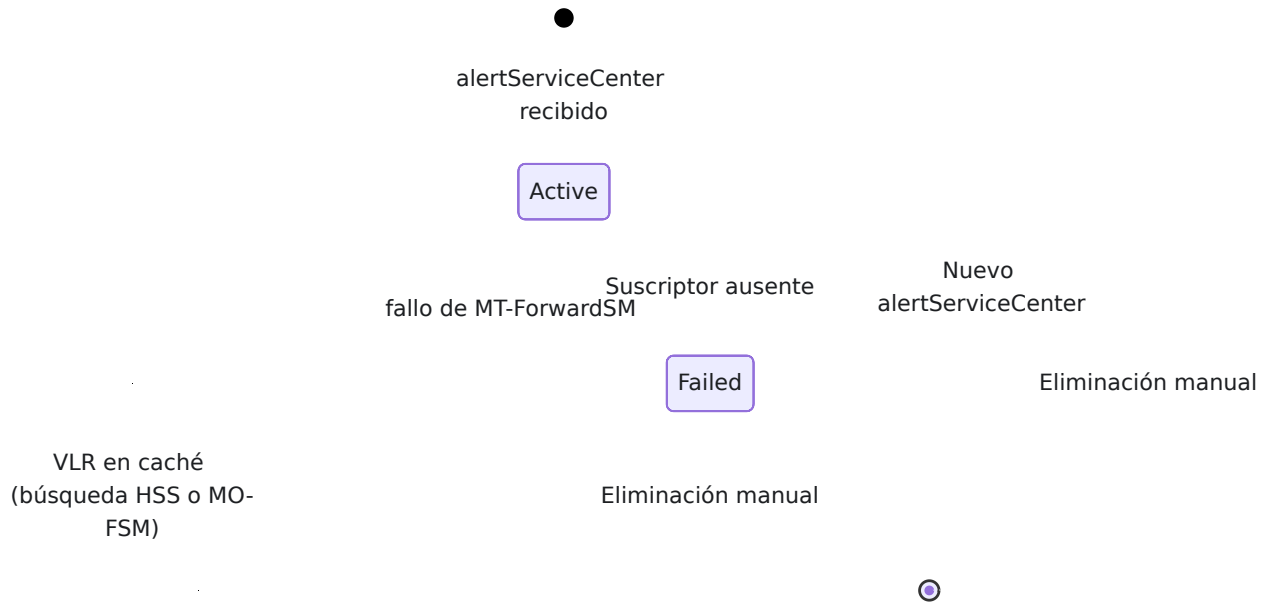
- **Contadores de mensajes**: Número de mensajes enviados/recibidos por suscriptor
- **Rastreo de fallos**: Marcar a los suscriptores como fallidos cuando los intentos de entrega fallan
- **Visibilidad en la Interfaz Web**: Panel en tiempo real que muestra todos los suscriptores rastreados

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`	HLR GT 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 GT 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 HLR GT
- Resolver IMSI real de la API HSS (si está configurada), de lo contrario usar IMSI sintético
- Preservar la caché VLR existente y los contadores de mensajes

Cuando la API HSS devuelve un suscriptor con VLR:

- Caché 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:

- Caché 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)
- Crea una entrada en el rastreador si no existe

Cuando falla la entrega de MT-ForwardSM:

- Borrar 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 un suscriptor:

- Eliminar de la tabla ETS
- Ya no aparece en la Interfaz Web

Interfaz Web - Página de Suscriptores SMSc

Ruta: `/smsc_subscribers` **Actualización automática**: Cada 2 segundos

Nota: Esta página solo está disponible cuando se ejecuta en modo SMSc. Después de habilitar la configuración de SMSc en `config/config.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, HLR GT, VLR GT (dirección VLR 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 por SMSc                                      Tota
├──────────────────────────────────────────────────────────────────────────────
│ 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  |  Activos: 2  |  Fallidos: 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 (fallo de SRI-for-SM)
SMSc.SubscriberTracker.mark_failed(msisdn)

# Eliminar del rastreo
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 de alertServiceCenter**: Llama a `alert_received/3` en la actualización de ubicación exitosa
- **Manejador de SRI-for-SM**: Incrementa `messages_sent` en el enrutamiento exitoso
- **Manejador 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 Dirección VLR

Cuando opera 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 llena desde dos fuentes:

1. **Búsqueda de API HSS**: Cuando el pipeline de entrega consulta la API HSS y obtiene un suscriptor con un VLR en 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, se verifica primero la caché. 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 borra cuando:

- **Falla la entrega de MT-ForwardSM** (tiempo de espera, error SCCP, rechazo de VLR): la VLR en caché puede estar obsoleta
- **Expira TTL**: el TTL es configurable a través de `vlr_cache_ttl_seconds`. La caché trata las entradas más antiguas que el TTL como fallos de caché.

La caché **no** se borra en alertServiceCenter. La entrada VLR existente se preserva porque el suscriptor aún puede ser alcanzable en el mismo VLR.

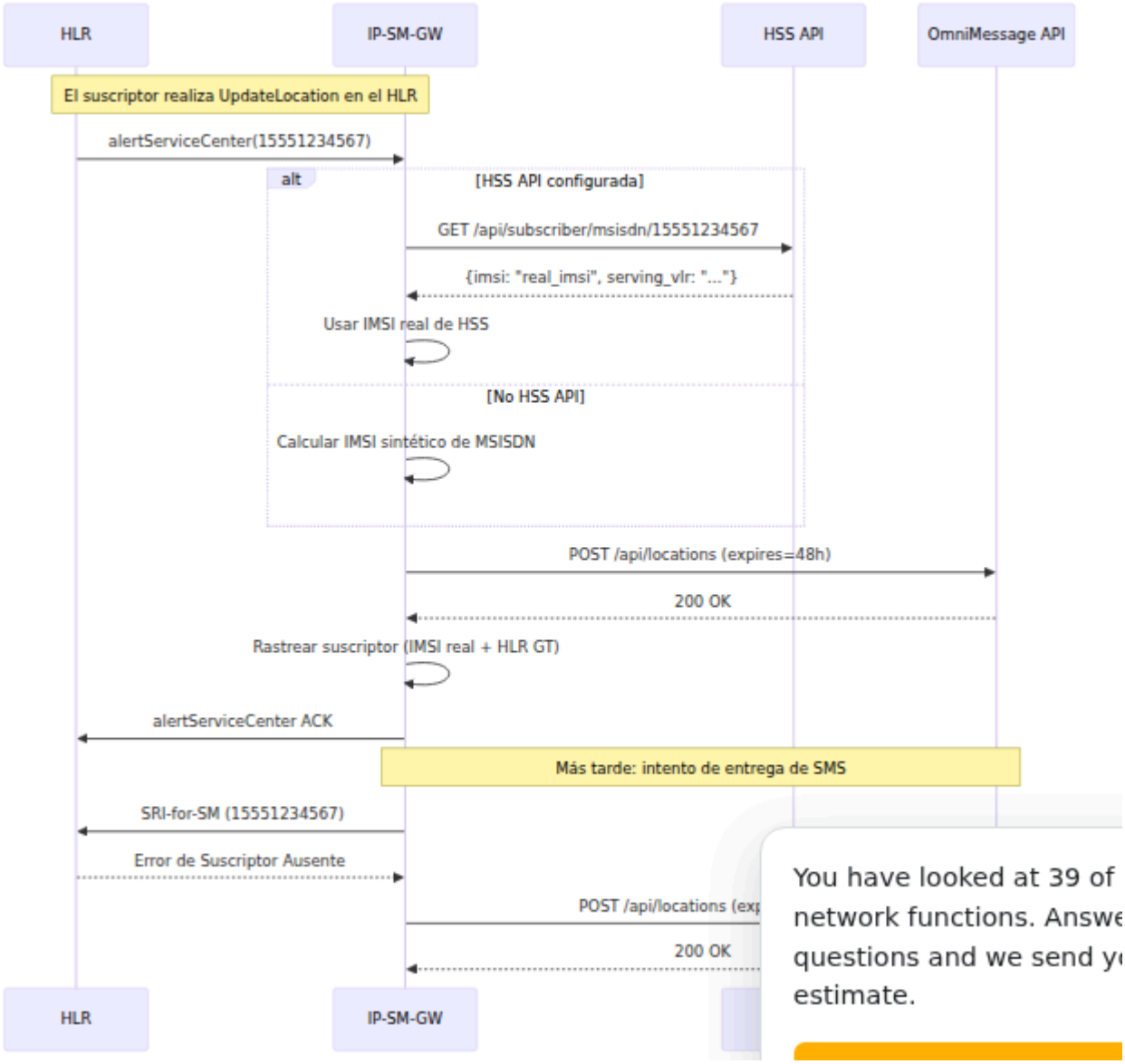
Configuración

```
config :omniss7,
  # Endpoint de API HSS/HLR (requerido para caché VLR y entrega en
la red)
  hlr_api_base_url: "https://omnihss.example.com:8443",
  # Clave de API OmniHSS (requerida cuando se establece
hlr_api_base_url)
  hlr_api_key: "<Clave API OmniHSS con el rol hlr_interworking>",

  # TTL de caché VLR en segundos (predeterminado: 3600 = 1 hora)
  vlr_cache_ttl_seconds: 3600
```

Parámetro	Tipo	Requerido	Predeterminado
`hlr_api_base_url`	String	No	`nil`
`hlr_api_key`	String	Cuando se establece `hlr_api_base_url`	`nil`

Parámetro	Tipo	Requerido	Predeterminado
`vlr_cache_ttl_seconds`	Integer	No	`3600`

Diagrama de Secuencia



Cola de SMS Auto-Flush

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 donde se debe enviar MT-ForwardSM
     - Consulte Detalles de SRI-for-SM en la Guía HLR para la documentación completa
   - En caso de éxito, enviar **MT-ForwardSM** al MSC/VLR
   - Actualizar el estado del mensaje a través de la API (entregado/fallido)
   - Agregar 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 la dirección del centro de servicio GT y la generación de IMSI sintéticos que preservan la privacidad, consulte la sección SRI-for-SM en la Guía de Configuración de HLR.

Métricas de SMSc

OmniSS7 expone métricas de Prometheus. Las métricas MAP compartidas rastrean la actividad de SMS. No hay contadores de cola/entrega específicos de SMSc. Las métricas relevantes para las operaciones de SMS son:

Métrica	Tipo	Etiquetas	Relevar S
`map_requests_total`	Contador	`operation`	Cuenta l operacio Las oper relevant SMS incl `sri_for` `forward` `MTforwa` `prn`.
`map_request_duration_milliseconds`	Histograma	`operation`	Duración ciclos de solicitud, MAP, inc SRI-for-S ForwardS

El conjunto completo de métricas declaradas (no todas específicas de SMS) es: `map_requests_total`, `cap_requests_total`, `ussd_requests_total`, `ussd_active_sessions`, `map_pending_requests`, `omniss7_license_status`, y `map_request_duration_milliseconds`.

Consultas de Ejemplo:

```
# Tasa de operaciones MT-ForwardSM en los últimos 5 minutos
rate(map_requests_total{operation="MTforward_sm"}[5m])

# Tasa de operaciones SRI-for-SM en los últimos 5 minutos
rate(map_requests_total{operation="sri_for_sm"}[5m])

# Duración promedio de MT-ForwardSM (últimos 5 minutos)
rate(map_request_duration_milliseconds_sum{operation="MTforward_sm"}
[5m]) /
rate(map_request_duration_milliseconds_count{operation="MTforward_sm"
[5m])
```

Solución de Problemas de SMSc

Problema: Mensajes No Entregados

Verificaciones:

1. Verifique que el auto-flush esté habilitado
2. Verifique la conexión a la base de datos
3. Monitoree los registros en busca de errores
4. Verifique que la conexión M3UA esté ACTIVA
5. Verifique los límites de TPS

Problema: Profundidad de Cola Alta

Causas Posibles:

- Límite de 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 la disponibilidad de HLR
- Revisar los registros de mensajes fallidos

API de MT-forwardSM

Enviar SMS a través de la API

Endpoint de API: `POST /api/MT-forwardSM`

Los cuatro campos son obligatorios. Los nombres de los campos están en snake_case (note que `destination_service_centre` utiliza la ortografía británica). `smsPDU` es el SMS TPDU codificado en hex.

Solicitud:

```
{
  "imsi": "234509876543210",
  "destination_service_centre": "447999555111",
  "originating_service_center": "447999123456",
  "smsPDU":
"040B917477218345F60000157030185714 0C0BD4F29C0E9281C4E1F11A"
}
```

Respuesta:

En caso de éxito (HTTP 200), la respuesta contiene una única clave `result` que contiene el resultado TCAP formateado del diálogo MT-ForwardSM (no hay `message_id`, y el valor no es la cadena literal `"success"`):

```
{
  "result": "<formatted TCAP result>"
}
```

Si el diálogo MAP se agota, el endpoint devuelve HTTP 504 con `{"error": "timeout"}`. Una solicitud con campos mal formados/faltantes devuelve HTTP 400 con `{"error": "invalid request"}`.

Documentación Relacionada

Documentación de OmniSS7:

- ← Volver a la Documentación Principal
- Guía de Configuración de HLR - Configuración y operaciones en modo HLR
  - **Detalles Técnicos de SRI-for-SM** - Documentación completa sobre el mapeo de MSISDN a IMSI y la 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 entregas, limitación de tasas y análisis, consulte la **documentación del producto OmniMessage**. OmniMessage contiene toda la lógica de enrutamiento de mensajes, algoritmos de reintento de colas, manejo de informes de entrega y motor de reglas de negocio.

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 Señales (STP)**.

Tabla de Contenidos

1. ¿Qué es un STP?
2. Roles de Red STP
3. Interfaz con Redes TDM
4. Habilitando el Modo STP
5. Configurando Pares
6. Soporte del Protocolo M2PA
    - M3UA vs M2PA
    - Configurando Pares M2PA
    - Configuración de SLTM
    - Gestión de M2PA a través de la Interfaz Web
    - Métricas de M2PA
7. Enrutamiento de Código de Punto
8. Enrutamiento de Título Global
9. Manejo de Tránsito SCCP
    - Ruta en SSN Después de GTT
    - Aplicación de Contador de Saltos
    - Devolución de Mensaje en Caso de Fallo
10. Estado de Destino (Gestión de Ruta de Señalización)
11. Filtrado de Puerta (Cortafuegos SCCP)
12. Características de Gestión de Rutas
    - Deshabilitando Rutas

    - Rutas DROP - Previniendo Bucles de Enrutamiento
13. Enrutamiento Avanzado
14. Pruebas de Configuración
15. Métricas y Monitoreo
16. Monitoreo de Pares M3UA
    - Manejo de Contexto de Enrutamiento M3UA
    - Desambiguación de Múltiples Pares con la Misma IP

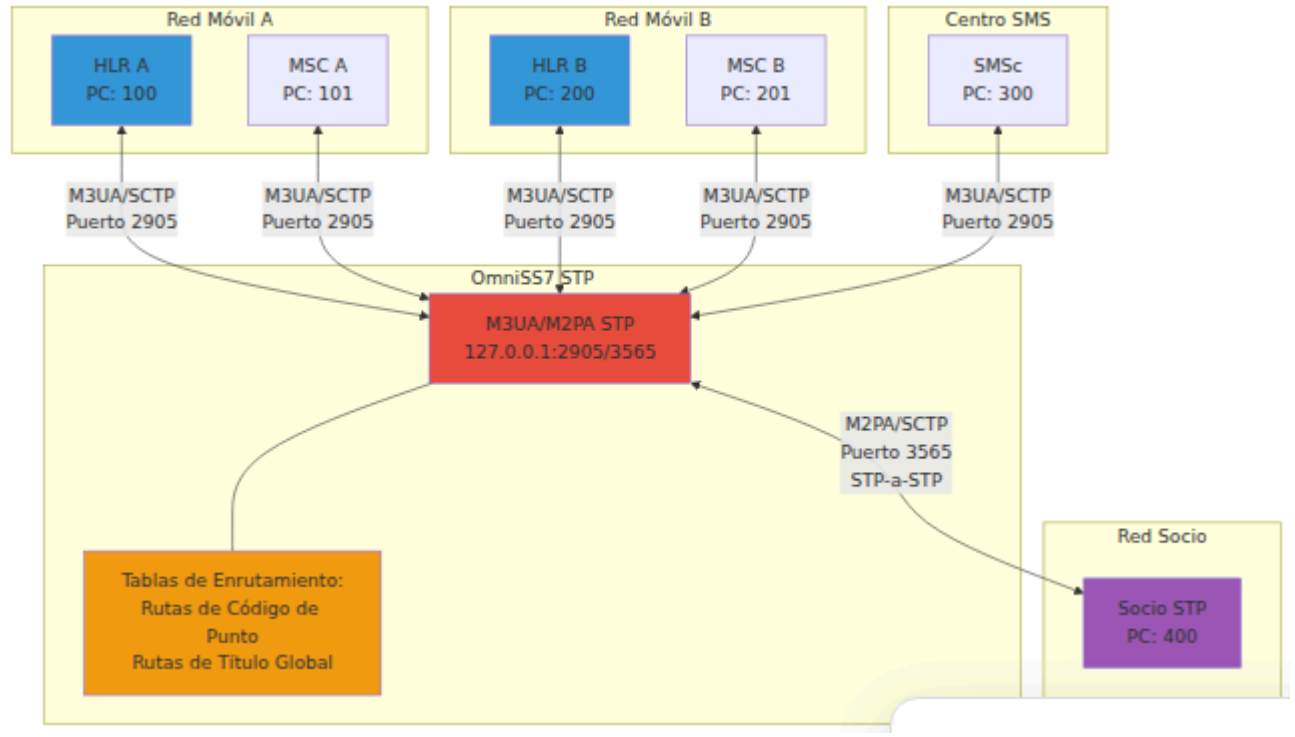
¿Qué es un Punto de Transferencia de Señales (STP)?

Un **Punto de Transferencia de Señales (STP)** es un elemento crítico de la red en redes de señalización SS7 e IP que enruta mensajes de señalización entre nodos de la red.

Funciones del STP

- **Enrutamiento de Mensajes**: Enruta el tráfico de señalización SS7 basado en el Código de Punto de Destino (PC) 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 prioridad
- **Puerta de Red**: Conecta diferentes redes de señalización y proveedores de servicios
- **Ocultación de Topología**: Puede reescribir direcciones para ocultar la topología interna de la red

Diagrama de Red STP



Roles de Red STP Explicados

ASP (Proceso del Servidor de Aplicaciones)

- **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 socio

SGP (Proceso de Puerta de Señales)

- **Rol**: Servidor que acepta conexiones de ASPs
- **Dirección**: Conexión entrante
- **Caso de Uso**: Redes socio se conectan a su STP

AS (Servidor de Aplicaciones)

- **Definición**: Agrupación lógica de uno o más ASPs
- **Propósito**: Proporciona redundancia y distribución de carga
- **Caso de Uso**: Múltiples ASPs sirviendo el mismo destino

Interfaz con Redes TDM v��a Puertas de Señales

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 Señales (SGW)** para unir los dos mundos.

¿Qué es una Puerta de Señales?

Una **Puerta de Señales (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. Permite 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 Señales	IP (SIGTRAN)
Parte de Usuario	SCCP/TCAP/MAP	← Transparente →	SCCP/TCAP/MAP
Red	MTP3	Conversión	M3UA/M2PA
Enlace	MTP2	Terminación	SCTP
Físico	MTP1 (E1/T1)	Terminación	IP/Ethernet

El SGW termina MTP1/MTP2 en el lado TDM y presenta los datos de usuario MTP3 sobre M3UA o M2PA en el lado IP. Las capas superiores (SCCP, TCAP, MAP, CAP) pasan a través de manera transparente.

Conectando OmniSS7 a una Puerta de Señales

OmniSS7 se conecta a una Puerta de Señales como un **ASP (Proceso del Servidor de Aplicaciones)**, mientras que el SGW actúa como un **SGP (Proceso de Puerta de Señales)**.

Ejemplo de Configuración

```
config :omniss7,
  # Conectar a la Puerta de Señales 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 Señales
    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 el 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
    }
  ],

  # Enrutar códigos de punto de la red TDM a través de la puerta
  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 Señales 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 la red TDM accesibles a través del SGW	Específico de la red
Modo de Tráfico	Modo de manejo de tráfico ASP	Sobrescribir o Compartir carga

Escenarios de Implementación

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:

```
IP SMSC → OmniSS7 STP → SGW → TDM HLR
              ↓
         HLR basado en IP
```

Escenario 3: Doble SGW para Redundancia

Conéctese a dos Puertas de Señales 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 STP M3UA

OmniSS7 puede operar en diferentes modos. Para usarlo como un STP, debe proporcionar la configuración de enrutamiento STP bajo la clave de aplicación `:omniss7`.

Dónde Vive la Configuración

Archivo	Propósito
`config/config.exs`	Configuración predeterminada, siempre cargada (interfaz web, API escuchando en el puerto 8445, registro)
`config/stp_example_runtime.exs`	Un **ejemplo de STP** completo y listo para editar (`sctp_handler`, pares, rutas de PC, rutas de GT)
`config/runtime.exs`	Sobrescrituras de tiempo de ejecución por implementación. En este repositorio es un stub mínimo; los servidores de producción despliegan el suyo propio

> **Nota**: Las versiones anteriores incluían tres bloques de modo comentados dentro de `config/runtime.exs`. Ese diseño ya no existe. El ejemplo de STP ahora vive en su propio archivo, `config/stp_example_runtime.exs`.

Cambiando al Modo STP

Para establecer un STP, copie la configuración de ejemplo en la configuración que su implementación carga en tiempo de ejecución:

1. **Abra** `config/stp_example_runtime.exs` y revise el ejemplo de STP (manejador SCTP, pares, rutas de PC, rutas de GT).
2. **Copie** el bloque `config :omniss7, ...` en el `config/runtime.exs` de su implementación (o fusione las claves relevantes).
3. **Personalice** las direcciones IP, puertos, códigos de punto, pares y rutas para su red.
4. **Reinicie** la aplicación: `iex -S mix`

Configuración del Modo STP

El ejemplo de STP en `config/stp_example_runtime.exs` configura un oyente STP entrante (`sctp_handler`), habilita el enrutamiento GT y define pares más rutas de PC y rutas de GT:

```
config :omniss7,
  # Oyente SCTP (servidor STP entrante) - ver Modo STP Autónomo a
continuación
  sctp_handler: %{
    enabled: true,
    local_ip: {127, 0, 0, 1},
    local_port: 2905,
    point_code: 101            # Código de punto de este STP
  },

  # Habilitar enrutamiento de Título Global (además del
enrutamiento de PC)
  enable_gt_routing: true,

  # Los pares M3UA, rutas de PC y rutas de GT siguen
  m3ua_peers: [ %{peer_id: 1, name: "STP_West", role: :client,
...} ],
  m3ua_routes: [ %{dest_pc: 100, peer_id: 1, priority: 1,
network_indicator: :international} ],
  m3ua_gt_routes: [ %{gt_prefix: "44", peer_id: 2, priority: 1,
description: "Números del Reino Unido"} ]
```

Vea Configurando Pares M3UA, Enrutamiento de Código de Punto, y Enrutamiento de Título Global a continuación para el conjunto completo de parámetros.

Parámetros de Configuración a Personalizar

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

Estas son las claves STP de nivel superior bajo `config :omniss7`:

Parámetro	Tipo	Predeterminado	Descripción	Eje
`sctp_handler`	Mapa	*Opcional*	Configuración del oyente STP entrante (ver Modo STP Autónomo)	`%{en` `true` `loca` `2905` `poin` `101}`
`enable_gt_routing`	Booleano	`false`	Habilitar enrutamiento de Título Global (además del enrutamiento de PC)	`true`
`m3ua_peers`	Lista	`[]`	Definiciones de pares M3UA / M2PA (ver Configurando Pares M3UA)	Ning
`m3ua_routes`	Lista	`[]`	Rutas de Código de Punto (ver Enrutamiento de Código de Punto)	Ning
`m3ua_gt_routes`	Lista	`[]`	Rutas de Título Global (ver Enrutamiento de Título Global)	Ning

> **Consejo**: Use la 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

Una vez que se configuran los pares y las rutas, la interfaz web muestra:

- **Eventos SS7** - Registro de eventos
- **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

Notas Importantes

- El protocolo SCTP (protocolo IP 132) debe ser permitido a través de los cortafuegos
- El puerto M3UA predeterminado es 2905 (estándar de la industria)
- Asegúrese de tener 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 la configuración, 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 la API REST y acceso a Swagger UI

Modo STP Autónomo

Puede ejecutar un **servidor STP M3UA autónomo** que escuche conexiones entrantes. Este es el modo utilizado por el ejemplo en `config/stp_example_runtime.exs`.

Habilitando STP Autónomo

Agregue esta configuración bajo `config :omniss7` en su configuración de tiempo de ejecución:

```
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 de este STP
  }
```

Parámetros de Configuración del STP

Parámetro	Tipo	Predeterminado	Descripción	Ejemplo
`enabled`	Booleano	`false`	Habilitar servidor STP autónomo	`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 de este STP	`100`

Cuándo Usar STP Autónomo

- **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 de 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 la 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 legado `m3ua_peers`), `m3ua_routes`, y `m3ua_gt_routes` se cargan al inicio de la aplicación
2. **Rutas de Interfaz Web**: Las rutas añadidas 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 de 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 por disco (predeterminado)	**Sobrevive a los reinicios**	Entornos de producción
`:ram_copies`	Solo en memoria	Perdido al reiniciar	Pruebas, desarrollo

Predeterminado: `:disc_copies`

Ubicación de la Base de Datos Mnesia

Mnesia almacena las tablas de enrutamiento en el directorio Mnesia de la aplicación:

- **Ubicación**: `Mnesia.{node_name}/` (por ejemplo, `Mnesia.nonode@nohost/`)
- **Tablas** (todas creadas en `M3UARouting.init_database`):

Tabla	Propósito
`m3ua_peer`	Puntos finales de conexión M3UA / M2PA
`m3ua_route`	Rutas de Código de Punto (DPC → par, con soporte de máscara)
`m3ua_gt_route`	Rutas de Título Global (prefijo → par, con coincidencias y reescrituras de SSN/TT/NPI/NAI)
`m3ua_dest_state`	Estado de disponibilidad por destino impulsado por la gestión de ruta de señalización MTP3 (TFP/TFA/TFR, Q.704 §13)
`ss7_screen_rule`	Reglas de permiso/denegación de filtrado de puerta (cortafuegos SCCP)

Gestión de Rutas

Tiene tres opciones para gestionar rutas:

1. **Runtime.exs** - Configuración estática cargada al inicio
2. **Interfaz Web** - Gestión interactiva de rutas (ver Guía de Interfaz Web)
3. **API REST** - Gestión programática de rutas (ver Guía API)

Mejor Práctica: Use `runtime.exs` para la configuración base y la Interfaz Web para cambios dinámicos de ruta 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` todavía se admite por compatibilidad hacia atrás.

```
config :omniss7,
  peers: [
    # Conexión saliente a STP Socio (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 de
puerto dinámica
      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, STP espera conexión entrante
    %{
      peer_id: 3,
      name: "Remote_MSC",
      role: :server,                        # Aceptar conexión
entrante
```

```
      remote_ip: {10, 0, 0, 30},           # IP de origen esperada
      remote_port: 2905,                    # Puerto de origen
esperado (0 = aceptar de cualquier puerto)
      routing_context: 3,
      point_code: 300,
      network_indicator: :international
    },

    # Conexión entrante con puerto de origen dinámico (sin
filtrado de puerto)
    %{
      peer_id: 4,
      name: "Dynamic_Client",
      role: :server,
      remote_ip: {10, 0, 0, 40},           # IP de origen esperada
      remote_port: 0,                       # 0 = aceptar
conexiones de cualquier puerto de origen
      routing_context: 4,
      point_code: 400,
      network_indicator: :international
    }
  ]
```

Parámetros de configuración del par

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) a 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 o lista única 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 cualquier puerto de origen)
`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 necesitas 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, puedes configurar múltiples direcciones IP tanto para `local_ip` como para `remote_ip` utilizando listas. Esto permite un failover automático si un camino de red falla. Consulta la Guía de Multihoming SCTP para ejemplos de configuración detallados y mejores prácticas.

Filtrado de puerto de origen para conexiones entrantes

Para **conexiones entrantes** (rol: `:server`), el parámetro `remote_port` controla el filtrado de puerto de origen:

- **Puerto específico** (por ejemplo, `remote_port: 2905`): Aceptar solo conexiones de ese puerto de origen exacto
  - Proporciona seguridad adicional al validar el puerto de origen
  - Usar cuando el par remoto utiliza un puerto de origen fijo
- **Cualquier puerto** (`remote_port: 0`): Aceptar conexiones de cualquier puerto de origen

- Útil cuando el par remoto utiliza puertos de origen dinámicos/efímeros
- Solo valida la dirección IP de origen
- Más flexible pero ligeramente menos seguro

Ejemplo:

```
# Aceptar solo de 192.0.2.10:2905 (puerto específico)
%{
  peer_id: 1,
  name: "Strict_Peer",
  role: :server,
  remote_ip: {10, 5, 198, 200},
  remote_port: 2905,
  # ... otra configuración
}

# Aceptar de 192.0.2.10 con cualquier puerto de origen
%{
  peer_id: 2,
  name: "Flexible_Peer",
  role: :server,
  remote_ip: {10, 5, 198, 200},
  remote_port: 0,  # Aceptar de cualquier puerto de origen
  # ... otra configuración
}
```

Soporte del protocolo M2PA

OmniSS7 admite tanto los protocolos **M3UA** como **M2PA** para el transporte de señalización SS7.

¿Qué es M2PA?

M2PA (Capa de Adaptación de Usuario MTP2 Peer-to-Peer) es un protocolo estandarizado por IETF (RFC 4165) para transportar mensajes MTP3 de SS7 sobre redes IP utilizando SCTP.

M3UA vs M2PA: Principales diferencias

Característica	M3UA	M2PA
Arquitectura	Cliente/Servidor (ASP/SGW)	Peer-to-Peer
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	24 bits BSN/FSN 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 protocolos

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**: Puenteando redes SS7 con señalización basada en IP
- **Conexiones de elementos de red**: Conectando HLRs, MSCs, SMSCs y otros elementos de red a tu STP

- **Puerta de enlace de señalización (SGW)**: Puerta de enlace central que acepta conexiones de múltiples servidores de aplicaciones
- **Topologías flexibles**: Arquitecturas cliente/servidor con control centralizado
- **Redes de múltiples proveedores**: Estándar de la industria ampliamente soportado (RFC 4666)

Usa M3UA para conectar elementos de red (HLR, MSC, SMSC, VLR, etc.) a tu STP.

Cuándo usar M2PA (Solo en 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 heredado**: Reemplazando enlaces TDM SS7 tradicionales cuando el sistema remoto requiere específicamente M2PA
- **Compatibilidad con MTP2 requerida**: Al conectarse a sistemas heredados que exigen gestión del estado del enlace al estilo MTP2
- **Requisito del socio**: Cuando un socio o interconexión requiere específicamente el protocolo M2PA

Importante: No uses M2PA para conectar elementos de red (HLR, MSC, SMSC) a tu STP - usa M3UA en su lugar. M2PA está diseñado para interconexiones STP-a-STP donde ambos lados operan como nodos de enrutamiento.

Configuración de pares M2PA

Los pares M2PA se configuran de la misma manera que los pares M3UA, con un parámetro adicional `protocol`.

Configuración de pares M2PA

Agrega pares M2PA a tu configuración de `peers` en `config/runtime.exs` (tanto los pares M3UA como M2PA comparten la misma sección de configuración, diferenciada por el parámetro `protocol`):

Parámetros clave para M2PA:

Parámetro	Valor	Descripción
`protocol`	`:m2pa`	Especifica el protocolo M2PA (por defecto es `: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	Tu código de punto
`adjacent_point_code`	Entero	Código de punto del par remoto (específico de M2PA)
`send_sltm`	Booleano	Controla el comportamiento de SLTM (ver Configuración de SLTM a continuación)
`network_indicator`	Átomo	`:international` o `:national` - debe coincidir con el par remoto

Nota: M2PA utiliza **puerto 3565** como estándar de la industria (diferente del puerto 2905 de M3UA).

Configuración de SLTM

SLTM (Mensaje de prueba de enlace de señalización) y **SLTA (Reconocimiento de prueba de enlace de señalización)** son mensajes de mantenimiento de 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 de 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 configurado (por defecto)	Sigue Q.707: el respondedor SCTP envía SLTM, el iniciador SCTP espera

Comportamiento por defecto (Cumple con Q.707):

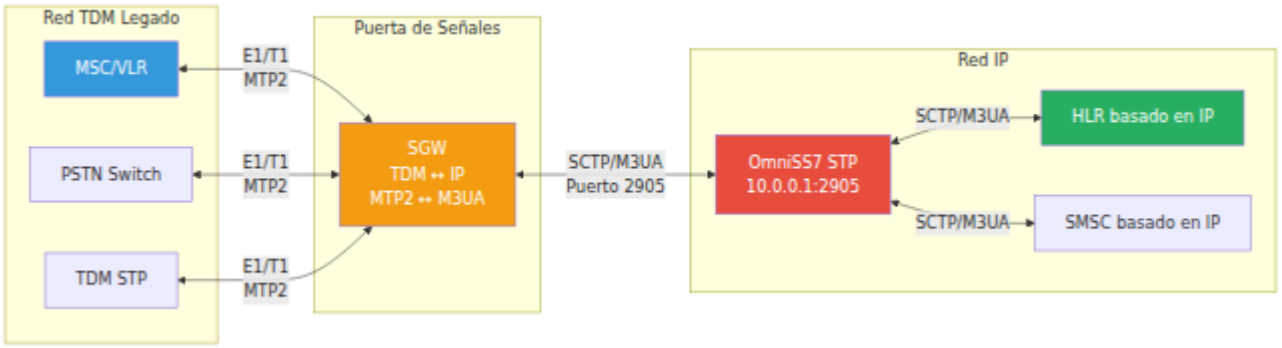
- Si `initiate_connection: true` (iniciador/client SCTP) → Esperamos que el par envíe SLTM
- Si `initiate_connection: false` (respondedor/server SCTP) → Enviamos SLTM

Cuándo anular los valores predeterminados de SLTM

Anula el comportamiento predeterminado al conectarte 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  # Anular: Enviamos SLTM aunque seamos el
iniciador SCTP
}
```

Flujo de mensajes SLTM



Solución de problemas de problemas de SLTM

Síntoma: El enlace alcanza LISTO pero no fluye tráfico

Ambos lados pueden estar esperando que el otro envíe SLTM. Verifica los registros para:

```
"esperando que el par envíe SLTM"
```

Resolución: Configura `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 del indicador de red (`:international` vs `:national`)
3. El par remoto no está configurado para responder a nuestro código de punto

Resolución: Verifica que `adjacent_point_code` coincida con el código de punto del STP (configurado en `sctp_handler.point_code`).

Estados del enlace M2PA

Los enlaces M2PA progresan a través de varios estados durante la inicialización:

1. **Abajo** - No se ha establecido 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. **Navega** a la página de Enrutamiento
2. **Selecciona** la pestaña "Pares"
3. **Haz clic** en "Agregar nuevo par"
4. **Elige** "M2PA (RFC 4165)" del menú desplegable de Protocolo
5. **Completa** la configuración del par:
   - Nombre del par (identificador descriptivo)
   - Protocolo: M2PA

    - Rol: cliente o servidor
    - Código de punto (tu PC)
    - Direcciones IP locales/remotas
    - Puertos locales/remotos (típicamente 3565 para M2PA)
    - Indicador de red (internacional o nacional)
6. **Haz clic** en "Guardar par"

La tabla de pares muestra el tipo de protocolo con codificación de colores:

- **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 de manera idéntica para M2PA y M3UA
- **Rutas de título global**: Totalmente soportadas en enlaces M2PA
- **Prioridad de ruta**: Los pares M2PA y M3UA se pueden mezclar en las mismas tablas de enrutamiento
- **Reenvío de mensajes**: Los mensajes pueden llegar a M2PA y ser enrutados a M3UA, y viceversa

Métricas de 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 (ABAJO → ALINEACIÓN → PRUEBA → LISTO)
  - Etiquetas: `link_name`, `from_state`, `to_state`

Métricas de error:

- `m2pa_errors_total` - Total de errores por tipo
  - `decode_error` - Fallos en la decodificación de mensajes M2PA
  - `encode_error` - Fallos en la codificación de mensajes M2PA
  - `sctp_send_error` - Fallos en la transmisión SCTP
  - Etiquetas: `link_name`, `error_type`

Métricas de acceso:

- Punto final de Prometheus: `http://your-server:9090/metrics`
- Las métricas se registran automáticamente al iniciar la aplicación

Mejores prácticas de M2PA

1. **Selección de puerto**: Usa el puerto 3565 para M2PA (estándar de la industria)
2. **Monitoreo de enlaces**: Monitorea los cambios de estado del enlace a través de métricas
3. **Reglas de firewall**: Asegúrate de que SCTP (protocolo IP 132) esté permitido
4. **Códigos de punto**: Asegúrate 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**: Usa la página de prueba de enrutamiento para verificar la conectividad después de la configuración

Requisitos de socket M2PA

M2PA utiliza sockets compartidos SCTP.SocketHandler. Todos los pares M2PA utilizan automáticamente el SCTP.SocketHandler para la gestión de sockets, lo que permite a múltiples pares compartir eficientemente el mismo puerto SCTP.

Requisitos:

- SCTP.SocketHandler debe estar habilitado y en funcionamiento
- El socket compartido debe configurarse antes de que comiencen los pares M2PA

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)

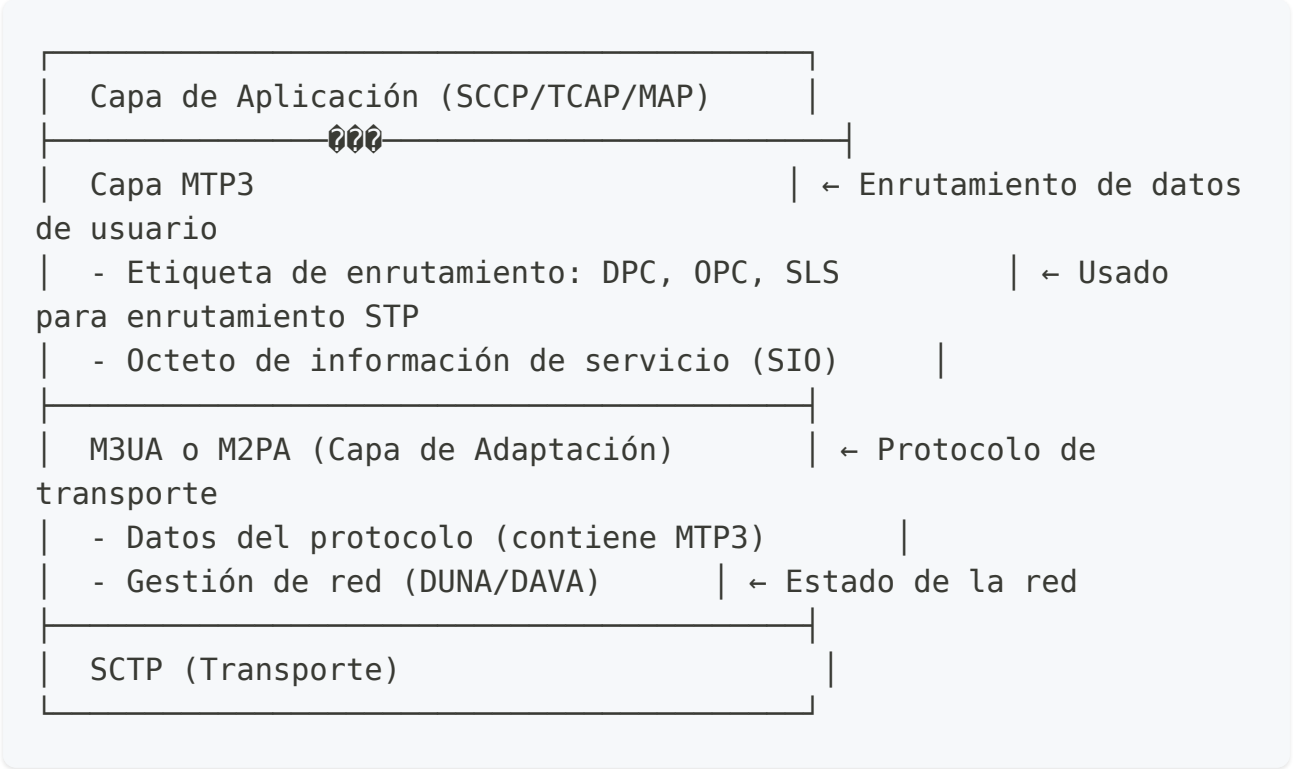
Configuración del enrutamiento de códigos de punto

El enrutamiento de códigos de punto dirige los mensajes en función del **Código de Punto de Destino (DPC)** en el encabezado MTP3.

Entendiendo los códigos de punto en la pila de protocolos SS7

Los códigos de punto existen en diferentes capas de la pila de protocolos SS7. Entender esta distinción es importante:

Capas de la pila de protocolos:



Dos tipos de códigos de punto:

1. **Códigos de punto de la capa MTP3** (Usados para enrutamiento):
    - Ubicados en la etiqueta de enrutamiento MTP3 (DPC, OPC)
    - Presentes en el parámetro de datos del protocolo M3UA (etiqueta 528)
    - Presentes en los mensajes de datos de usuario M2PA

    - **STP utiliza estos valores DPC para decisiones de enrutamiento**
    - Estos determinan dónde se entrega finalmente el mensaje

2. **Códigos de punto de la capa M3UA** (Usados para gestión de red):
    - Presentes en mensajes de gestión M3UA (DUNA, DAVA, SCON, DUPU)
    - Indican los códigos de punto afectados para el estado de la red
    - Indican 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**: STP extrae el mensaje MTP3 del parámetro de datos del 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**: 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 señalizar el estado de la red entre pares.

Rutas básicas de códigos de punto

Agrega rutas a `config/runtime.exs`:

```
config :omniss7,
  m3ua_routes: [
    # Rutar todo el tráfico para PC 100 al par 1 (STP Socio)
    %{
      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: por defecto es
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 por defecto es `14` (coincidencia exacta de código de punto). Solo especifica `mask` cuando necesites enrutamiento basado

en rangos (ver sección de Máscaras de Códigos de Punto a continuación).

Lógica de enrutamiento

1. STP recibe un mensaje M3UA DATA o un mensaje de datos de usuario M2PA
2. STP extrae el **mensaje MTP3** del campo de datos del protocolo (M3UA) o del campo de datos de usuario (M2PA)
3. 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 las máscaras)
5. Si existen múltiples rutas, selecciona la ruta con la **más específica máscara** (valor de máscara más alto), luego **número de prioridad más bajo**
6. Envuelve el mensaje MTP3 en M3UA DATA o en 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 códigos 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, puedes usar **máscaras de códigos de punto** para crear rutas que coincidan con **rangos** de códigos de punto.

Entendiendo las 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 tener 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 (por defecto)
`/13`	2 PCs	Pequeño rango
`/12`	4 PCs	Pequeño rango
`/11`	8 PCs	Pequeño rango
`/10`	16 PCs	Rango mediano
`/9`	32 PCs	Rango mediano
`/8`	64 PCs	Rango mediano
`/7`	128 PCs	Rango mediano-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 por defecto/respaldo)

Ejemplos de máscaras de códigos de punto

Nota: El campo `mask` es **opcional** en todos los ejemplos. Si se omite, por defecto es `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 predeterminada es 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 mediano

```
%{
  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 por defecto/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 respaldo/captura con baja prioridad
```

Combinando rutas específicas y enmascaradas

Puedes combinar rutas específicas con rutas enmascaradas para un enrutamiento flexible:

```
config :omniss7,
  m3ua_routes: [
    # Ruta específica para PC 1000 (tiene prioridad)
    %{
      dest_pc: 1000,
      peer_id: 1,
      priority: 1,
      network_indicator: :international
      # la máscara predeterminada es 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 máscara `/14` ruta (PC 1000 exactamente) - **Seleccionada** (más específica)
2. También coincide con la máscara `/8` ruta (rango PC 1000-1063) - Ignorada (menos específica)
3. También coincide con la máscara `/0` ruta (todos los PCs) - Ignorada (menos específica)

Decisión de Enrutamiento para DPC 1015:

1. No coincide con la máscara `/14` ruta (solo PC 1000)
2. Coincide con la máscara `/8` ruta (rango PC 1000-1063) - **Seleccionada** (coincidencia más específica)
3. También coincide con la máscara `/0` ruta (todos los PCs) - Ignorada (menos específica)

Decisión de Enrutamiento para DPC 5000:

1. No coincide con la máscara `/14` ruta
2. No coincide con la máscara `/8` ruta
3. Coincide con la máscara `/0` ruta (todos los PCs) - **Seleccionada** (única coincidencia, ruta de respaldo)

Mejores Prácticas

1. **Omitir `mask` para Destinos Únicos**: Para coincidencias exactas de código de punto, omita el campo `mask` por completo (predeterminado 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**: Enrutar segmentos de red completos a pares específicos con máscaras `/0` a `/13`
4. **Usar `/0` como Respaldo**: Crear una ruta predeterminada con baja prioridad para capturar tráfico no coincidente
5. **La Más Específica Gana**: El motor de enrutamiento siempre selecciona primero la ruta coincidente más específica (valor de máscara más alto)
6. **Prioridad como Desempate**: Si múltiples rutas tienen la misma máscara, el número de prioridad más bajo gana

---

Configuración del Enrutamiento de Título Global (GT)

El enrutamiento de 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/parte llamada, consulte la Guía de NAT de Título Global.

Requisitos Previos

- Habilitar el enrutamiento GT: `enable_gt_routing: true` en `config/runtime.exs`

Configuración de Ruta GT

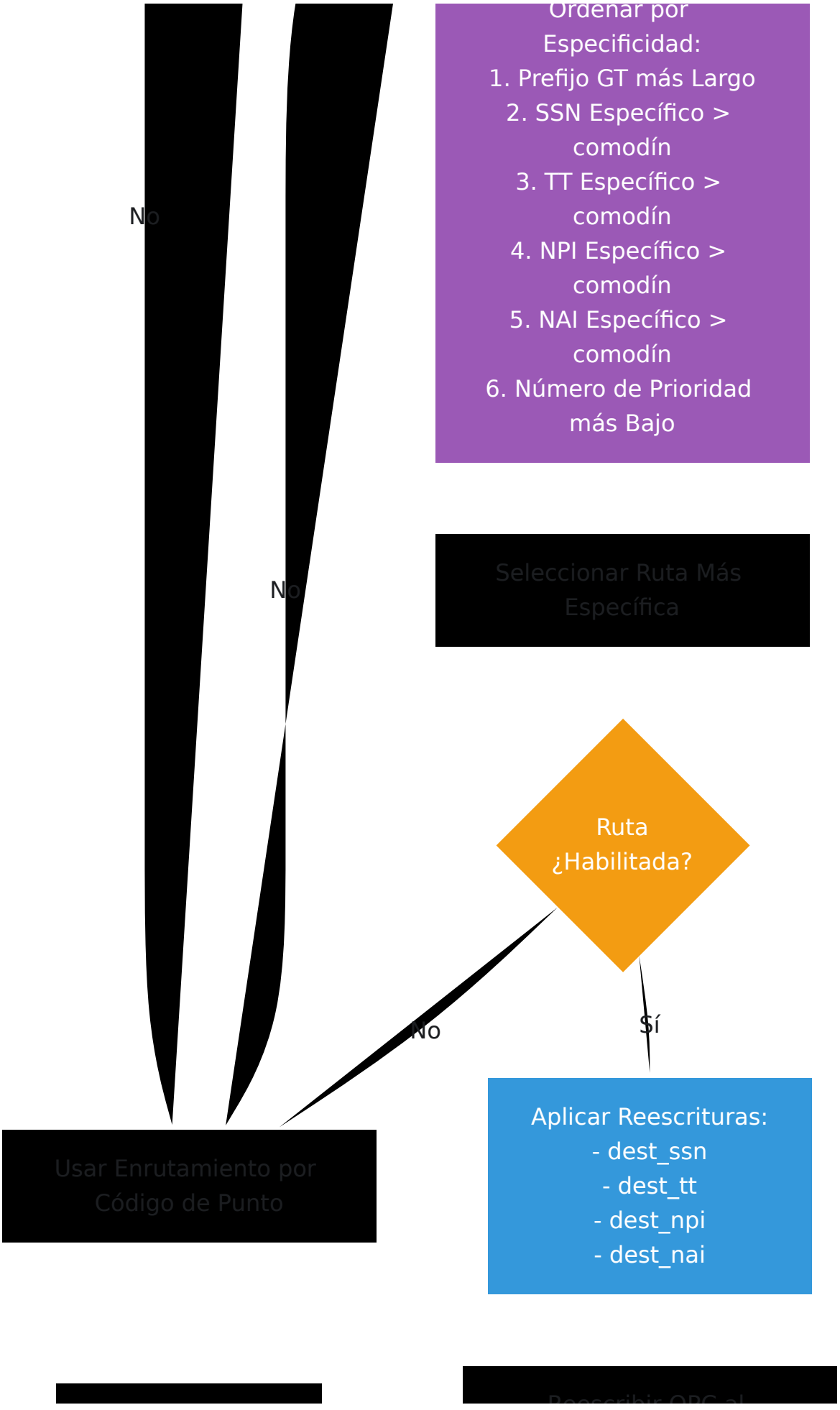
```
config :omniss7,
  # Habilitar el enrutamiento GT
  enable_gt_routing: true,

  m3ua_gt_routes: [
    # Ruta para 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
    },

    # Ruta para 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 de
prefijo 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,                  # Coincide solo si SSN de
origen = 8 (SMSC)
      peer_id: 4,
      dest_ssn: 6,                    # Reescribir SSN de destino
```

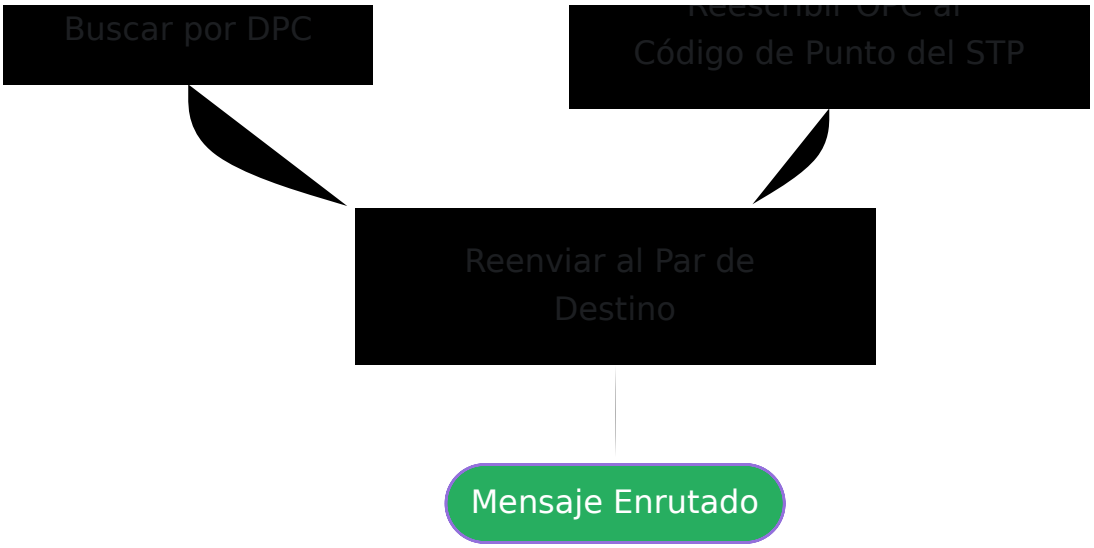
```
a 6 (HLR)
      priority: 1,
      description: "Tráfico SMS para prefijo 61"
    }
  ]
```

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. Número de Prioridad más Bajo
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



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 tanto con "44" como con "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 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 la normalización de protocolos y la interconexión de redes

6. **Reversa a Código de Punto**:
    - Si ninguna ruta GT coincide, el STP retrocede al enrutamiento por Código de Punto utilizando el DPC

Enrutamiento GT Avanzado: Tipo de Traducción, NPI y NAI

Además de la coincidencia de prefijo GT y SSN, el STP admite el enrutamiento y la transformación basados en los indicadores de Título Global SCCP:

- **Tipo de Traducción (TT)**: Identifica el plan de numeraci��n y el tipo de dirección
- **Indicador de Plan de Numeración (NPI)**: Define el plan de numeración (por ejemplo, ISDN, Datos, Telex)
- **Indicador de Naturaleza de la Dirección (NAI)**: Especifica el formato de dirección (por ejemplo, Internacional, Nacional, Suscriptor)

Coincidencia (Indicadores de Origen)

Las rutas pueden coincidir en los indicadores del mensaje entrante:

- `source_tt`: Coincidir mensajes con Tipo de Traducción específico

- `source_npi`: Coincidir mensajes con Indicador de Plan de Numeración específico
- `source_nai`: Coincidir mensajes con Indicador de Naturaleza de Dirección específico
- Valor `nil` = comodín (coincide con cualquier valor)

Transformación (Indicadores de Destino)

Las rutas pueden reescribir indicadores al reenviar:

- `dest_tt`: Transformar Tipo de Traducción a nuevo valor
- `dest_npi`: Transformar Indicador de Plan de Numeración a nuevo valor
- `dest_nai`: Transformar Indicador de Naturaleza de Dirección a nuevo valor
- Valor `nil` = preservar el valor original (sin transformación)

Selección Basada en Especificidad

Cuando múltiples rutas coinciden, se selecciona la ruta más específica utilizando este orden de prioridad:

1. Coincidencia de prefijo GT más largo
2. SSN de origen específico sobre SSN comodín
3. TT de origen específico sobre TT comodín
4. NPI de origen específico sobre NPI comodín
5. NAI de origen específico sobre NAI comodín
6. Número de prioridad más bajo

Ejemplos de Configuración

```
config :omniss7,
  enable_gt_routing: true,

  m3ua_gt_routes: [
    # Ejemplo 1: Coincidir y transformar 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 indicador
    %{
      gt_prefix: "33",
      source_ssn: 8,     # Coincidir tráfico SMSC
      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 la 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 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: Se selecciona la Ruta C (más específica: coincide con GT + TT + NPI)

El mensaje se reenvía con indicadores transformados según los valores de `dest_tt`, `dest_npi`, `dest_nai` de la Ruta C.

Ejemplos de Enrutamiento GT

GT Llamado	SSN de Origen	TT	NPI	NAI	Ruta Coincidente	Razón
447712345678	6	-	-	-	"447" → par 3	Coincidencia prefijo más la
441234567890	6	-	-	-	"44" → par 1	Coincidencia prefijo, no ha ruta más específica
12125551234	6	-	-	-	"1" → par 2	Coincidencia prefijo para números de E UU.
555881234567	8	-	-	-	"555" (SSN 8) → par 4	Coincidencia GT + SSN, reescribe SSN 6
555881234567	6	-	-	-	"555" (comodín de SSN) → par X	Coincidencia GT, sin reescritura de SSN
441234567890	6	0	1	4	"44" (TT=0) → par 1	Coincidencia GT + TT, transforma T 3
12125551234	8	0	1	4	"1" (TT=0, NPI=1, NAI=4)	Más específic coincidencia GT+TT+NPI+

Casos de Uso Prácticos para Enrutamiento TT/NPI/NAI

1. **Normalización de Interconexión de Redes**
    - Diferentes redes pueden usar diferentes convenciones de indicadores
    - Transformar indicadores en el punto de interconexión para asegurar compatibilidad
    - Ejemplo: La red asociada usa TT=0 para internacional, su red usa TT=1
2. **Conversión de Protocolo**
    - Convertir entre planes de numeración al enrutar entre diferentes tipos de redes
    - Ejemplo: Ruta desde la red móvil (NPI=6) a PSTN (NPI=1)
3. **Estandarización del Formato de Dirección**
    - Normalizar todo el tráfico entrante para usar valores NAI consistentes
    - Ejemplo: Convertir todo el formato internacional (NAI=4) a formato nacional (NAI=3) para enrutamiento nacional
4. **Enrutamiento Específico de Proveedor**
    - Enrutar según el tipo de traducción a diferentes proveedores de servicios
    - Ejemplo: TT=0 se enruta a Carrier A, TT=2 se enruta a Carrier B
5. **Integración de Sistemas Legados**
    - Los sistemas modernos pueden usar diferentes valores de indicadores que los sistemas legados
    - Transformar en el STP para mantener la compatibilidad hacia atrás

---

Manejo de Tránsito SCCP

Más allá de seleccionar un par de destino, el STP realiza las funciones de tránsito de capa SCCP requeridas de un verdadero Punto de Control de

Conexión de Señalización. Estas se aplican a las clases de mensajes sin conexión que el STP reenvía: **UDT** (0x09), **XUDT** (0x11) y **LUDT** (0x13).

Ruta sobre SSN Después de la Traducción de Título Global

Por defecto, el STP **preserva el indicador de enrutamiento de la Parte Llamada como ruta sobre GT** cuando reenvía un mensaje coincidente con una ruta GT. Esto es deliberado: seleccionar un siguiente salto por prefijo GT es *enrutamiento GT* (intermedio), no *traducción final de GT*. Según ITU-T Q.714 §2.4, un nodo intermedio debe dejar el indicador de enrutamiento como ruta sobre GT para que el nodo descendente realice la traducción. Cambiar a ruta sobre SSN allí etiqueta incorrectamente un resultado intermedio como final. El mensaje luego lleva un GT con ruta sobre SSN y sin código de punto, que un nodo descendente (por ejemplo, un MSS que esperaba ejecutar su propio GTT) rechaza con `providerAbort(abnormalDialogue)`.

Establezca el indicador de enrutamiento a ruta sobre SSN **solo** cuando *este* STP sea el traductor final y la ruta entregue a un punto final solo SSN que no pueda realizar más GTT. Eso es una opción explícita por ruta a través de la bandera `route_on_ssn: true` en la ruta GT:

```
m3ua_gt_routes: [
  # Enrutamiento de tránsito - preserva ruta sobre GT
(predeterminado); el nodo descendente realiza GTT
  %{gt_prefix: "44", peer_id: 2, priority: 1, description:
"Números del Reino Unido"},

  # Entrega final a un punto final solo SSN - cambiar a ruta sobre
SSN
  %{
    gt_prefix: "1234",
    peer_id: 1,
    priority: 1,
    dest_ssn: 8,
    route_on_ssn: true,
    description: "Entrega final a MSC solo SSN"
  }
]
```

La bandera también se puede establecer por ruta a través de la API de enrutamiento (`route_on_ssn` en `POST /api/routing/gt-routes`) y predetermina a `false`. Cuando se habilita, el cambio solo se aplica a una ruta actualmente dirigida a la Parte Llamada con ruta sobre GT que lleva un SSN no cero (ITU-T Q.714 §2.4).

Aplicación del Contador de Saltos (XUDT / LUDT)

Los mensajes XUDT y LUDT llevan un **contador de saltos SCCP** que protege contra bucles de enrutamiento entre nodos de retransmisión SCCP. En cada salto de tránsito, el STP:

1. **Decrementa** el contador de saltos en el mensaje reenviado (Q.714 §2.4).
2. Si decrementar llevaría el contador a **cero**, el mensaje es **descartado** en lugar de reenviado (Q.714 §2.8.6).
3. Un descarte incrementa `m3ua_stp_routing_failures_total{reason="hcounter_violation"}`.
4. Si el mensaje original solicitó retorno en caso de error, se genera un mensaje `XUDTS`/`LUDTS` de retorno al originador (ver más abajo).

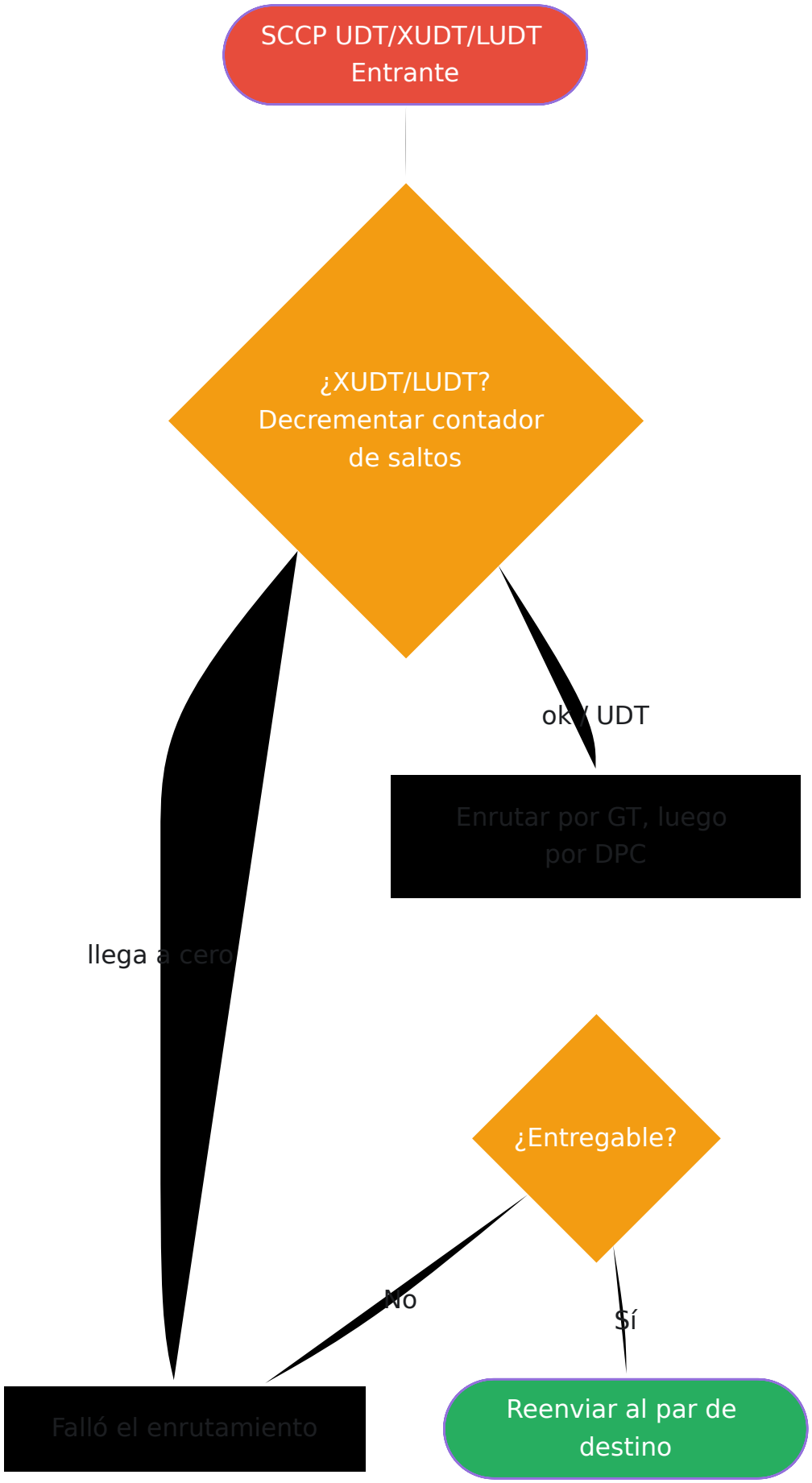
UDT no tiene contador de saltos y se reenvía sin cambios en este paso.

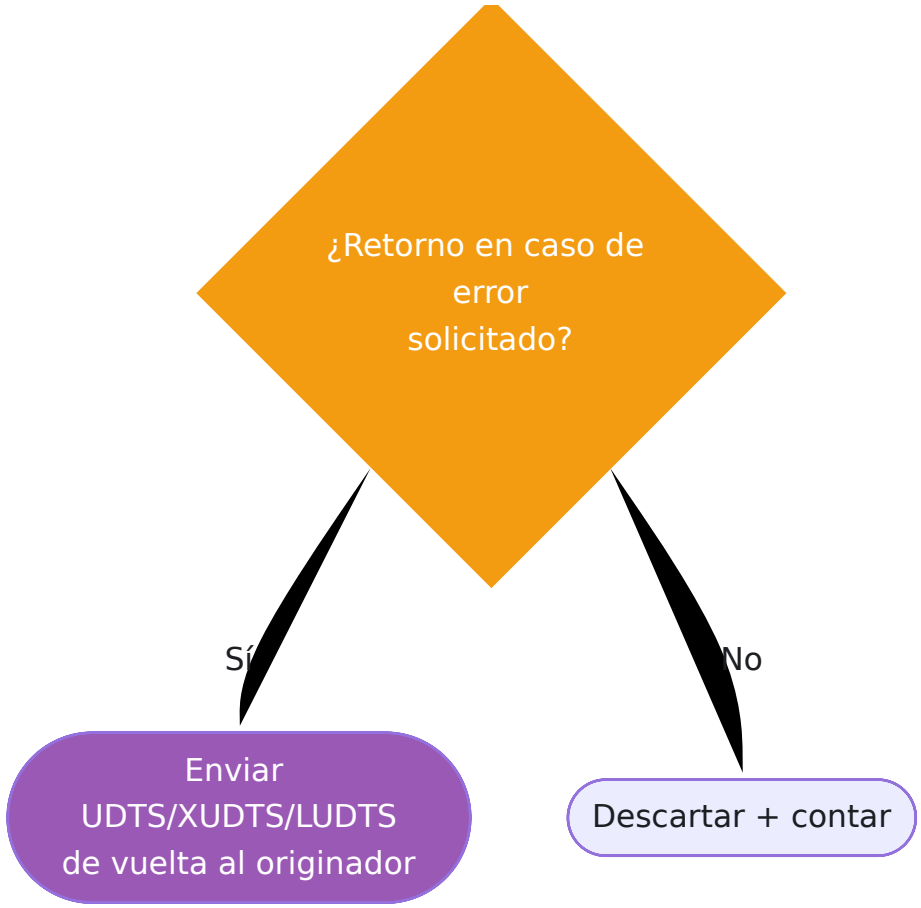
Retorno de Mensaje en Caso de Fallo de Enrutamiento (UDTS / XUDTS / LUDTS)

Cuando un mensaje sin conexión no puede ser entregado (sin ruta, destino prohibido, contador de saltos agotado, etc.) **y el originador estableció la opción de retorno en caso de error**, el STP construye un mensaje de retorno SCCP y lo envía de vuelta hacia el originador (ITU-T Q.714 §4.2):

Mensaje original	Devuelto como
UDT (0x09)	UDTS (0x0A)
XUDT (0x11)	XUDTS (0x12)
LUDT (0x13)	LUDTS (0x14)

La causa de retorno llevada en el mensaje refleja la razón del fallo (por ejemplo, un fallo de MTP por un destino prohibido, o una violación de contador de saltos por un contador agotado). Si el originador **no** solicitó retorno en caso de error, el mensaje se descarta simplemente y se cuenta.


SCCP UDT/XUDT/LUDT
Entrante
¿XUDT/LUDT?
Decrementar contador
de saltos
ok / UDT
Enrutar por GT, luego
por DPC
llega a cero
¿Entregable?
No
Sí
Falló el enrutamiento
Reenviar al par de
destino



Estado de Destino (Gestión de Rutas de Señalización)

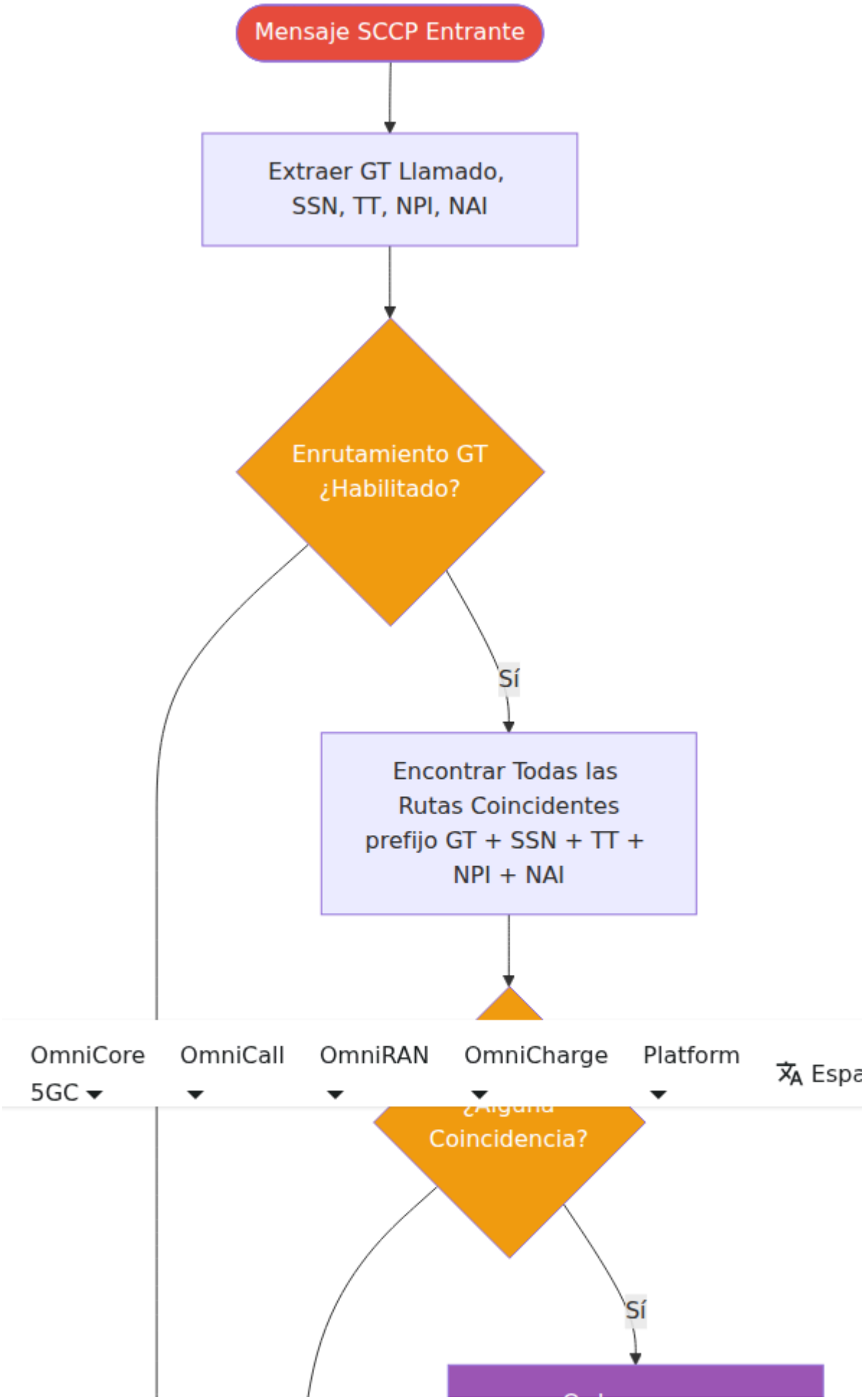
El STP rastrea la disponibilidad de cada código de punto de destino utilizando mensajes de gestión de rutas de señalización MTP3, según ITU-T Q.704 §13. El estado se mantiene en la tabla Mnesia `m3ua_dest_state`; la ausencia de una fila significa que el destino está **permitido** (el predeterminado).

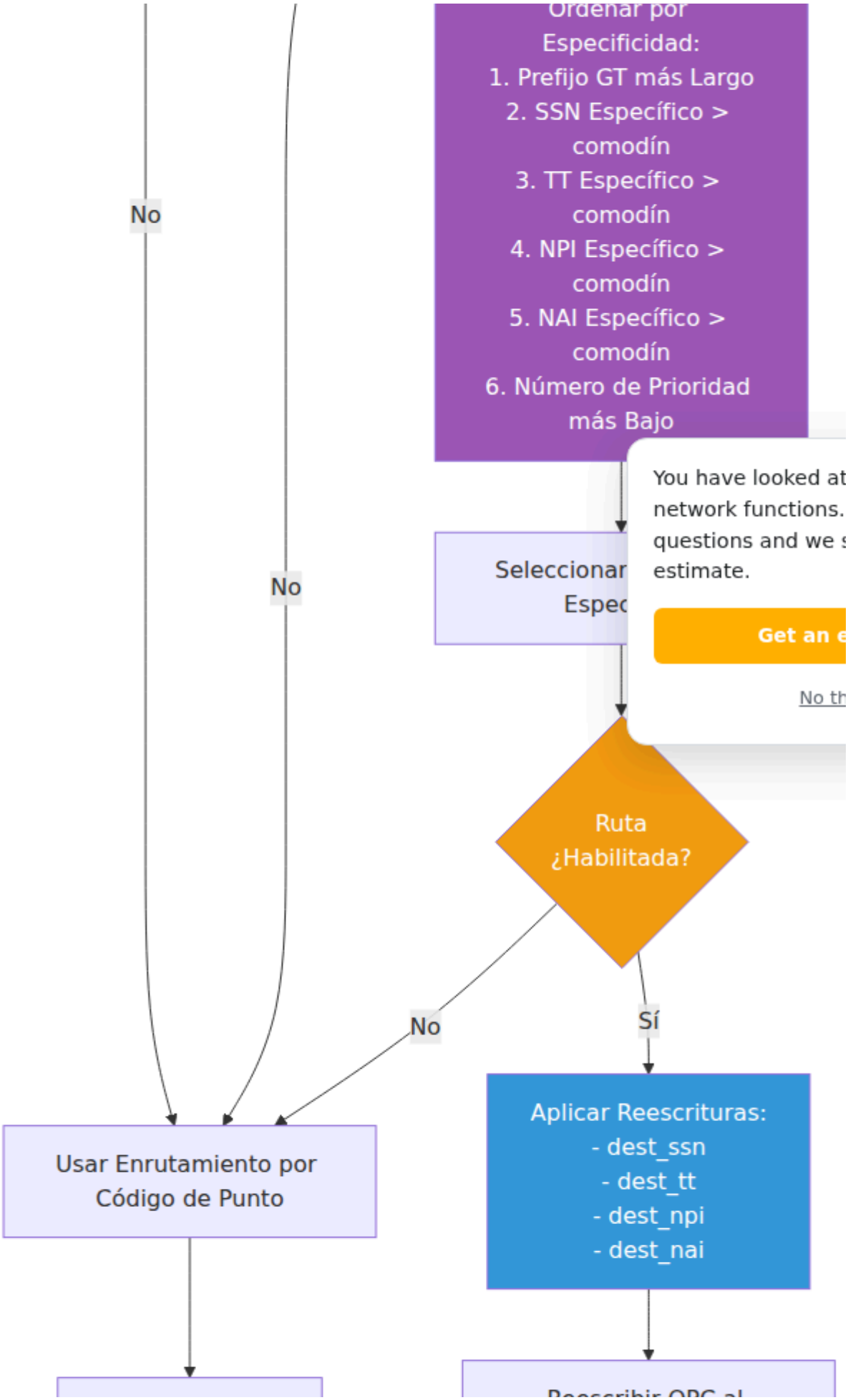
Manejo de TFP / TFA / TFR

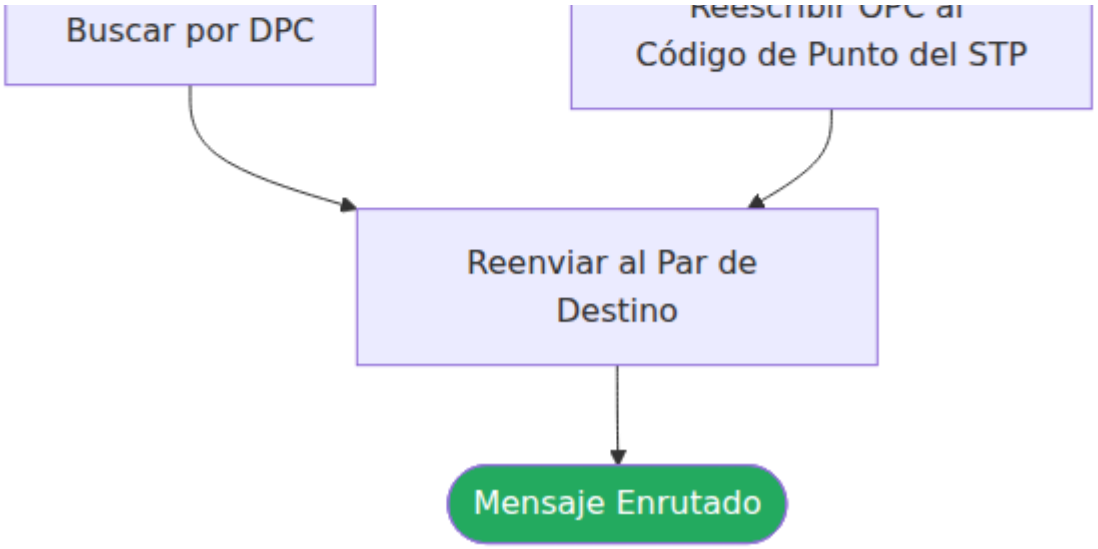
Mensaje	Significado	Efecto en el estado del STP
TFP (Transferencia Prohibida)	Un destino ya no es alcanzable a través de esta ruta	Destino marcado como `:prohibited`
TFA (Transferencia Permitida)	Un destino es alcanzable nuevamente	Estado del destino borrado (de vuelta a permitido)
TFR (Transferencia Restringida)	Un destino debe usarse solo cuando no existe una alternativa	Destino marcado como `:restricted`

Efecto en el Enrutamiento

- Durante una búsqueda de Código de Punto, el STP primero verifica el estado del destino. Un destino marcado como **transferencia prohibida** se omite. La búsqueda devuelve `{:error, :prohibited}` y el mensaje no se enruta allí (Q.704 §13.2.3). Esto incrementa `m3ua_stp_routing_failures_total{reason="prohibited"}`.
- Las solicitudes **DAUD** (Auditoría de Destino) se responden desde este estado real en lugar de con un "disponible" general: los destinos alcanzables se informan con **DAVA**, y los destinos inalcanzables o prohibidos con **DUNA** (RFC 4666 §3.4.3).


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
OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Platform
Espa
Coincidencia?
Sí


No
No
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. Número de Prioridad
más Bajo
Seleccionar
Espec
You have looked at
network functions.
questions and we
estimate.
Get an e
No th
Ruta
¿Habilitada?
No
Sí
Usar Enrutamiento por
Código de Punto
Aplicar Reescrituras:
- dest_ssn
- dest_tt
- dest_npi
- dest_nai



Filtrado de Puerta de Enlace (Cortafuegos SCCP)

El filtrado de puerta de enlace permite al STP **permitir o denegar tráfico de tránsito** antes de que sea enrutado. Sin él, el STP reenvía cada mensaje incondicionalmente. Con él, los operadores pueden bloquear señales no deseadas en la puerta de enlace. Este es un control de seguridad central de SS7.

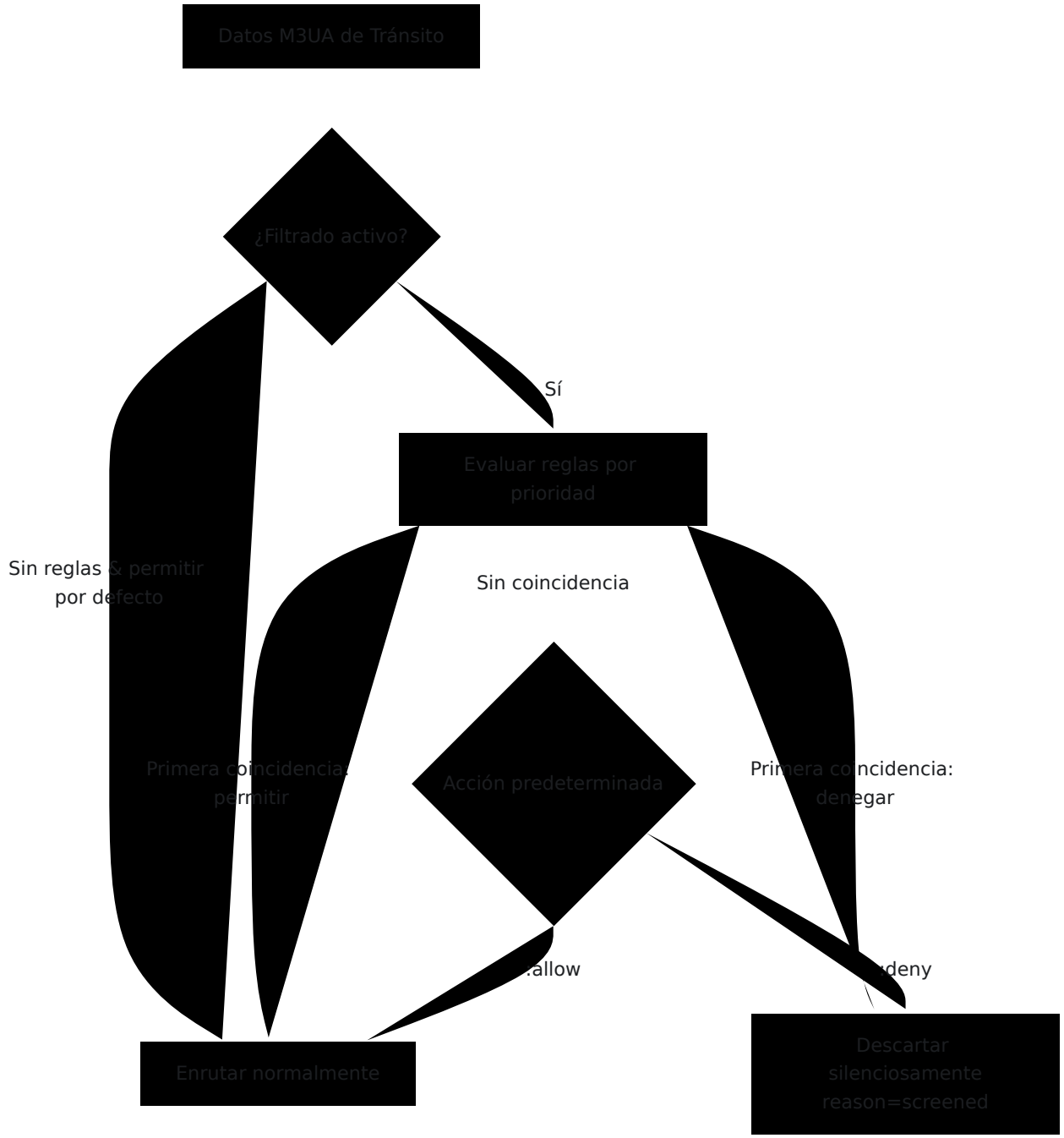
Cada mensaje de datos M3UA de tránsito se evalúa contra un conjunto ordenado de reglas. Una regla coincide con cualquier combinación de los atributos a continuación (un atributo no establecido es un comodín):

Atributo	Coincide
Par de origen	El par del que llegó el mensaje
OPC	Código de Punto de Origen
DPC	Código de Punto de Destino
Indicador de Servicio	SI MTP3
Prefijo GTA de la Parte Llamada	Título Global de la Parte Llamada SCCP (coincidencia de prefijo)
Prefijo GTA de la Parte Llamadora	Título Global de la Parte Llamadora SCCP (coincidencia de prefijo)
SSN	Número de Subsistema SCCP

La **primera regla habilitada que coincida** (valor de `priority` más bajo primero) decide el resultado: `permitir` o `denegar`. Si ninguna regla coincide, se aplica la **acción predeterminada** configurable (`config :omniss7, :screening_default_action`, predeterminado `:allow`).

Un mensaje denegado es **descartado silenciosamente** y contado como `m3ua_stp_routing_failures_total{reason="screened"}`. No se genera UDTS/retorno, porque un cortafuegos no debe filtrar el bloqueo de vuelta al remitente.

Comportamiento



La evaluación se omite por completo (costo cero en la ruta caliente) cuando no hay reglas configuradas **y** la acción predeterminada es `:allow`; se activa tan pronto como existe una regla o se establece la acción predeterminada en `:deny`.

Acción Predeterminada

Parámetro	Tipo	Predeterminado	Descripción
`screening_default_action`	`:allow` | `:deny`	`:allow`	Acción aplicada cuando ninguna regla coincide. Establezca en `:deny` para una postura de denegación predeterminada (lista de permitidos).

```
config :omniss7,
  # Bloquear todo excepto lo que las reglas permiten
explícitamente.
  screening_default_action: :deny
```

Las reglas se almacenan en la tabla Mnesia `ss7_screen_rule` y se gestionan programáticamente (cada regla lleva una prioridad, los atributos de coincidencia anteriores, una acción `allow`/`deny` y una bandera `enabled`). Consulte la Referencia de Configuración para el parámetro de acción predeterminada.

Características de Gestión de Rutas

Deshabilitar Rutas

Las rutas se pueden deshabilitar temporalmente sin eliminarlas. Esto es útil para pruebas, mantenimiento o gestión del tráfico.

Bandera Habilitada

Tanto las rutas de Código de Punto como las de Título Global admiten una bandera `enabled` opcional:

```
config :omniss7,
  m3ua_routes: [
    # Ruta activa
    %{
      dest_pc: 100,
      peer_id: 1,
      priority: 1,
      network_indicator: :international,
      enabled: true  # Ruta está activa (predeterminado si se
omite)
    },

    # Ruta deshabilitada (no evaluada durante el enrutamiento)
    %{
      dest_pc: 200,
      peer_id: 2,
      priority: 1,
      network_indicator: :international,
      enabled: false  # Ruta está deshabilitada
    }
  ],

  m3ua_gt_routes: [
    # Ruta GT deshabilitada
    %{
      gt_prefix: "44",
      peer_id: 1,
      priority: 1,
      description: "Números del Reino Unido - temporalmente
deshabilitada",
      enabled: false
    }
  ]
```

Comportamiento Predeterminado:

- Si `enabled` no se especifica, las rutas se establecen de forma predeterminada en `enabled: true`
- Las rutas deshabilitadas se omiten por completo durante la búsqueda de rutas
- Utilice la interfaz web para activar/desactivar rutas sin editar la configuración

Casos de Uso:

- Probar cambios en el flujo de tráfico
- Ventanas de mantenimiento temporales
- Pruebas A/B de diferentes rutas de enrutamiento
- Implementación gradual de nuevas rutas

Rutas DROP: Prevención de Bucles de Enrutamiento

Las rutas DROP (con `peer_id: 0`) descartan silenciosamente el tráfico en lugar de reenviarlo. Esto previene bucles de enrutamiento y permite un filtrado avanzado del tráfico.

Configuración de Rutas DROP

```
config :omniss7,
  m3ua_routes: [
    # Ruta DROP para un código de punto específico
    %{
      dest_pc: 999,
      peer_id: 0,        # peer_id=0 significa DROP
      priority: 1,
      network_indicator: :international
    }
  ],

  m3ua_gt_routes: [
    # Ruta DROP para prefijo GT
    %{
      gt_prefix: "999",
      peer_id: 0,        # peer_id=0 significa DROP
      priority: 99,
      description: "Bloquear rango de prueba"
    }
  ]
```

Cómo Funcionan las Rutas DROP

Cuando un mensaje coincide con una ruta DROP:

1. El motor de enrutamiento identifica `peer_id: 0`
2. El mensaje es **silenciosamente descartado** (no reenviado)
3. Se genera un **registro INFO**: `"DROP route matched for DPC 999"` o `"DROP route matched for GT 999"`
4. La búsqueda de enrutamiento devuelve `{:error, :dropped}`

Importante: El tráfico descartado se registra a nivel INFO para monitoreo y solución de problemas.

Caso de Uso Común: Lista Blanca de Prefijos

Uno de los usos más poderosos de las rutas DROP es **la lista blanca de prefijos** - permitiendo solo números específicos dentro de un gran rango mientras se bloquean todos los demás.

El Patrón:

1. Crear una ruta DROP para todo el prefijo con un **número de prioridad alto** (por ejemplo, 99)
2. Crear rutas de permiso específicas para números individuales con **números de prioridad bajos** (por ejemplo, 1)
3. Dado que los números de prioridad más bajos se evalúan primero, las rutas permitidas coinciden antes que la ruta DROP
4. Cualquier número que no esté explícitamente permitido es atrapado por la ruta DROP

Escenario de Ejemplo:

Tienes un prefijo GT `1234` que representa un rango de 10,000 números (1234000000 - 1234999999), pero solo deseas enrutar 3 números específicos: `1234567890`, `1234555000`, y `1234111222`.

```
config :omniss7,
  m3ua_gt_routes: [
    # Ruta DROP con número de prioridad ALTA (evaluada al final)
    %{
      gt_prefix: "1234",
      peer_id: 0,              # DROP
      priority: 99,            # Número alto = baja prioridad =
evaluada al final
      description: "Bloquear todos los 1234* excepto números en la
lista blanca"
    },

    # Rutas de permiso específicas con números de prioridad BAJA
(evaluadas primero)
    %{
      gt_prefix: "1234567890",
      peer_id: 1,              # Ruta al peer 1
      priority: 1,             # Número bajo = alta prioridad =
evaluada primero
      description: "Número permitido 1"
    },
    %{
      gt_prefix: "1234555000",
      peer_id: 1,
      priority: 1,
      description: "Número permitido 2"
    },
    %{
      gt_prefix: "1234111222",
      peer_id: 1,
      priority: 1,
      description: "Número permitido 3"
    }
  ]
```

Comportamiento de Enrutamiento:

GT Entrante	Rutas Coincidentes	Ruta Seleccionada	Acción
1234567890	✅ "1234567890" (prioridad 1) ✅ "1234" DROP (prioridad 99)	"1234567890" (más específica, mayor prioridad)	**Enrutado al peer 1**
1234555000	✅ "1234555000" (prioridad 1) ✅ "1234" DROP (prioridad 99)	"1234555000" (más específica, mayor prioridad)	**Enrutado al peer 1**
1234111222	✅ "1234111222" (prioridad 1) ✅ "1234" DROP (prioridad 99)	"1234111222" (más específica, mayor prioridad)	**Enrutado al peer 1**
1234999999	✅ "1234" DROP (prioridad 99)	"1234" DROP (única coincidencia)	**Descartado + registrado**
1234000000	✅ "1234" DROP (prioridad 99)	"1234" DROP (única coincidencia)	**Descartado + registrado**

Resultado:

- ✅ Solo 3 números específicos son enrutados al peer 1
- ✅ Todos los demás números `1234*` son descartados silenciosamente
- ✅ Todo el tráfico descartado se registra para monitoreo

Registros Generados:

```
[INFO] DROP route matched for GT 1234999999
[INFO] DROP route matched for GT 1234000000
```

Rutas DROP para Códigos de Punto

El mismo patrón de lista blanca funciona para el enrutamiento de Códigos de Punto:

```
config :omniss7,
  m3ua_routes: [
    # DROP todo el rango /8 (64 códigos de punto: 1000-1063)
    %{
      dest_pc: 1000,
      peer_id: 0,
      priority: 99,
      mask: 8,
      network_indicator: :international
    },

    # Permitir PCs específicas
    %{dest_pc: 1010, peer_id: 1, priority: 1, network_indicator:
:international},
    %{dest_pc: 1020, peer_id: 1, priority: 1, network_indicator:
:international},
    %{dest_pc: 1030, peer_id: 1, priority: 1, network_indicator:
:international}
  ]
```

Resultado: Solo los PCs 1010, 1020 y 1030 son enrutados. Todos los demás PCs en el rango 1000-1063 son descartados.

Monitoreo de Rutas DROP

Verificar Registros:

```
# Monitorear tráfico descartado
tail -f logs/app.log | grep "DROP route matched"

# Salida esperada:
[INFO] DROP route matched for GT 1234999999
[INFO] DROP route matched for DPC 1050
```

A través de la Interfaz Web:

- Navegar a la pestaña **Registros del Sistema**

- Filtrar por nivel `INFO`
- Buscar "DROP route matched"

Mejores Prácticas:

1. ⚠ Monitorear los registros regularmente para asegurar que las rutas DROP no estén bloqueando tráfico legítimo
2. □ Utilizar campos `description` descriptivos para documentar por qué se descartan las rutas
3. □ Utilizar números de prioridad altos (90-99) para las rutas DROP para asegurar que sean rutas de captura
4. □ Probar el comportamiento de la ruta DROP antes de implementar en producción
5. □ Configurar alertas para aumentos inesperados en el tráfico descartado

---

Enrutamiento Avanzado: Enrutamiento y Reescritura Basados en SSN

Números de Subsistema (SSN)

Los Números de Subsistema identifican la capa de aplicación:

- **SSN 6**: HLR (Registro de Ubicación del Hogar)
- **SSN 7**: VLR (Registro de Ubicación del Visitante)
- **SSN 8**: MSC (Centro de Conmutación Móvil) / SMSC (Centro de SMS)
- **SSN 9**: GMLC (Centro de Ubicación Móvil de Puerta de Enlace)

Ejemplo de Enrutamiento Basado en SSN

Enrutar tráfico SMS a diferentes HLR según el prefijo del número:

```
m3ua_gt_routes: [
  # Enrutar SMS para números del Reino Unido a HLR del Reino
Unido, reescribir SSN de 8 (SMSC) a 6 (HLR)
  %{
    gt_prefix: "44",
    source_ssn: 8,                    # Coincidir con SSN entrante
8 (SMSC)
    peer_id: 1,
    dest_ssn: 6,                      # Reescribir a SSN 6 (HLR)
    priority: 1,
    description: "SMS del Reino Unido a HLR"
  },

  # Enrutar tráfico de voz para números del Reino Unido (SSN 6)
sin reescritura
  %{
    gt_prefix: "44",
    source_ssn: 6,                    # Coincidir con SSN entrante
6 (HLR)
    peer_id: 1,
    dest_ssn: nil,                    # Sin reescritura de SSN
    priority: 1,
    description: "Tráfico de voz del Reino Unido"
  }
]
```

Pruebas de Configuración de Enrutamiento STP

Después de configurar pares y rutas, verifique su configuración:

1. Verificar Estado del Par

A través de la Interfaz Web:

- Navegar a http://localhost
- Verificar la página de Estado de M3UA

- Verificar que los pares muestren **Estado: ACTIVO**

A través de la Consola IEx:

```
# Obtener todos los estados de pares
SCTP.SocketHandler.get_peers_status()

# Salida esperada:
# [
#   %{peer_id: 1, name: "Partner_STP_West", status: :active,
point_code: 100, ...},
#   %{peer_id: 2, name: "Local_HLR", status: :active, point_code:
200, ...}
# ]
```

2. Probar Enrutamiento de Código de Punto

```
# Enviar mensaje M3UA de prueba a DPC 100
test_payload = <<1, 2, 3, 4>>  # Carga útil ficticia
SCTP.SocketHandler.route_by_pc(100, test_payload, 0)

# Verificar registros para la decisión de enrutamiento
# Registro esperado: "Routing message: OPC=... -> DPC=100 via peer
1"
```

3. Probar Enrutamiento de Título Global

```
# Buscar ruta GT manualmente
M3UARouting.lookup_peer_by_gt("447712345678")

# Devuelve {:ok, peer, rewrites}. `peer` es el registro m3ua_peer
de 13 elementos;
# `rewrites` es un MAP de reescrituras de indicadores (nil =
mantener original):
# {:ok, {:m3ua_peer, 3, "UK_Mobile_Peer", ...},
#       %{dest_ssn: nil, dest_tt: nil, dest_npi: nil, dest_nai:
nil}}

# Buscar ruta GT con SSN (una ruta que reescribe SSN 8 → 6)
M3UARouting.lookup_peer_by_gt("555881234567", 8)

# La reescritura de SSN aparece en el mapa de reescrituras, NO
como un entero:
# {:ok, {:m3ua_peer, 4, "SMS_HLR_Peer", ...},
#       %{dest_ssn: 6, dest_tt: nil, dest_npi: nil, dest_nai:
nil}}
```

4. Monitorear Métricas de Enrutamiento

Acceda a las métricas de Prometheus en `/metrics`

Métricas clave:

```
# Mensajes recibidos por par
m3ua_stp_messages_received_total{peer_name="Partner_STP_West",point_c
1523

# Mensajes enviados por par
m3ua_stp_messages_sent_total{peer_name="Local_HLR",point_code="200"}

# Fallos de enrutamiento (por razón)
m3ua_stp_routing_failures_total{reason="no_route"} 5
m3ua_stp_routing_failures_total{reason="no_gt_route"} 2
m3ua_stp_routing_failures_total{reason="no_active_peer"} 1
m3ua_stp_routing_failures_total{reason="prohibited"} 3
m3ua_stp_routing_failures_total{reason="hcounter_violation"} 4
m3ua_stp_routing_failures_total{reason="screened"} 7
```

Métricas y Monitoreo STP

Métricas Disponibles

Métricas de Tráfico por Par:

- `m3ua_stp_messages_received_total` - Total de mensajes recibidos de cada par
  - Etiquetas: `peer_name`, `point_code`
- `m3ua_stp_messages_sent_total` - Total de mensajes reenviados a cada par
  - Etiquetas: `peer_name`, `point_code`

Métricas de Fallos de Enrutamiento:

- `m3ua_stp_routing_failures_total` - Conteo de fallos de enrutamiento por razón

- Etiquetas: `reason`

`reason`	Significado
`no_route`	No se encontró ruta de Código de Punto para el destino
`no_gt_route`	No se encontró ruta de Título Global, y el enrutamiento de PC también falló
`no_active_peer`	Se coincidió una ruta, pero su par es desconocido/no está activo
`prohibited`	El destino está marcado como prohibido para la transferencia (se recibió un TFP, Q.704 §13)
`hcounter_violation`	El contador de salto SCCP llegó a cero en un XUDT/LUDT (Q.714 §2.8.6)

Interpretación de Métricas

- **Altas cuentas de mensajes**: Indica un flujo de tráfico activo
- **Fallos de enrutamiento**: Indica rutas faltantes, un destino inalcanzable o un mensaje en bucle
  - `no_route`: No se encontró ruta de Código de Punto para el destino
  - `no_gt_route`: No se encontró ruta de Título Global, y el enrutamiento de PC también falló
  - `no_active_peer`: Existe una ruta pero el par objetivo está fuera de servicio/desconocido
  - `prohibited`: Destino marcado como prohibido para la transferencia por la gestión de rutas de señalización
  - `hcounter_violation`: Contador de salto agotado; el mensaje fue descartado (y devuelto si se configuró la devolución en caso de error)

Solución de Problemas con Métricas

Escenario: Sin tráfico llegando al destino

1. Verifique si se están recibiendo mensajes:

```
m3ua_stp_messages_received_total{peer_name="Source_Peer"} > 0
```

2. Verifique si se están enviando mensajes:

```
m3ua_stp_messages_sent_total{peer_name="Dest_Peer"} > 0
```

3. Verifique si hay fallos de enrutamiento:

```
m3ua_stp_routing_failures_total{reason="no_route"} > 0
```

Solución: Si los fallos de enrutamiento son altos, agregue rutas faltantes en la configuración.

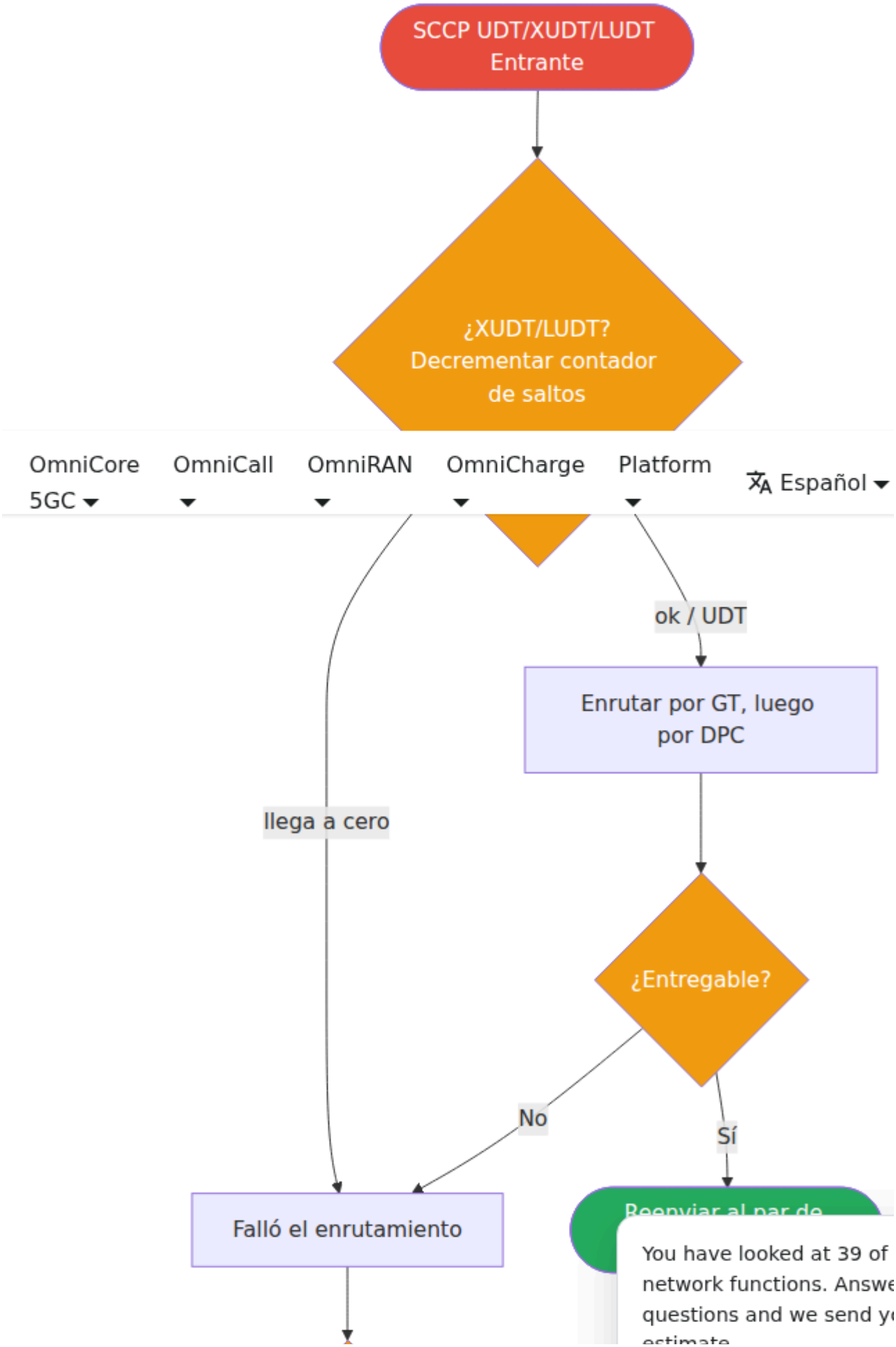
Monitoreo del Estado de los Pares M3UA

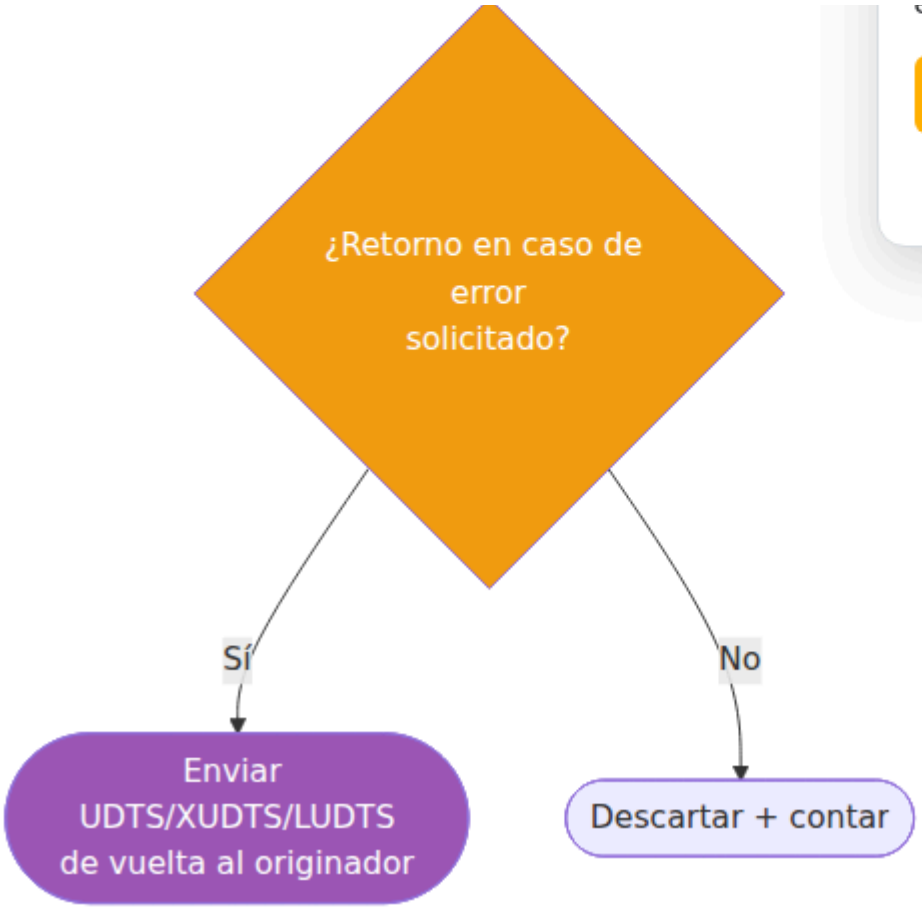
Entendiendo M3UA

M3UA (Capa de Adaptación de Usuario MTP3) es un protocolo que permite que la señalización SS7 se transporte sobre redes IP utilizando SCTP.

Estados de Conexión M3UA

Las conexiones M3UA progresan a través de varios estados:


SCCP UDT/XUDT/LUDT
Entrante
¿XUDT/LUDT?
Decrementar contador
de saltos
OmniCore
5GC
OmniCall
OmniRAN
OmniCharge
Platform
Español
ok / UDT
Enrutar por GT, luego
por DPC
llega a cero
¿Entregable?
No
Sí
Falló el enrutamiento
You have looked at 39 of
network functions. Answe
questions and we send y



Descripciones de Estado:

- **DOWN** - Sin conexión SCTP
- **CONNECTING** - Conexión SCTP en progreso
- **ASPUP_SENT** - Esperando la confirmación de ASPUP
- **INACTIVE** - ASP está activo pero no transportando tráfico activamente (puede transitar a ACTIVE a través de ASPAC)
- **ASPAC_SENT** - Esperando la confirmación de ASPAC
- **ACTIVE** - Listo para tráfico, completamente operativo
- **ASPDOWN_SENT** - Apagado gracioso en progreso

Monitoreo de Pares M3UA a través de la Interfaz Web

La interfaz web proporciona monitoreo en tiempo real de las conexiones de pares M3UA.

Accediendo a la Página de Estado de M3UA:

1. Navegar a la página de inicio de la interfaz web
2. Hacer clic en "Estado de M3UA" en el menú de navegación
3. La página se actualiza automáticamente cada segundo

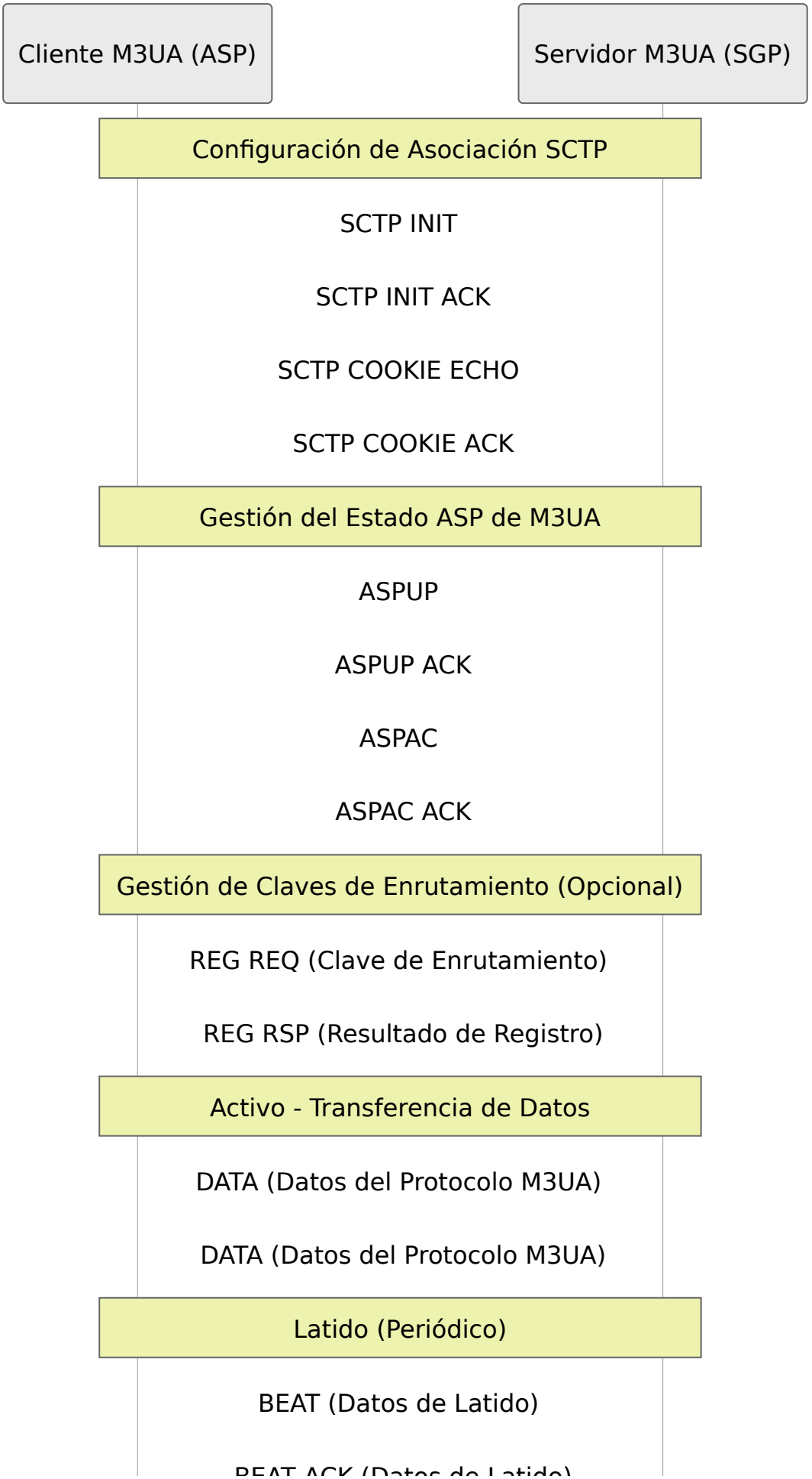
Tabla de Estado de M3UA:

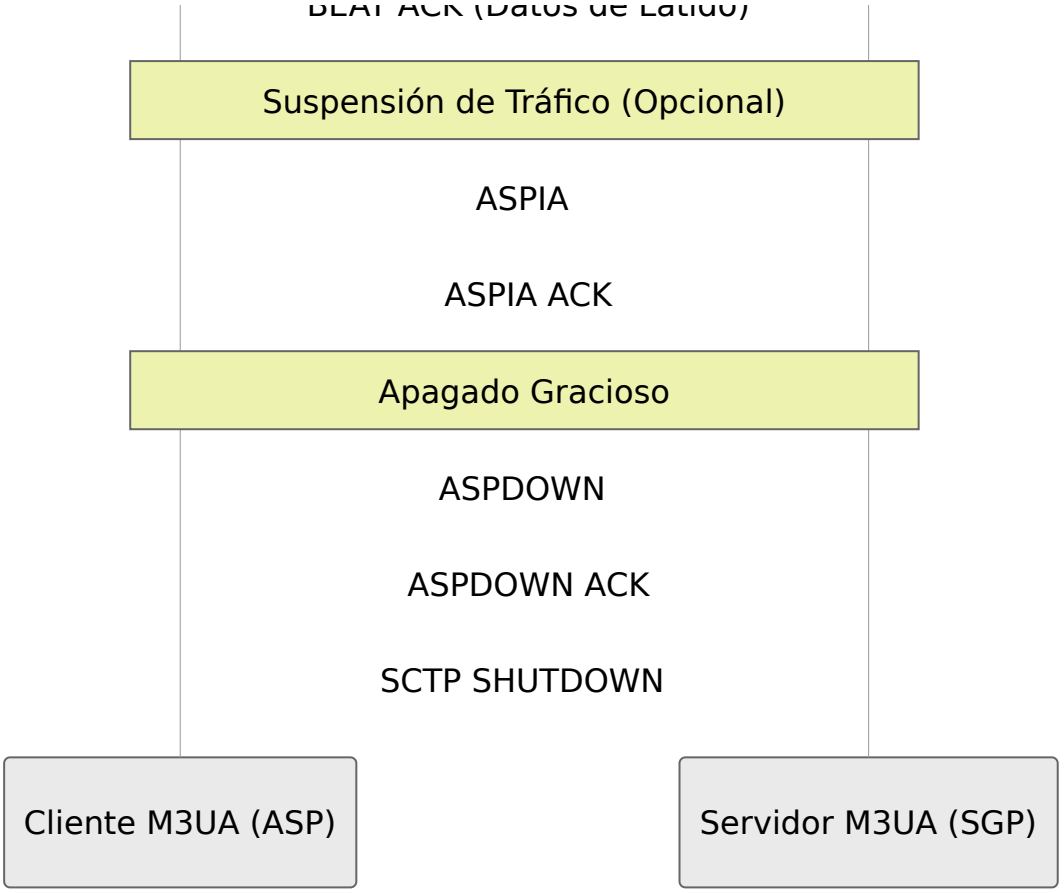
Columna	Descripción
Nombre	Nombre de la conexión (por ejemplo, `testASP`)
PID	Identificador de proceso
Estado	UP (verde) o DOWN (rojo)
Estado ASP	Estado actual de M3UA (por ejemplo, ACTIVE, INACTIVE)
Assoc/SCTP	Estado de asociación SCTP
Local	IP:Puerto local
Remoto	IP:Puerto remoto
RC	ID de Contexto de Enrutamiento

Indicadores de Estado:

- **Verde (UP)** - La conexión está activa y saludable
- **Rojo (DOWN)** - La conexión está caída o no disponible
- **Estado ASP** - Muestra el estado actual de la conexión M3UA
- **Assoc/SCTP** - Muestra el estado de la asociación SCTP

Flujo de Mensajes M3UA


Cliente M3UA (ASP)
Servidor M3UA (SGP)
Configuración de Asociación SCTP
SCTP INIT
SCTP INIT ACK
SCTP COOKIE ECHO
SCTP COOKIE ACK
Gestión del Estado ASP de M3UA
ASPUP
ASPUP ACK
ASPAC
ASPAC ACK
Gestión de Claves de Enrutamiento (Opcional)
REG REQ (Clave de Enrutamiento)
REG RSP (Resultado de Registro)
Activo - Transferencia de Datos
DATA (Datos del Protocolo M3UA)
DATA (Datos del Protocolo M3UA)
Latido (Periódico)
BEAT (Datos de Latido)



Soporte de Mensajes M3UA según RFC 4666

OmniSS7 implementa el conjunto completo de tipos de mensajes M3UA definidos en RFC 4666:

Clase	Tipo	Mensaje	Dirección	Comportamiento
0 (MGMT)	0	ERR	Ambos	Registra el código de error y la información de diagnóstico
0 (MGMT)	1	NTFY	Ambos	Registra la notificación
1 (XFER)	1	DATA	Ambos	Rutas la carga útil a través de PC o GT
2 (SSNM)	1	DUNA	Recibir	Registra destino no disponible
2 (SSNM)	2	DAVA	Recibir	Registra destino disponible
2 (SSNM)	3	DAUD	Recibir	Responde según el estado del destino: DAVA para PCs alcanzables, DUNA para PCs inalcanzables/prohibidos
2 (SSNM)	4	SCON	Recibir	Registra congestión
2 (SSNM)	5	DUPU	Recibir	Registra parte de usuario no disponible
2 (SSNM)	6	DRST	Recibir	Registra destino restringido
3 (ASPSM)	1	ASPUP	Ambos	Envía/recibe ACK de ASPUP
3 (ASPSM)	2	ASPDN	Ambos	Envía ACK de ASPDN, desmantela rutas de pares

Clase	Tipo	Mensaje	Dirección	Comportamiento
3 (ASPSM)	3	BEAT	Ambos	Eco de datos de latido en ACK de BEAT
3 (ASPSM)	4	ASPUP ACK	Recibir	Transiciona par a UP
3 (ASPSM)	5	ASPDN ACK	Recibir	Transiciona par a DOWN
3 (ASPSM)	6	BEAT ACK	Recibir	Reconoce latido
4 (ASPTM)	1	ASPAC	Ambos	Envía/recibe ACK de ASPAC, activa rutas
4 (ASPTM)	2	ASPIA	Ambos	Envía ACK de ASPIA, transiciona par a INACTIVE
4 (ASPTM)	3	ASPAC ACK	Recibir	Transiciona par a ACTIVE
4 (ASPTM)	4	ASPIA ACK	Recibir	Transiciona par a INACTIVE
9 (RKM)	1	REG REQ	Recibe SGP	Analiza clave de enrutamiento, responde con REG RSP
9 (RKM)	2	REG RSP	Recibe ASP	Registra resultado de registro
9 (RKM)	3	DEREG REQ	Recibe SGP	Responde con DEREG RSP

Clase	Tipo	Mensaje	Dirección	Comportamiento
9 (RKM)	4	DEREG RSP	Recibe ASP	Registra resultado de desregistro

Latido (BEAT/BEAT ACK): Cuando el STP recibe un mensaje BEAT, ecoa los datos de latido de vuelta en un BEAT ACK. Esto permite a los pares detectar fallos de enlace sin depender únicamente de los tiempos de espera a nivel SCTP.

Gestión de Claves de Enrutamiento (RKM): Cuando un ASP envía un REG REQ, el SGP analiza la clave de enrutamiento (identificador RK local, contexto de enrutamiento, modo de tráfico, DPC) y responde con un REG RSP que contiene el estado del registro. Esto permite la provisión dinámica de rutas sin requerir configuración estática.

ASP Abajo (ASPDN): Cuando el SGP recibe un ASPDN, responde con ACK de ASPDN, marca al par como DOWN y elimina las rutas asociadas de la tabla de enrutamiento.

ASP Inactivo (ASPIA): Cuando el SGP recibe un ASPIA, responde con ACK de ASPIA y transiciona al par a INACTIVE, pausando el tráfico sin desmantelar la asociación.

Manejo del Contexto de Enrutamiento M3UA

El **Contexto de Enrutamiento (RC)** es un identificador de 32 bits en M3UA que identifica el contexto del Servidor de Aplicaciones (AS) en cada extremo. Entender cómo se maneja el RC en diferentes puntos de la ruta de señalización es importante para una correcta configuración de la red.

¿Qué es el Contexto de Enrutamiento?

Según RFC 4666 Sección 3.3.1, el Contexto de Enrutamiento:

- Identifica de manera única un Servidor de Aplicaciones (AS) en un extremo
- Se transporta en los mensajes DATA de M3UA (etiqueta de parámetro 6)

- Permite a una Puerta de Señalización determinar qué AS debe recibir un mensaje
- Es significativo por salto, no de extremo a extremo

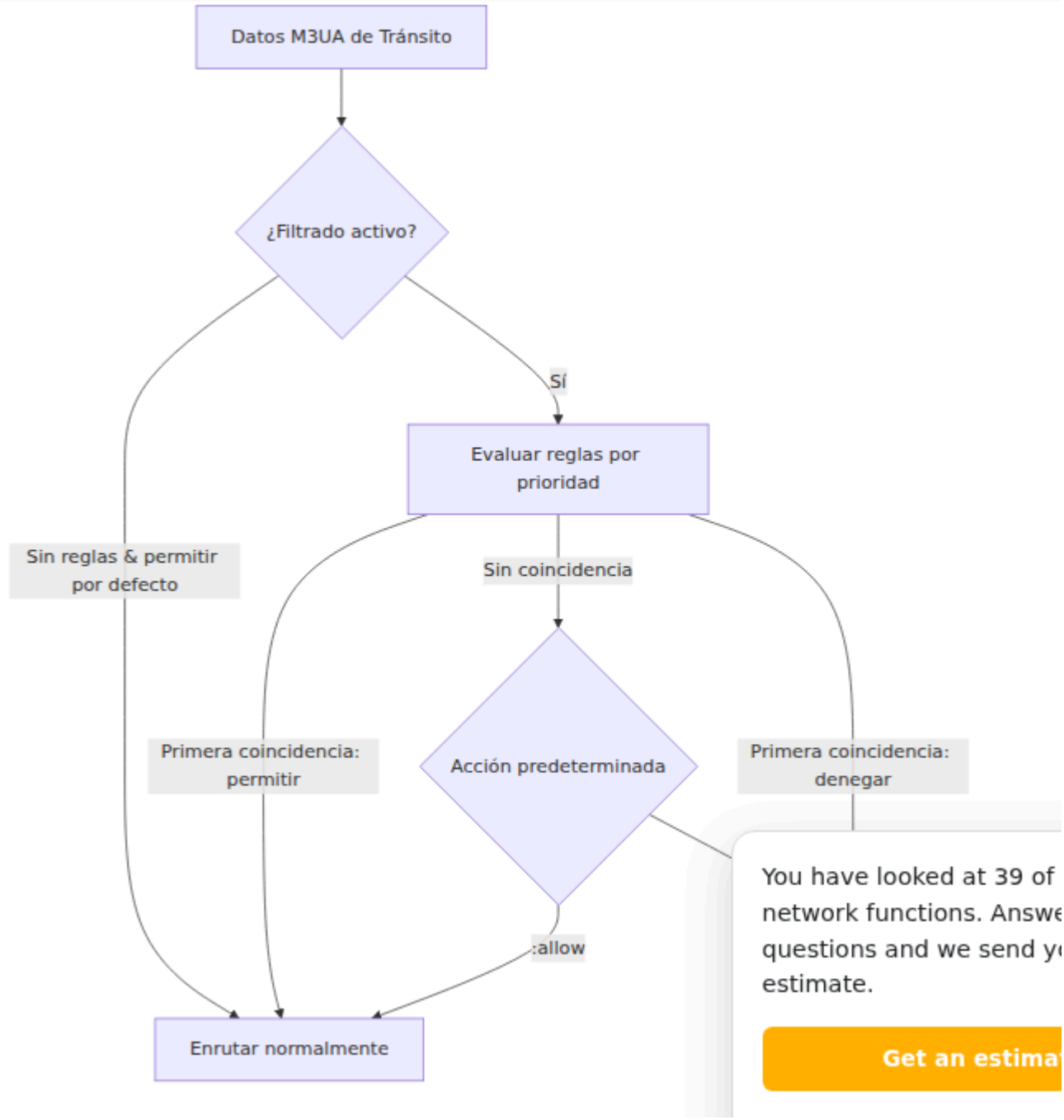
Contexto de Enrutamiento en el Protocolo M3UA



El parámetro de Contexto de Enrutamiento (etiqueta 6) aparece en los mensajes DATA de M3UA junto con el parámetro de Datos del Protocolo (etiqueta 528) que contiene la carga útil real de MTP3/SCCP.

Cómo Maneja OmniSS7 el Contexto de Enrutamiento

OmniSS7 maneja el Contexto de Enrutamiento de manera diferente dependiendo del componente y la dirección del mensaje:



Cliente MAP y Cliente CAP

Para los clientes de capa de aplicación (MAP y CAP), el contexto de enrutamiento se maneja de la siguiente manera:

Escenario	Manejo de RC	Configuración
Respuesta a solicitud entrante	Reflejar el RC entrante en la respuesta	Automático - no se necesita configuración
Solicitud iniciada por el cliente (SRI, PRN, IDP, etc.)	Usar el RC configurado de la configuración de la aplicación	Establecer `routing_context` en `map_client_m3ua` o `cap_client_m3ua`
Sin RC en la solicitud entrante	Omitir RC de la respuesta	Mantiene la compatibilidad hacia atrás
Sin RC configurado para solicitudes de cliente	Omitir RC del mensaje saliente	Mantiene la compatibilidad hacia atrás

Ejemplo de Configuración:

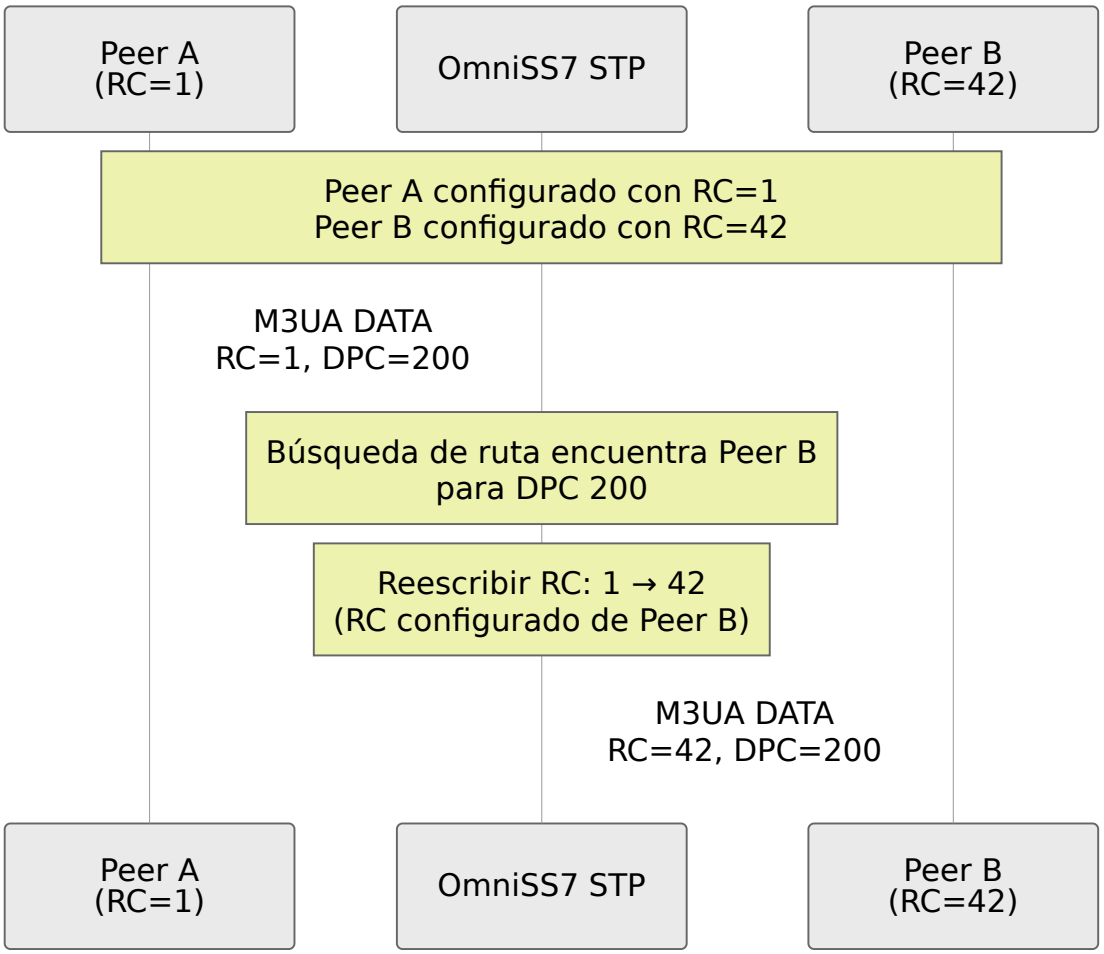
```
config :omniss7,
  # Configuración de RC del Cliente MAP
  map_client_m3ua: %{
    mode: "ASP",
    local_ip: {10, 0, 0, 1},
    local_port: 2905,
    remote_ip: {10, 0, 0, 10},
    remote_port: 2905,
    routing_context: 1    # Usado para solicitudes MAP iniciadas
por el cliente (SRI, PRN, etc.)
  },

  # Configuración de RC del Cliente CAP
  cap_client_m3ua: %{
    mode: "ASP",
    local_ip: {10, 0, 0, 1},
    local_port: 2906,
    remote_ip: {10, 0, 0, 20},
    remote_port: 2906,
    routing_context: 2    # Usado para solicitudes CAP iniciadas
por el cliente (InitialDP, etc.)
  }
```

Reescritura del Contexto de Enrutamiento STP

Cuando el STP reenvía mensajes entre pares, **reescribe** el Contexto de Enrutamiento al RC configurado del par de destino. Este es el comportamiento correcto según el RFC 4666 porque:

1. El RC identifica el contexto AS en cada extremo
2. Diferentes pares pueden estar asociados con diferentes contextos AS
3. El STP traduce los valores de RC al enrutar entre diferentes AS

Flujo de Reescritura de RC del STP:



Configuración:

El contexto de enrutamiento de cada par se configura en la definición del par:

```
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 1,
      name: "Network_A_STP",
      role: :server,
      remote_ip: {10, 0, 0, 100},
      remote_port: 2905,
      routing_context: 1,        # RC esperado por/desde este par
      point_code: 100,
      network_indicator: :international
    },
    %{
      peer_id: 2,
      name: "Network_B_HLR",
      role: :client,
      local_ip: {10, 0, 0, 1},
      local_port: 0,
      remote_ip: {10, 0, 0, 200},
      remote_port: 2905,
      routing_context: 42,       # RC esperado por este par
      point_code: 200,
      network_indicator: :international
    }
  ]
```

Cuando un mensaje llega de Peer 1 (RC=1) destinado a Peer 2, el STP:

1. Enruta el mensaje basado en DPC (200 → Peer 2)
2. Reescribe RC de 1 a 42 (RC configurado de Peer 2)
3. Reenvía el mensaje a Peer 2

Resumen del Manejo de RC

Componente	Solicitud Entrante	Respuesta Saliente	Iniciada por Cliente
Cliente MAP	Extraer RC	Reflejar RC entrante	Usar RC configurado
Cliente CAP	Extraer RC	Reflejar RC entrante	Usar RC configurado
STP	Extraer RC	N/A	Reescribir al RC del par de destino

Cuando se Omitió el RC

Si un par no tiene `routing_context` configurado (o está establecido en `nil`):

- **Respuestas de Cliente MAP/CAP**: El parámetro RC se omite de la respuesta
- **Solicitudes de Cliente MAP/CAP**: El parámetro RC se omite de la solicitud
- **Reenvío del STP**: RC no se reescribe (RC original preservado u omitido)

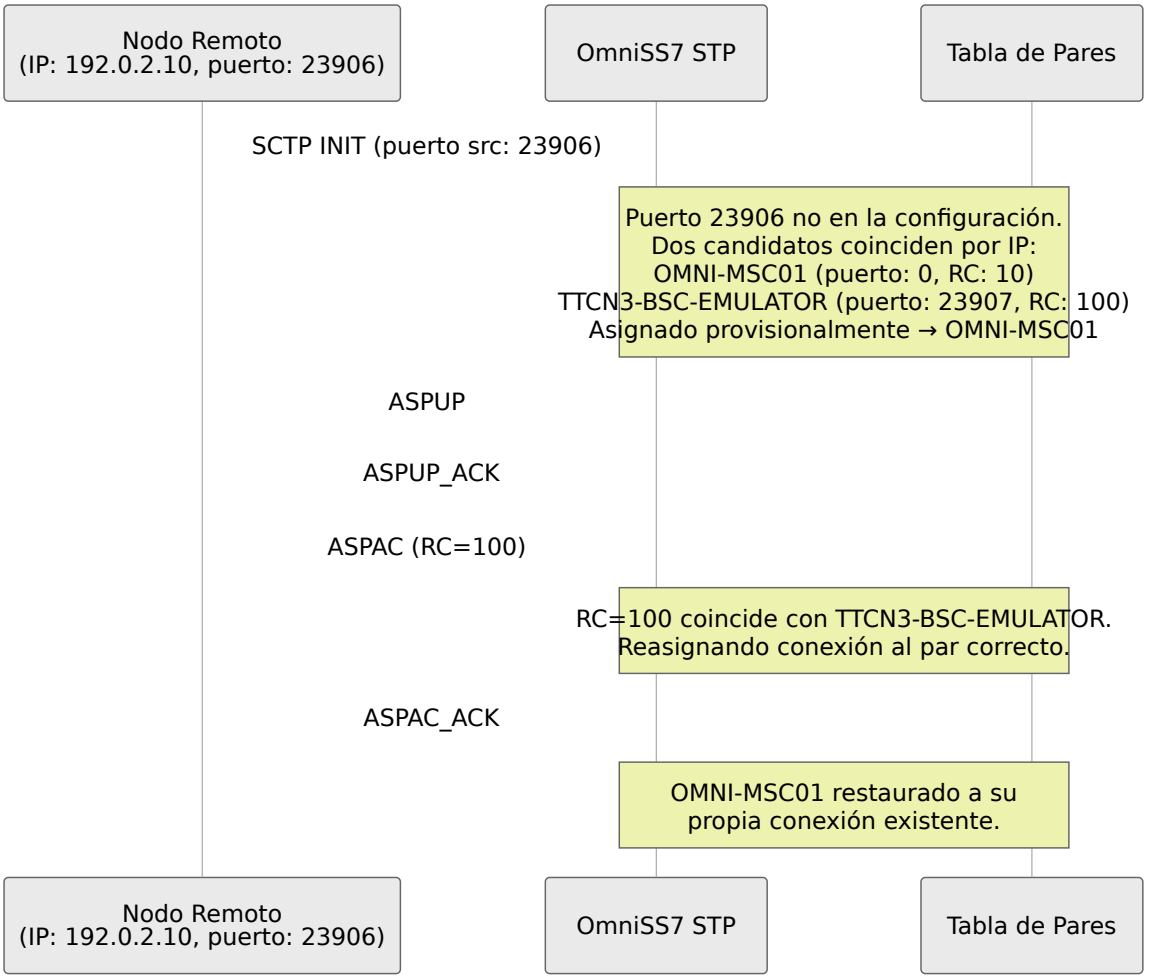
Esto mantiene la compatibilidad hacia atrás con pares que no utilizan o esperan contexto de enrutamiento.

Desambiguación de Múltiples Pares con la Misma IP

Dos pares configurados pueden compartir la misma dirección IP remota: por ejemplo, un MSC de producción y un emulador de prueba que se ejecuta en el mismo host. Cuando llega una conexión SCTP entrante, OmniSS7 no puede determinar la identidad correcta del par solo a partir de la IP de origen. Un par configurado con `remote_port: 0` (comodín) coincidirá con cualquier conexión desde esa IP, lo que puede causar que una nueva conexión del otro par se asigne provisionalmente a la entrada incorrecta.

OmniSS7 resuelve esto automáticamente utilizando el **contexto de enrutamiento ASPAC de M3UA**. La conexión se acepta y se asigna provisionalmente al par que mejor coincida (la coincidencia de puerto específico se prefiere sobre el comodín). Cuando el extremo remoto envía un mensaje `ASPAC` que lleva un contexto de enrutamiento, el sistema compara ese RC con el `routing_context` configurado de cada par. Si el RC coincide con un par diferente, la conexión se reasigna silenciosamente al correcto.

Cómo Funciona



Salida de Registro

Cuando múltiples pares coinciden con una conexión entrante, OmniSS7 registra todos los candidatos para que los operadores puedan verificar que la desambiguación está funcionando correctamente:

```
Conexión entrante de 192.0.2.10:23906 coincide con pares: ["OMNI-
MSC01", "TTCN3-BSC-EMULATOR"] - usando provisionalmente "OMNI-
MSC01" hasta que el RC de ASPAC confirme la identidad
```

Después de recibir el contexto de enrutamiento ASPAC:

```
RC=100 de ASPAC coincide con el par 9 (TTCN3-BSC-EMULATOR), pero
la conexión fue asignada al par 8 (OMNI-MSC01) - reasignando
```

Requisitos de Configuración

Para que la desambiguación de pares con la misma IP funcione correctamente, **cada par debe tener un `routing_context` único configurado**. Si dos pares en la misma IP comparten el mismo RC (o ambos tienen `nil`), la reasignación no puede determinar qué par es correcto.

```
config :omniss7,
  m3ua_peers: [
    %{
      peer_id: 8,
      name: "OMNI-MSC01",
      role: :server,
      remote_ip: {10, 5, 198, 200},
      remote_port: 0,              # Comodín - acepta cualquier
puerto de origen
      routing_context: 10,         # Debe ser único entre pares
que comparten esta IP
      point_code: 500,
      network_indicator: :national
    },
    %{
      peer_id: 9,
      name: "TTCN3-BSC-EMULATOR",
      role: :server,
      remote_ip: {10, 5, 198, 200},
      remote_port: 23907,          # Puerto de origen esperado
(puede diferir para puertos efímeros)
      routing_context: 100,        # RC diferente - usado para
identificar este par después de ASPAC
      point_code: 193,
      network_indicator: :national
    }
  ]
```

> **Nota sobre puertos de origen efímeros**: La coincidencia de puertos es exacta. Si un par está configurado con `remote_port: 23907` pero se conecta desde el puerto 23906, ese par se excluye de la coincidencia inicial y el par comodín gana en su lugar. Esto es esperado. El contexto de enrutamiento ASPAC es la señal de identidad autoritativa. El `remote_port` configurado en un par solo se utiliza para la heurística de asignación de conexión inicial, no para la identidad final. Asigne a cada par un `routing_context` distinto y la reasignación se resolverá correctamente independientemente de qué puerto de origen utilice el remoto.

Solución de Problemas de Conexiones M3UA

Problema: La conexión no se establece

Síntomas:

- El estado muestra DOWN
- Sin asociación SCTP

Verificaciones:

1. Verificar conectividad de red: `ping remote_ip`
2. Comprobar que el firewall permite SCTP (protocolo 132)
3. Verificar que el STP/SGP remoto está escuchando en el puerto correcto
4. Comprobar `remote_ip` y `remote_port` en la configuración
5. Revisar los registros de la aplicación en busca de errores SCTP

Problema: Conexión establecida pero ASP no activo

Síntomas:

- La asociación SCTP existe
- El estado de ASP atascado en INACTIVE o ASPUP_SENT

Verificaciones:

1. Verificar que el contexto de enrutamiento coincida con la configuración remota
2. Comprobar que el STP remoto acepte su código de punto
3. Revisar registros en busca de rechazos de ASPUP/ASPAC
4. Verificar que no haya requisitos de autenticación/seguridad

Problema: No fluye datos

Síntomas:

- El estado de ASP muestra ACTIVE
- No se están enrutando mensajes

Verificaciones:

1. Verificar el contexto de enrutamiento en los mensajes
2. Comprobar la dirección SCCP (formato GT, valores SSN)
3. Verificar que las tablas de enrutamiento estén configuradas correctamente
4. Revisar la página `/events` en busca de errores SCCP
5. Comprobar el enrutamiento del código de punto a nivel de STP

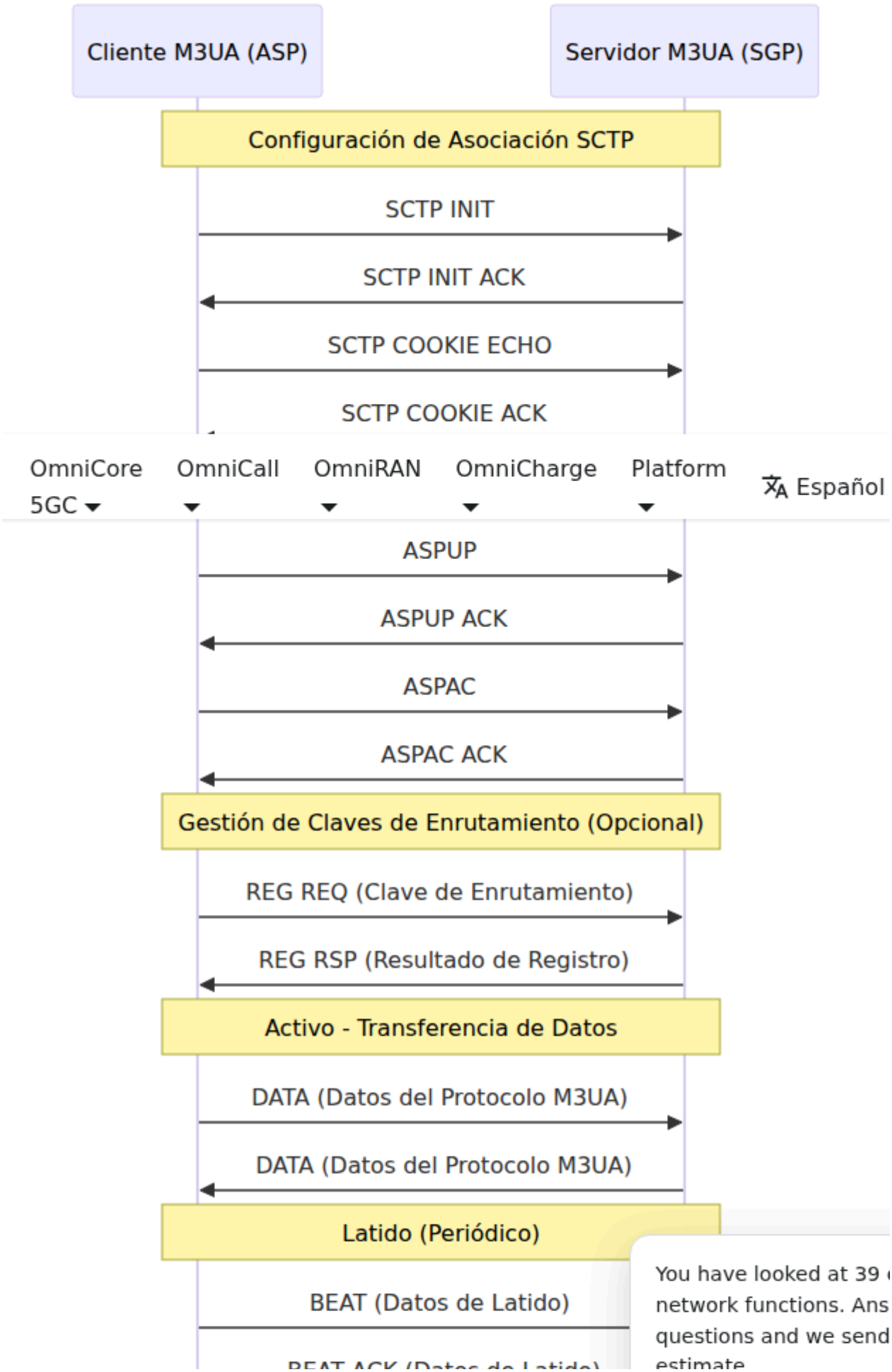
Monitoreo del Estado de Pares M2PA

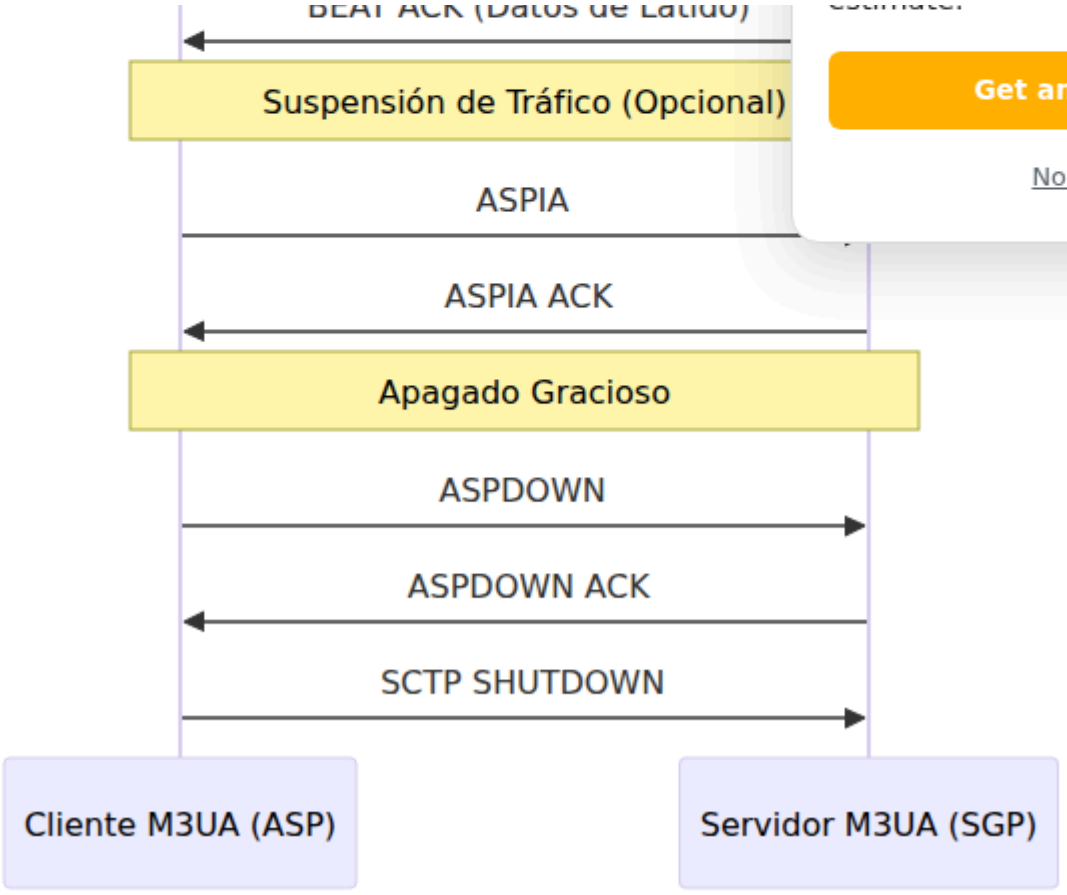
Entendiendo M2PA

M2PA (Capa de Adaptación Peer-to-Peer de MTP2) es un protocolo definido en el RFC 4165 que proporciona transporte de mensajes MTP3 punto a punto sobre SCTP. A diferencia de M3UA que utiliza una arquitectura ASP/SGP, M2PA proporciona enlaces peer-to-peer similares a los enlaces tradicionales TDM SS7.

Estados de Enlace M2PA

Los enlaces M2PA progresan a través de varios estados durante el establecimiento:


Cliente M3UA (ASP)
Servidor M3UA (SGP)
Configuración de Asociación SCTP
SCTP INIT
SCTP INIT ACK
SCTP COOKIE ECHO
SCTP COOKIE ACK
OmniCore 5GC
OmniCall
OmniRAN
OmniCharge
Platform
Español
ASPUP
ASPUP ACK
ASPAC
ASPAC ACK
Gestión de Claves de Enrutamiento (Opcional)
REG REQ (Clave de Enrutamiento)
REG RSP (Resultado de Registro)
Activo - Transferencia de Datos
DATA (Datos del Protocolo M3UA)
DATA (Datos del Protocolo M3UA)
Latido (Periódico)
BEAT (Datos de Latido)



Descripciones de Estado:

- **DOWN** - Sin conexión SCTP, enlace inactivo
- **CONNECTING** - Asociación SCTP en progreso
- **ALIGNMENT** - Mensajes de Estado de Enlace intercambiados (~1 segundo)
- **PROVING** - Período de prueba de enlace, probando la integridad del enlace (~2 segundos)
- **READY** - Enlace operativo, listo para la transferencia de datos de usuario MTP3
- **ALIGNMENT (reentrada)** - Cambio de estado de enlace requiere re-alineación

Progresión del Estado del Enlace:

1. **Conexión SCTP**: Establece la asociación SCTP (DOWN → CONNECTING)
2. **Alineación**: Intercambia mensajes de Estado de Enlace para sincronizar (CONNECTING → ALIGNMENT)
3. **Prueba**: Prueba la fiabilidad del enlace y la sincronización de números de secuencia (ALIGNMENT → PROVING)

4. **Listo**: El enlace se vuelve operativo para la transferencia de datos (PROVING → READY)

Flujo de Mensajes M2PA


Par M2PA 1 (PC: 100)
Par M2PA 2 (PC: 200)
Configuración de Asociación SCTP
SCTP INIT
SCTP INIT ACK
SCTP COOKIE ECHO
SCTP COOKIE ACK
Alineación de Enlace M2PA
Estado de Enlace (ALINEACIÓN)
Estado de Enlace (ALINEACIÓN)
Prueba de Enlace M2PA (~2s)
Estado de Enlace (PRUEBA)
Estado de Enlace (PRUEBA)
Enlace Listo
Estado de Enlace (LISTO)
Estado de Enlace (LISTO)
Listo - Transferencia de Datos MTP3
Datos de Usuario (Mensaje MTP3, BSN/FSN)
Datos de Usuario (Mensaje MTP3, BSN/FSN)
Datos de Usuario (Mensaje MTP3, BSN/FSN)
Datos de Usuario (Mensaje MTP3, BSN/FSN)
Apagado Graceful
Estado de Enlace (FUERA DE SERVICIO)



Monitoreo de Pares M2PA a través de la Interfaz Web

La Interfaz Web proporciona monitoreo en tiempo real de las conexiones de pares M2PA.

Accediendo a la Página de Gestión de Enrutamiento:

1. Navegar a la página de inicio de la Interfaz Web
2. Hacer clic en "Gestión de Enrutamiento" en el menú de navegación
3. Ver la tabla "Pares M3UA/M2PA"

Tabla de Pares M2PA:

Columna	Descripción
ID de Par	Identificador único del par
Nombre	Nombre del par (por ejemplo, `M2PA_Link_STP_A`)
Protocolo	Muestra "M2PA" en verde
Código de Punto	Código de punto local
PC Adyacente	Código de punto del par adyacente
Local	IP:Puerto local (típicamente puerto 3565)
Remoto	IP:Puerto remoto
Estado	Estado del enlace (por ejemplo, LISTO, ALINEACIÓN, DOWN)

Indicadores de Estado:

- **LISTO (Verde)** - El enlace está operativo y pasando tráfico
- **ALINEACIÓN (Amarillo)** - El enlace se está alineando, aún no está listo
- **PRUEBA (Amarillo)** - El enlace está en estado de prueba
- **DOWN (Rojo)** - El enlace está caído o no disponible

Solución de Problemas de Conexiones M2PA

Problema: Enlace atascado en ALINEACIÓN

Síntomas:

- El estado del enlace muestra ALINEACIÓN durante un período prolongado
- Sin progresión a PRUEBA o LISTO

Verificaciones:

1. Verificar que ambos lados estén configurados con los códigos de punto correctos
2. Comprobar que el firewall SCTP permita el protocolo 132
3. Verificar que `point_code` y `adjacent_point_code` estén configurados correctamente
4. Revisar los registros de la aplicación en busca de errores en los mensajes de Estado de Enlace
5. Asegurarse de que el par remoto también esté en estado de ALINEACIÓN

Problema: Enlace atascado en PRUEBA

Síntomas:

- El enlace alcanza PRUEBA pero no transiciona a LISTO
- El período de prueba excede 2-3 segundos

Verificaciones:

1. Verificar la estabilidad de la red (sin pérdida de paquetes)
2. Comprobar errores de asociación SCTP
3. Revisar registros en busca de desajustes en los números de secuencia
4. Asegurarse de que el par remoto también esté en estado de PRUEBA
5. Verificar que el multihoming SCTP no esté causando problemas de enrutamiento

Problema: Fluctuación del Enlace (DOWN ↔ LISTO)

Síntomas:

- El enlace cicla repetidamente entre LISTO y DOWN
- Re-alineaciones frecuentes

Verificaciones:

1. Comprobar la estabilidad de la conectividad de red
2. Verificar la configuración de los latidos SCTP
3. Revisar la configuración de tiempo de espera de sesiones del firewall
4. Comprobar problemas de MTU/fragmentación

5. Verificar que no haya direcciones IP duplicadas

Problema: No fluye datos

Síntomas:

- El estado del enlace muestra LISTO
- No se están transfiriendo mensajes MTP3

Verificaciones:

1. Verificar que las tablas de enrutamiento incluyan rutas a este par
2. Comprobar que el enrutamiento del código de punto MTP3 esté configurado
3. Revisar que los valores DPC en los mensajes coincidan con las rutas esperadas
4. Comprobar la página `/events` en busca de errores de enrutamiento
5. Verificar que los números de secuencia (BSN/FSN) estén incrementando

---

Documentación Relacionada

- ← Volver a la Documentación Principal
- Guía de Funciones Comunes - Interfaz Web, API, Monitoreo
- Guía del Cliente MAP - Enviando solicitudes MAP
- Guía del Centro de SMS - Entrega de SMS
- Referencia Técnica - Especificaciones del protocolo

OmniSS7 por Omnitouch Network Services

Guía del Gateway USSD

← Volver a la Documentación Principal

Esta guía cubre el **Gateway USSD** de OmniSS7, que conecta diálogos USSD de SS7/MAP a callbacks HTTP/JSON, permitiendo a los desarrolladores de terceros construir aplicaciones USSD con un simple endpoint HTTP.

Tabla de Contenidos

1. Descripción General
2. Arquitectura
3. Habilitando el Gateway USSD
4. Configuración
5. Protocolo de Callback HTTP
6. USSD Originado en la Red (API Push)
7. Ciclo de Vida de la Sesión
8. Manejo de Errores
9. Métricas y Monitoreo
10. Servidor de Callback de Ejemplo
11. Solución de Problemas

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` de MAP (código de operación 59). Reenvía la solicitud a tu callback HTTP. Luego, retransmite tu respuesta 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` de MAP (código de operación 60) a través de SS7. Luego, dirige 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 de 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

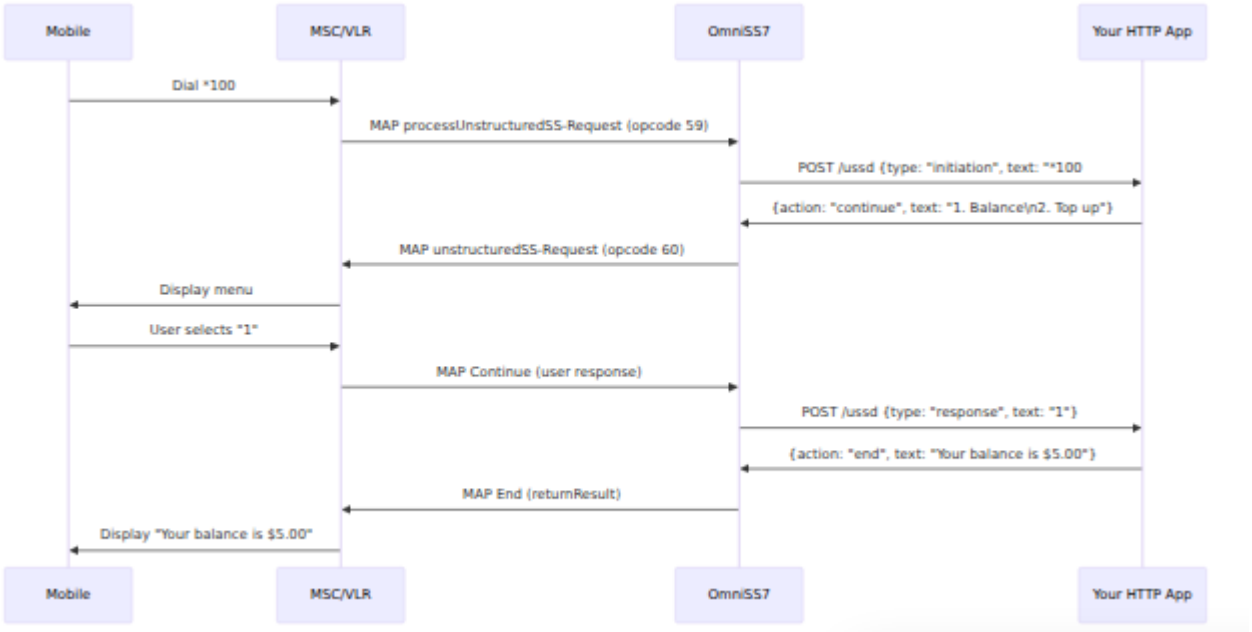
Referencias de 3GPP

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

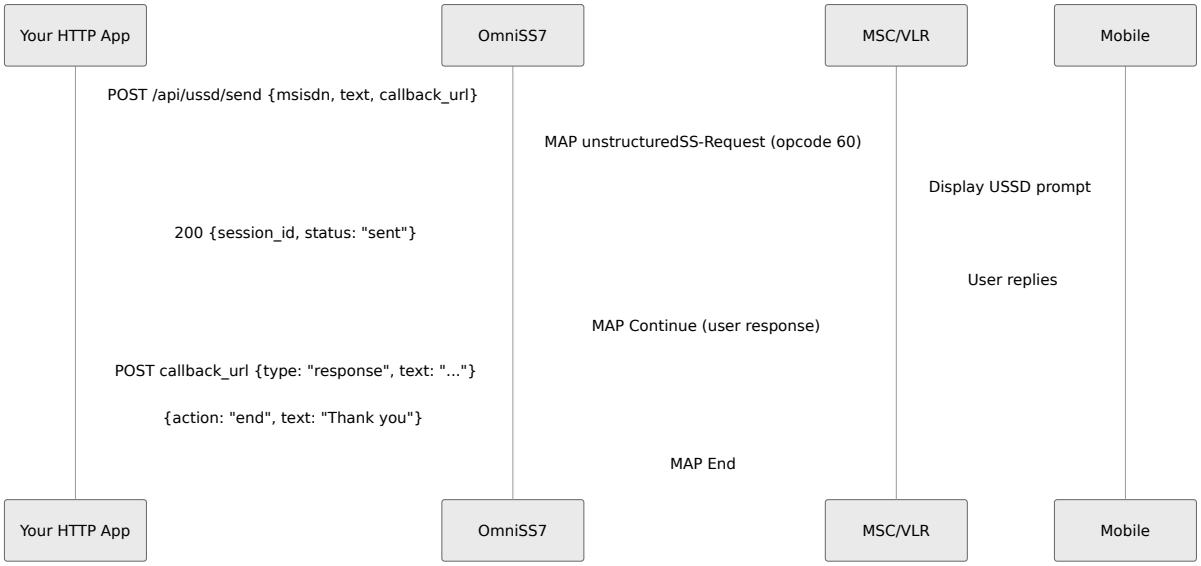
---

Arquitectura

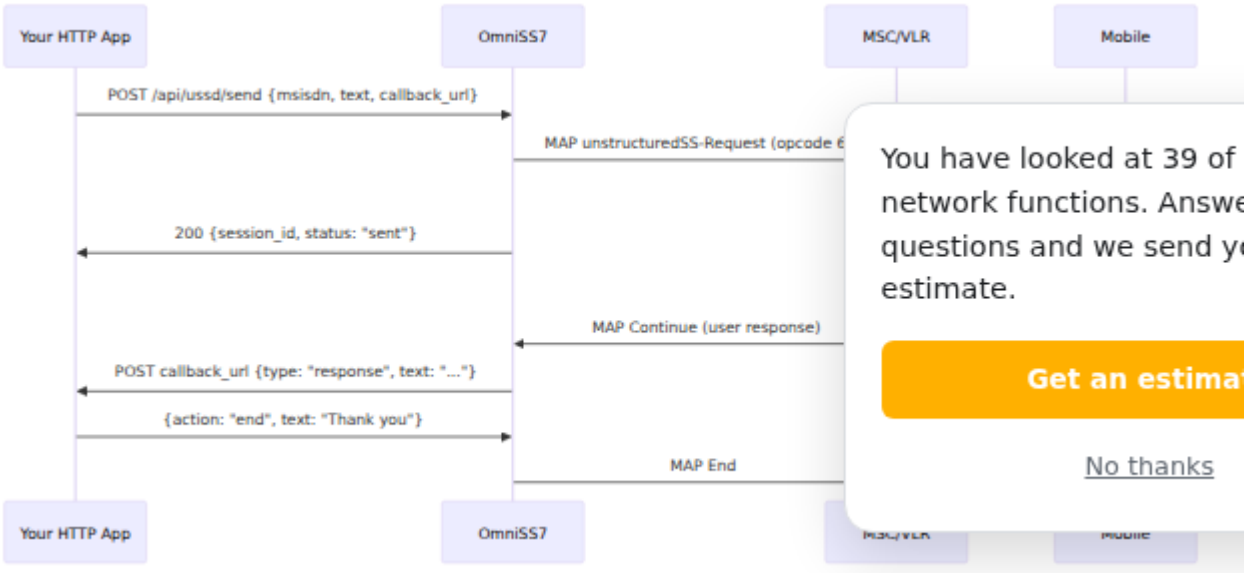
Flujo Originado en Móvil (Entrante)



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 característica.

```
config :omniss7,
  map_client_enabled: true,
  ussd_gateway_enabled: true
```

El gateway también requiere una conexión M3UA funcional (ver Guía del Cliente MAP para la configuración de M3UA).

Configuración

Parámetros del Gateway USSD

```
config :omniss7,
  ussd_gateway_enabled: true,
  ussd_gateway: %{
    # Enrutamiento de código corto - 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)

    # Configuraciones de callback HTTP
    http_timeout_ms: 5_000,         # Tiempo de espera para HTTP
POST a tu aplicación (5 segundos)

    # Límites de texto
    max_text_length: 182            # Máximo de 7 bits de 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): seleccionado
2. `"*10"` coincide (longitud 3): omitido, más corto
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 (inicio 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 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 a 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 a 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": "solicitud inválida", "required": ["msisdn", "text", "callback_url"]}`	Falta uno o más campos requeridos, o el cuerpo no es un JSON válido
500	`{"error": "{:gsm7_encode_failed, ...}"}`	El texto contiene caracteres no en el alfabeto GSM de 7 bits. La falla de codificación se presenta a través de la rama de error genérico, por lo que el estado es **500**, no 400.
500	`{"error": "..."}`	Cualquier otra falla de envío (por ejemplo, conectividad M3UA). El campo `error` lleva la razón interna inspeccionada.
503	`{"error": "gateway USSD no habilitado"}`	`ussd_gateway_enabled` es `false`

> **Nota sobre errores de validación vs. envío:** Los problemas de validación de campos (claves faltantes, JSON mal formado) devuelven **400** **`solicitud inválida`**. Una vez que la validación pasa, cualquier falla del camino de envío (incluidas las fallas de codificación GSM de 7 bits) se informa como **500** con la razón inspeccionada en el campo `error`.

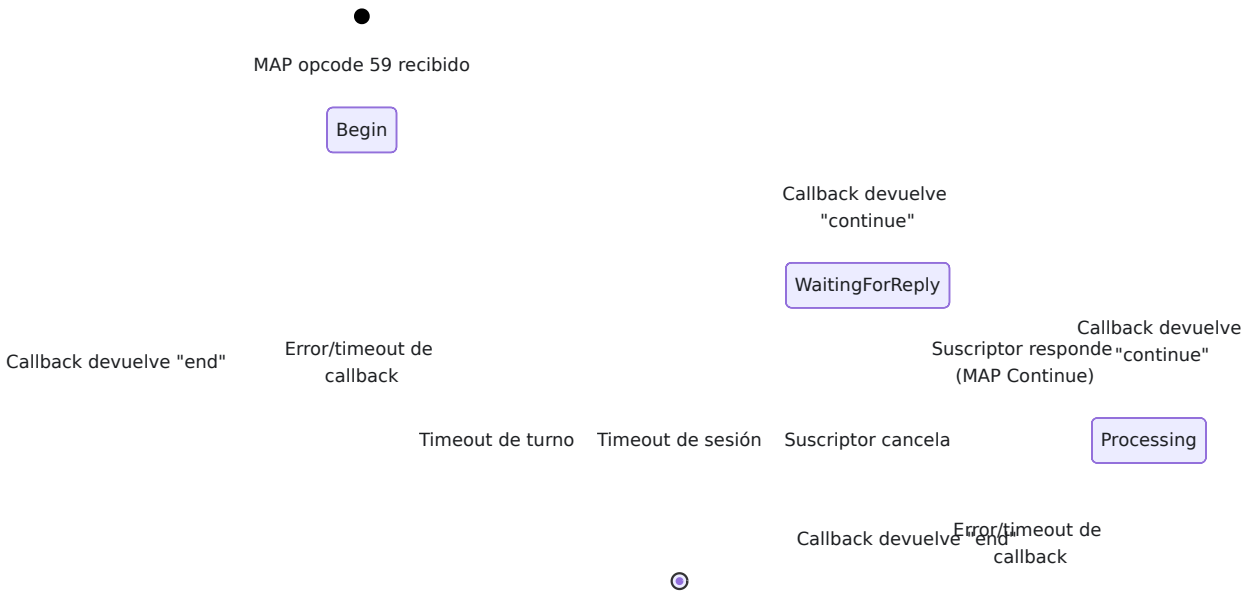
Ejemplo de cURL

```
curl -X POST http://localhost:9090/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 timeout

Comportamiento de Timeout

Timeout	Por Defecto	Efecto
Timeout de turno	30 segundos	Si el suscriptor no responde dentro de esta ventana, la sesión se termina con un error MAP.
Timeout de sesión	3 minutos	Duración total de la sesión. Termina la sesión independientemente de la actividad.
Timeout 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 fallas 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 timeout de red.

Escenario	Acción del Gateway	Código de Error MAP
Timeout 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)
Timeout 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 timeout de red	N/A (salida de proceso)
Gateway USSD no habilitado	Retornar Facility Not Supported	21 (facilityNotSupported)

Métricas y Monitoreo

El Gateway USSD expone métricas de Prometheus en el endpoint estándar `/metrics` (puerto 9090).

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**: Declarado como un medidor para el número de sesiones USSD activas.

> **Advertencia:** Este medidor está **declarado pero nunca actualizado** en la construcción actual: siempre lee `0`. No confíes en él para contar sesiones en vivo. Rastrea la actividad de la sesión a través de `ussd_requests_total` en su lugar.

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 (entrante vs saliente)
rate(ussd_requests_total[5m])

# 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. Consultar 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'Compraste ${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. Consultar 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 catch-all)
3. Verifica el estado del par M3UA en la interfaz web (página de Pares)

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
- Firewall bloqueando 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

Sesiones Expirando 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 una respuesta **500** de `/api/ussd/send` cuyo campo `error` contiene `{:gsm7_encode_failed, ...}`.

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 de caracteres básicos de GSM: letras ASCII, dígitos, puntuación común y algunos caracteres griegos/nórdicos

- Consulta 3GPP TS 23.038 Sección 6.2.1 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).

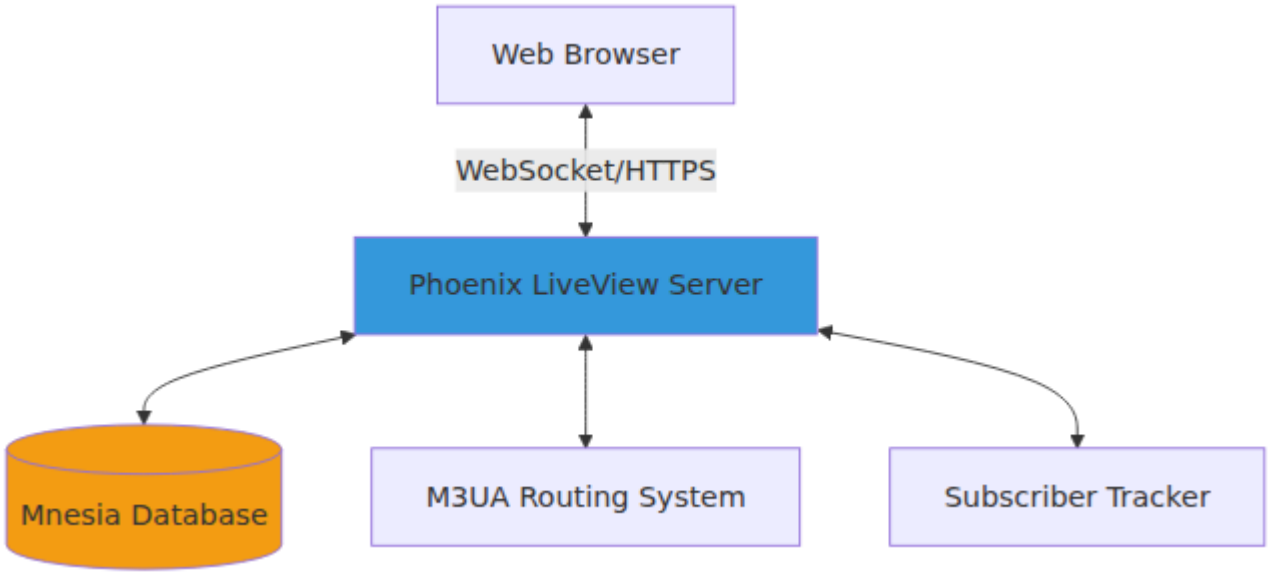
Tabla de Contenidos

1. Descripción General
2. Accediendo a la Interfaz Web
3. Página de Gestión de Enrutamiento
4. Página de Suscriptores Activos
5. Operaciones Comunes
6. Comportamiento de Auto-Actualización

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 esté activo (STP, HLR o SMSc).

Arquitectura de la Interfaz Web



Configuración del Servidor

Parámetro	Valor
Protocolo	HTTPS (`force_ssl` habilitado)
Puerto	`8087` (configurado bajo `config :control_panel, ControlPanelWeb.Endpoint` → `https: [port: 8087]`)
IP Predeterminada	`0.0.0.0` (escucha en todas las interfaces)
Certificado	`priv/cert/omnitouch.crt`
Clave privada	`priv/cert/omnitouch.pem`
Adaptador	Bandit (Phoenix)

URL de Acceso: `https://[server-ip]:8087`

Accediendo 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 `8087` esté abierto para tráfico HTTPS

Páginas Disponibles por Modo

La etiqueta de la página que se muestra aquí es exactamente la etiqueta registrada en `config/config.exs` (`use_builtin_pages` / `use_additional_pages`). Las columnas de modo indican el modo operativo (STP/HLR/SMSc) en el que una página es típicamente relevante; la configuración en tiempo de ejecución filtra la lista de páginas por implementación.

Página (ruta)	Etiqueta	Modo STP	Modo HLR	Modo SMSc	Descrip
`/application`	**Recursos**	□	□	□	Recursos sistema y monitore
`/configuration`	**Configuración**	□	□	□	Visor de configura
`/license`	**Licencia**	□	□	□	Estado d licencia
`/events`	**Eventos SS7**	□	□	□	Registro eventos y captura d mensajes SCCP
`/logs`	**Registros del Sistema**	□	□	□	Visor de registros aplicació vivo
`/client`	**Cliente SS7**	□	□	□	Pruebas operació MAP mar
`/m3ua`	**Pares**	□	□	□	Estado d conexión pares / A M3UA
`/sctp`	**Conexiones SCTP**	□	□	□	Estado d asociació SCTP en

Página (ruta)	Etiqueta	Modo STP	Modo HLR	Modo SMSc	Descripc
`/routing`	**Enrutamiento**	✅	✅	✅	Gestión d tabla de enrutami M3UA
`/routing_test`	**Prueba de Enrutamiento**	✅	✅	✅	Pruebas y validació rutas
`/hlr_links`	**Enlaces HLR**	✅	✅	✅	Estado d API HLR y gestión d suscripto
`/subscribers`	**Suscriptores Activos**	✅	✅	✅	Seguimie de ubicac de suscripto en tiemp real (HLR
`/smsc_links`	**Enlaces SMSc**	✅	✅	✅	Estado d API SMSc gestión d colas
`/smsc_subscribers`	**Suscriptores SMSc**	✅	✅	✅	Seguimie de suscripto en tiemp real (SMS
`/camel_sessions`	**Sesiones CAMEL**	✅	✅	✅	Monitore diálogo

Página (ruta)	Etiqueta	Modo STP	Modo HLR	Modo SMSc	Descripc
					CAMEL/C en vivo
`/camel_request`	**Constructor de Solicitudes CAMEL**	□	□	□	Construc de solicitude CAP interactiv
`/msisdn_imsi_test`	**Prueba MSISDN/IMSI**	□	□	□	Búsqued prueba d conversio MSISDN ‹ IMSI

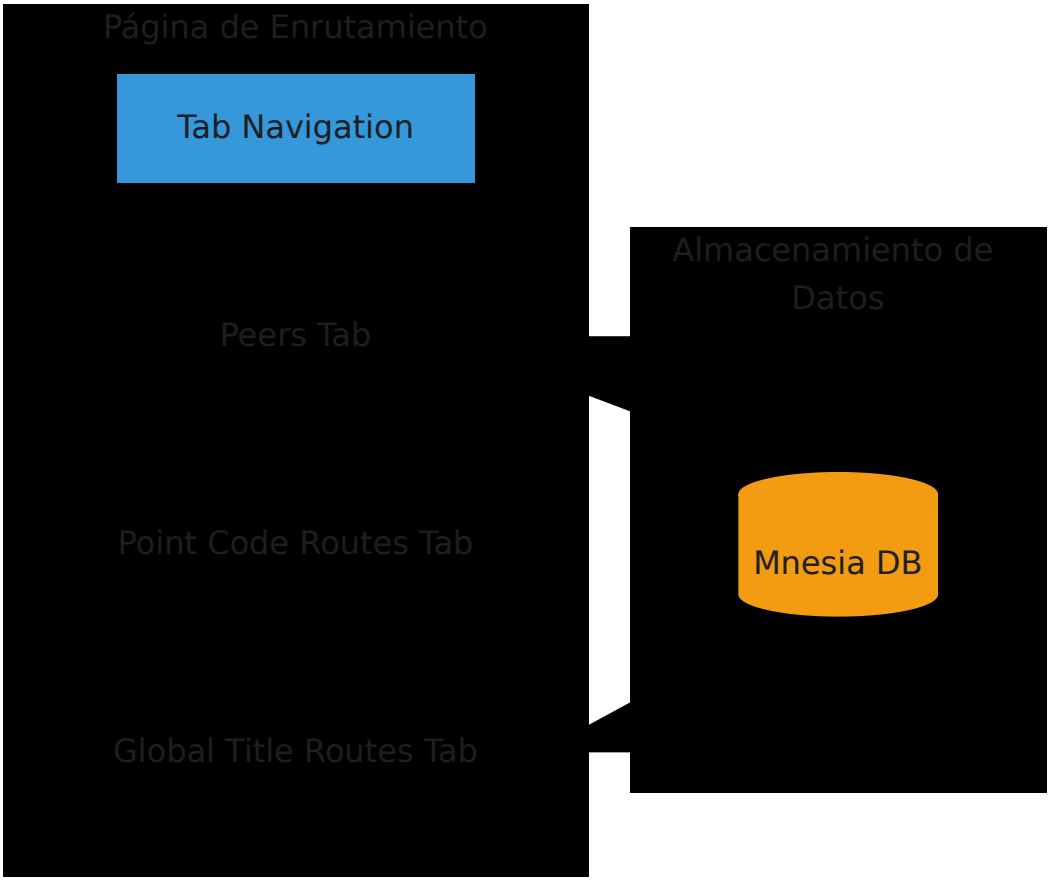
> La página M3UA está etiquetada como **"Pares"** en la navegación (ruta `/m3ua`).

Página de 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 Pares

Gestionar conexiones de pares M3UA (otros STPs, HLRs, MSCs, SMSCs).

Columnas de la Tabla de Pares

Columna	Descripción	Ejemplo
ID	Identificador único del par	`1`
Nombre	Nombre del par legible por humanos	`"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 Editar/Eliminar	-

Agregar un Par

1. **Haga clic** en la pestaña de Pares
2. **Complete** los campos del formulario:
   - **ID del Par**: Generado automáticamente si se deja vacío
   - **Nombre del Par**: Nombre descriptivo (requerido)
   - **Rol**: Seleccione `client`, `server` o `stp`
   - **Point Code**: Código de punto SS7 (requerido)
   - **IP Local**: Dirección IP de su sistema
   - **Puerto Local**: 0 para asignación dinámica de puerto
   - **IP Remota**: Dirección IP del par
   - **Puerto Remoto**: Puerto del par (típicamente 2905)
   - **Contexto de Enrutamiento**: ID de contexto de enrutamiento M3UA
   - **Indicador de Red**: `international` o `national`
3. **Haga clic** en "Agregar Par"

Persistencia: El par se guarda inmediatamente en Mnesia y sobrevive al reinicio.

Editar un Par

1. **Haga clic** en el botón "Editar" en la fila del par
2. **Modifique** los campos del formulario según sea necesario
3. **Haga clic** en "Actualizar Par"

Nota: Si cambia el ID del Par, el par antiguo se elimina y se crea uno nuevo.

Eliminar un Par

1. **Haga clic** en el botón "Eliminar" 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
PC de Destino	Código de punto objetivo (formato zona.area.id)	`1.2.3` (100)
Máscara	Máscara de subred para coincidencia de PC	`/14` (exacto), `/8` (rango)
ID del Par	Par objetivo para esta ruta	`1`
Nombre del Par	Nombre del par objetivo	`"STP_West"`
Prioridad	Prioridad de la ruta (1 = más alta)	`1`
Red	Indicador de red	`international`
Acciones	Botones Editar/Eliminar	-

Agregar una Ruta de Código de Punto

1. **Haga clic** en la pestaña "Rutas de Código de Punto"
2. **Complete** los campos del formulario:
   - **Código de Punto de Destino**: Ingrese como `zona.area.id` (por ejemplo, `1.2.3`) o entero (0-16383)
   - **Máscara**: Seleccione máscara `/14` para coincidencia exacta, valores más bajos para rangos
   - **ID del Par**: Seleccione el par objetivo del menú desplegable
   - **Prioridad**: Ingrese prioridad (1 = más alta, predeterminado)
   - **Indicador de Red**: Seleccione `international` o `national`
3. **Haga clic** en "Agregar Ruta"

Formato de Código de Punto: Puede ingresar códigos de punto en dos formatos:

- **Formato 3-8-3**: `zona.area.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 un 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 Títulos Globales

Configure reglas de enrutamiento basadas en direcciones de Títulos Globales SCCP.

Requisito: El enrutamiento de Títulos Globales debe estar habilitado en la configuración:

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

Columnas de la Tabla de Rutas

Columna	Descripción	Ejemplo
Prefijo GT	Prefijo GT del partido llamado (vacío = respaldo)	`"1234"`, `""`
SSN de Origen	Coincidir en SSN del partido llamado (opcional)	`6` (HLR), `any`
ID del Par	Par objetivo	`1`
Par	Nombre del par	`"HLR_West (1)"`
SSN de Destino	Reescribir SSN al reenviar (opcional)	`6`, `preserve`
Prioridad	Prioridad de la ruta	`1`
Descripción	Descripción de la ruta	`"Números de EE. UU."`
Acciones	Botones Editar/Eliminar	-

Agregar una Ruta de Título Global

1. **Haga clic** en la pestaña "Rutas de Títulos Globales"
2. **Complete** los campos del formulario:
   - **Prefijo GT**: Deje vacío para la ruta de respaldo, o ingrese dígitos (por ejemplo, `"1234"`)
   - **SSN de Origen**: Opcional - filtrar por SSN del partido llamado
   - **ID del Par**: Seleccione el par objetivo
   - **SSN de Destino**: Opcional - reescribir SSN al reenviar

    - **Prioridad**: Prioridad de la ruta (1 = más alta)
    - **Descripción**: Descripción legible por humanos
3. **Haga clic** en "Agregar Ruta"

Rutas de Respaldo: Si el Prefijo GT 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 del Hogar)
7	VLR (Registro de Ubicación del Visitante)
8	MSC (Centro de Conmutación Móvil)
9	EIR (Registro de Identidad del Equipo)
10	AUC (Centro de Autenticación)
142	RANAP
145	gsmSCF (Función de Control de Servicio)
146	SGSN

Reescritura de SSN

- **SSN de Origen**: Coincidir en el SSN del Partido Llamado en mensajes entrantes
- **SSN de Destino**: Si se establece, reescribe el SSN del Partido Llamado al reenviar
    - Vacío = preservar 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:

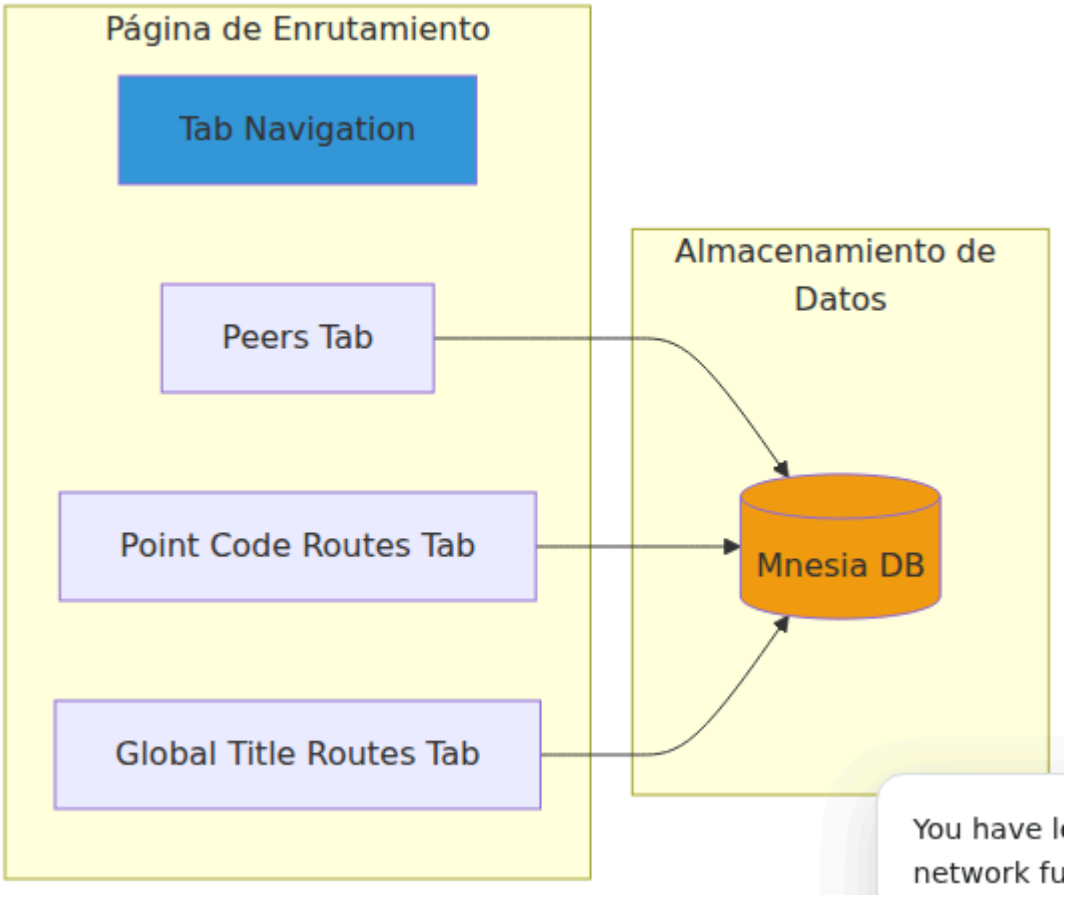
1. **Más Específico Primero**: Valores de máscara más altos (más específicos) tienen prioridad
2. **Campo de Prioridad**: Números de prioridad más bajos enrutan primero (1 = mayor prioridad)
3. **Estado del Par**: Solo se utilizan rutas a pares `active`

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, una tarjeta de resumen muestra:

- **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

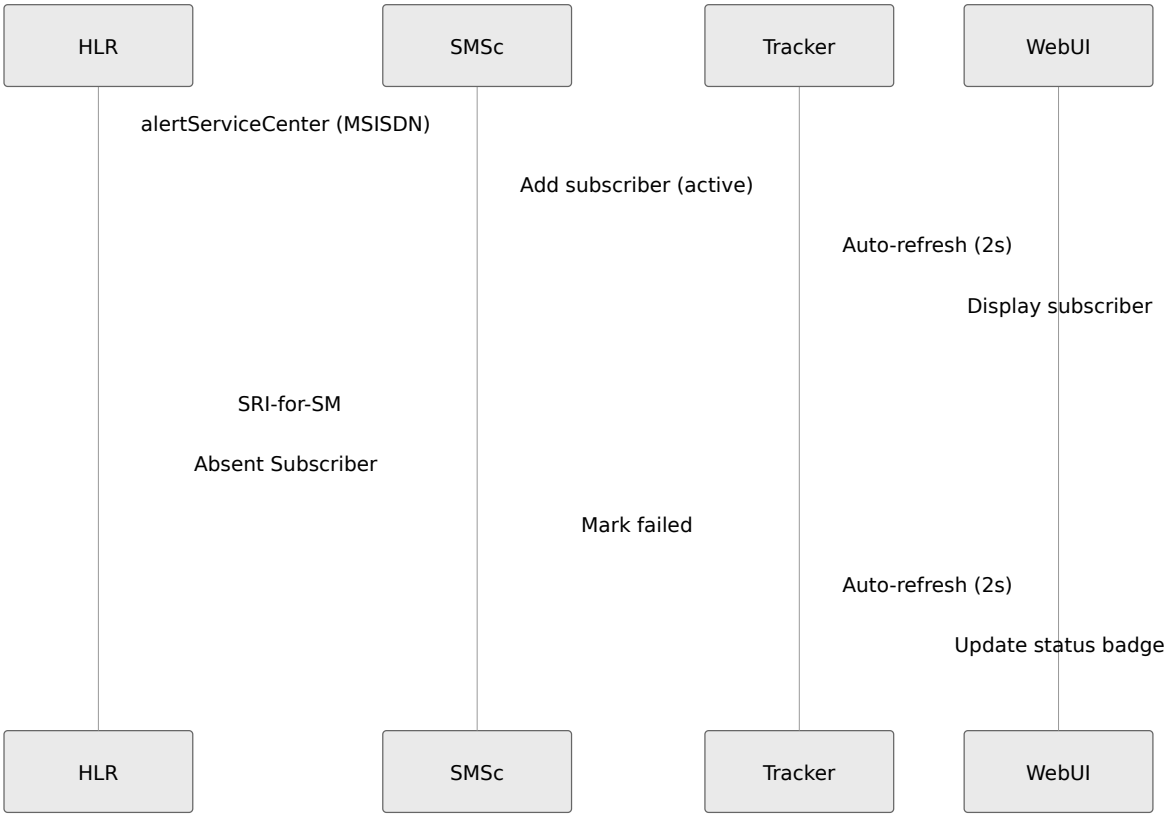
La página se actualiza automáticamente cada **2 segundos** para mostrar actualizaciones en tiempo real de los suscriptores.

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	`Activo` o `Fallido` (codificado por color)	`● Activo`
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

- ● **Activo** (Verde): El suscriptor es accesible, última alertServiceCenter recibida con éxito
- ○ **Fallido** (Rojo): Último intento de entrega fallido (error SRI-for-SM o suscriptor ausente)

Resumen de Estadísticas

Cuando hay suscriptores presentes, una tarjeta de resumen muestra:

- **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 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. Utilice 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**: Utilice la API REST para acceso programático
2. **Opción 2**: Edite `config/runtime.exs` y reinicie la aplicación
3. **Opción 3**: Utilice la Interfaz Web para cambios individuales de rutas

Exportar/Importar

Nota: La Interfaz Web no admite actualmente 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 hacer una copia de seguridad de las rutas:

1. **Mnesia**: Haga una copia de seguridad del 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 poco frecuentes
Suscriptores Activos	2 segundos	El estado de los suscriptores cambia con frecuencia
Estado 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 `priv/cert/omnitouch.pem` estén presentes
2. **Verifique el Puerto 8087**: Revise las reglas del cortafuegos para permitir tráfico HTTPS en el puerto `8087`
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 Persistiendo

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 para 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 una 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 reiniciar
 Monitoreo de Estado - Estado de conexión y estado de pares en vivo

Para operaciones avanzadas o automatización, consulte la Guía API.

Ejemplos Avanzados de Enrutamiento de Títulos Globales

← Volver a la Documentación Principal

Este documento proporciona ejemplos prácticos de uso del Tipo de Traducción (TT), Indicador de Plan de Numeración (NPI) y Indicador de Naturaleza de la Dirección (NAI) para escenarios avanzados de enrutamiento de GT.

Tabla de Contenidos

1. Conceptos Básicos
2. Ejemplos de Tipo de Traducción (TT)
3. Ejemplos de Indicador de Plan de Numeración (NPI)
4. Ejemplos de Naturaleza de la Dirección (NAI)
5. Ejemplos Combinados
6. Casos de Uso del Mundo Real

Conceptos Básicos

Referencia de Valores de Indicadores

Tipo de Traducción (TT):

- `0` = Desconocido
- `1` = Internacional
- `2` = Nacional
- `3` = Específico de la Red

Indicador de Plan de Numeraci��n (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 la Dirección (NAI):

- `0` = Desconocido
- `1` = Número de Suscriptor
- `2` = Reservado para Uso Nacional
- `3` = Número Nacional Significativo
- `4` = Número Internacional

Coincidencia vs Transformación

- **Indicadores de origen** (source_tt, source_npi, source_nai): Utilizados para **coincidir** mensajes entrantes
- **Indicadores de destino** (dest_tt, dest_npi, dest_nai): Utilizados para **transformar** valores al reenviar
- Deje los campos vacíos (`nil`) para coincidencia comodín o para preservar valores originales

---

Ejemplos de Tipo de Traducción (TT)

Ejemplo 1: Conversión de TT Desconocido a Internacional

Escenario: La red asociada envía todos los GTs con TT=0 (Desconocido), pero su HLR de downstream espera TT=1 (Internacional).

```
%{
  gt_prefix: "44",
  peer_id: 1,
  priority: 1,
  description: "Reino Unido: Convertir TT Desconocido a
Internacional",
  source_tt: 0,      # Coincidir TT entrante=0
  dest_tt: 1         # Transformar a TT=1
}
```

Flujo de tráfico:

- Entrante: GT=441234567890, TT=0
- Enrutamiento: Coincide gt_prefix="44", source_tt=0
- Saliente: GT=441234567890, TT=1 (transformado)

Ejemplo 2: Rutas Separadas por Tipo de Traducción

Escenario: Enrutar diferentes valores de TT a diferentes redes.

```
# Enrutar TT=0 a la Red A
%{
  gt_prefix: "33",
  peer_id: 1,
  priority: 1,
  description: "Números franceses con TT=0 a la Red A",
  source_tt: 0
},

# Enrutar TT=2 a la Red B
%{
  gt_prefix: "33",
  peer_id: 2,
  priority: 1,
  description: "Números franceses con TT=2 a la Red B",
  source_tt: 2
}
```

Flujo de tráfico:

- GT="33123456789", TT=0 → Enruta a peer 1
- GT="33123456789", TT=2 → Enruta a peer 2

Ejemplos de Indicador de Plan de Numeración (NPI)

Ejemplo 1: Conversión de NPI Móvil a ISDN

Escenario: La red móvil utiliza NPI=6 (E.212), PSTN espera NPI=1 (E.164).

```
%{
  gt_prefix: "49",
  peer_id: 1,
  priority: 1,
  description: "Conversión móvil alemán→ISDN",
  source_npi: 6,     # Coincidir NPI=6 (Móvil/E.212)
  dest_npi: 1        # Transformar a NPI=1 (ISDN/E.164)
}
```

Caso de uso: Interconectar redes móviles y fijas con diferentes planes de numeración.

Ejemplo 2: Enrutamiento Basado en el Plan de Numeración

Escenario: Separar el tráfico de red de datos del tráfico de voz.

```
# Enrutar tráfico de red de datos (NPI=3)
%{
  gt_prefix: "1",
  peer_id: 1,
  priority: 1,
  description: "Tráfico de red de datos",
  source_npi: 3      # Coincidir NPI=3 (Datos/X.121)
},

# Enrutar tráfico de red de voz (NPI=1)
%{
  gt_prefix: "1",
  peer_id: 2,
  priority: 1,
  description: "Tráfico de red de voz",
  source_npi: 1      # Coincidir NPI=1 (ISDN)
}
```

Ejemplos de Naturaleza de la Dirección (NAI)

Ejemplo 1: Conversión de NAI Internacional a Nacional

Escenario: Formato internacional entrante (NAI=4), convertir a formato nacional (NAI=3) para enrutamiento nacional.

```
%{
  gt_prefix: "1",
  peer_id: 1,
  priority: 1,
  description: "EE. UU.: Formato Internacional→Nacional",
  source_nai: 4,     # Coincidir NAI=4 (Internacional)
  dest_nai: 3        # Transformar a NAI=3 (Nacional)
}
```

Flujo de tráfico:

- Entrante: GT=12125551234, NAI=4 (formato internacional)
- Saliente: GT=12125551234, NAI=3 (formato nacional)

Ejemplo 2: Enrutamiento de NAI de Suscriptor vs Red

Escenario: Enrutar según si es un número de suscriptor o un número de red.

```
# Enrutar números de suscriptor a HLR
%{
  gt_prefix: "44",
  peer_id: 1,
  priority: 1,
  description: "Números de suscriptor del Reino Unido a HLR",
  source_nai: 1      # Coincidir NAI=1 (Número de Suscriptor)
},

# Enrutar números de red a MSC
%{
  gt_prefix: "44",
  peer_id: 2,
  priority: 1,
  description: "Direcciones de red del Reino Unido a MSC",
  source_nai: 3      # Coincidir NAI=3 (Número Nacional
Significativo)
}
```

Ejemplos Combinados

Ejemplo 1: Normalización del Tráfico SMS

Escenario: Normalizar todo el tráfico SMS de diversas fuentes antes de enrutar a HLR.

```
%{
  gt_prefix: "86",
  source_ssn: 8,           # Coincidir SSN=8 (SMSC)
  peer_id: 1,
  dest_ssn: 6,             # Reescribir a SSN=6 (HLR)
  priority: 1,
  description: "Normalización de SMS en China",
  source_tt: 0,            # Coincidir TT=0 (Desconocido)
  dest_tt: 2,              # Transformar a TT=2 (Nacional)
  source_npi: 6,           # Coincidir NPI=6 (Móvil)
  dest_npi: 1,             # Transformar a NPI=1 (ISDN)
  source_nai: 4,           # Coincidir NAI=4 (Internacional)
  dest_nai: 3              # Transformar a NAI=3 (Nacional)
}
```

Qué hace esto:

- Coincide el tráfico SMS (SSN=8) para números chinos (prefijo GT "86")
- Normaliza todos los indicadores a valores estándar
- Reescribe el SSN de destino de SMSC (8) a HLR (6)
- Perfecto para escenarios de interconexión con uso inconsistente de indicadores

Ejemplo 2: Enrutamiento Multi-Operador con Conversión de Formato

Escenario: Enrutar a diferentes operadores según TT, convirtiendo formatos según sea necesario.

```
# Operador A: Espera TT Nacional y NPI ISDN
%{
  gt_prefix: "44",
  peer_id: 1,
  priority: 1,
  description: "Operador A del Reino Unido: Normalizar a
Nacional/ISDN",
  source_tt: 0,           # Coincidir cualquier TT=0
  dest_tt: 2,             # Convertir a TT=2 (Nacional)
  source_npi: 6,          # Coincidir NPI Móvil
  dest_npi: 1             # Convertir a NPI ISDN
},

# Operador B: Espera TT Internacional, preservar NPI
%{
  gt_prefix: "44",
  peer_id: 2,
  priority: 2,            # Prioridad más baja (número más alto)
  description: "Operador B del Reino Unido: Convertir a TT
Internacional",
  source_tt: 2,           # Coincidir TT Nacional
  dest_tt: 1              # Convertir a TT Internacional
  # NPI dejado vacío = preservar original
}
```

Casos de Uso del Mundo Real

Caso de Uso 1: Integración de Sistemas Legados

Problema: Integrar una red moderna con un sistema legado que utiliza diferentes convenciones de indicadores.

Solución: Transformar todos los indicadores al formato legado.

```
%{
  gt_prefix: "555",
  peer_id: 1,
  priority: 1,
  description: "Normalización de HLR legado",
  # Coincidir cualquier formato moderno
  source_tt: 1,            # Moderno: Internacional
  dest_tt: 0,              # Legado: Desconocido
  source_npi: 1,           # Moderno: ISDN
  dest_npi: 0,             # Legado: Desconocido
  source_nai: 4,           # Moderno: Internacional
  dest_nai: 0              # Legado: Desconocido
}
```

Caso de Uso 2: Estandarización de Interconexión Internacional

Problema: Múltiples socios internacionales envían tráfico con indicadores inconsistentes.

Solución: Normalizar todo el tráfico entrante a un formato consistente.

```
# El socio A envía TT=0, convertir a TT estándar=1
%{
  gt_prefix: "1",
  peer_id: 1,
  priority: 1,
  description: "Normalización del socio A de EE. UU.",
  source_tt: 0,
  dest_tt: 1,
  dest_npi: 1,             # Asegurar NPI ISDN
  dest_nai: 4              # Asegurar NAI Internacional
},

# El socio B envía TT=3 (Específico de la Red), convertir a TT=1
%{
  gt_prefix: "44",
  peer_id: 2,
  priority: 1,
  description: "Normalización del socio B del Reino Unido",
  source_tt: 3,
  dest_tt: 1,
  dest_npi: 1,
  dest_nai: 4
}
```

Caso de Uso 3: Gestión del Tráfico de Roaming

Problema: La red de origen y los socios de roaming utilizan diferentes formatos de dirección.

Solución: Convertir el tráfico de roaming al formato de la red de origen.

```
# Roaming entrante: Convertir al formato de la red de origen
%{
  gt_prefix: "12025",      # Área de Washington DC
  peer_id: 1,
  priority: 1,
  description: "Normalización de roaming entrante",
  source_nai: 4,           # Internacional del socio de roaming
  dest_nai: 3,             # Nacional para la red de origen
  source_npi: 6,           # Red móvil
  dest_npi: 1              # ISDN para HLR de origen
}
```

Caso de Uso 4: Migración de Red

Problema: Migrar gradualmente de un antiguo esquema de direccionamiento a un nuevo esquema.

Solución: Enrutar según los indicadores de origen, transformar para el destino.

```
# Tráfico de formato antiguo: Transformar a nuevo formato
%{
  gt_prefix: "999",
  peer_id: 1,
  priority: 1,
  description: "Migración de formato legado a nuevo",
  source_tt: 0,            # Antiguo: Desconocido
  dest_tt: 2,              # Nuevo: Nacional
  source_npi: 0,           # Antiguo: Desconocido
  dest_npi: 1,             # Nuevo: ISDN
  source_nai: 0,           # Antiguo: Desconocido
  dest_nai: 3              # Nuevo: Nacional
},

# Tráfico de nuevo formato: Pasar sin cambios
%{
  gt_prefix: "999",
  peer_id: 1,
  priority: 2,
  description: "Pasar a través del nuevo formato",
  source_tt: 2,            # Ya en nuevo formato
  # Sin campos dest_ = preservar todos los valores
}
```

Pruebas de Sus Rutas

A través de la Interfaz Web

1. Navegue a **Enrutamiento → Rutas de Títulos Globales**
2. Haga clic en **"Mostrar Enrutamiento Avanzado (TT/NPI/NAI)"**
3. Complete sus indicadores de origen y destino
4. Haga clic en **Agregar Ruta**
5. Vaya a la página de **Prueba de Enrutamiento** para probar la ruta

A través de la API

El endpoint de ruta GT está montado en `/routing/gt-routes` en el listener de API (puerto **8445**, TLS habilitado). El cuerpo JSON utiliza el campo `prefix` (no `gt_prefix`), y `prefix` y `peer_id` son obligatorios. Los coincidencias y reescrituras de TT/NPI/NAI son opcionales.

```
curl -k -X POST https://localhost:8445/routing/gt-routes \
  -H "Content-Type: application/json" \
  -d '{
    "prefix": "44",
    "peer_id": 1,
    "priority": 1,
    "description": "Conversión TT del Reino Unido",
    "source_tt": 0,
    "dest_tt": 1
  }'
```

Campo	Tipo	Requerido	Descripción
`prefix`	String	Sí	Prefijo GT a coincidir
`peer_id`	Integer	Sí	ID del peer objetivo (`0` = DESCARTAR)
`priority`	Integer	No	Prioridad de ruta (por defecto `1`, menor gana)
`description`	String	No	Descripción legible por humanos
`source_ssn` / `dest_ssn`	Integer	No	Coincidir / reescribir SSN de la Parte Llamada
`source_tt` / `dest_tt`	Integer	No	Coincidir / reescribir Tipo de Traducción
`source_npi` / `dest_npi`	Integer	No	Coincidir / reescribir Indicador de Plan de Numeración
`source_nai` / `dest_nai`	Integer	No	Coincidir / reescribir Indicador de Naturaleza de la Dirección

Para eliminar una ruta, emita un `DELETE` a `/routing/gt-routes/{prefix}:{peer_id}` (por ejemplo `/routing/gt-routes/44:1`).

Prueba de Coincidencia de Rutas

Utilice la página de Prueba de Enrutamiento para simular tráfico entrante con indicadores específicos:

1. Ingrese el prefijo GT (por ejemplo, "441234567890")
2. Establezca los valores de TT, NPI, NAI
3. Haga clic en "Probar Ruta"
4. Vea qué ruta coincide y qué transformaciones se aplican

Solución de Problemas

Rutas No Coincidiendo

Verifique el orden de especificidad:

1. El prefijo GT más largo gana
2. Luego, SSN de origen específico sobre comodín
3. Luego, TT de origen específico sobre comodín
4. Luego, NPI de origen específico sobre comodín
5. Luego, NAI de origen específico sobre comodín
6. Finalmente, el número de prioridad más bajo gana

Transformaciones No Aplicadas

Verifique que los campos dest_ estén configurados:

- Vacío dest_tt/dest_npi/dest_nai significa "preservar el valor original"
- Debe establecer explícitamente dest_* para aplicar la transformación

Problemas de Rutas Duplicadas

La tabla :bag permite duplicados:

- La tabla de rutas GT es de tipo `:bag` (permite múltiples registros con la misma clave)
- La clave es `{gt_prefix, source_ssn}`
- Use TT/NPI/NAI para diferenciar rutas con el mismo prefijo+SSN
- Elimine rutas duplicadas no deseadas

Mejores Prácticas

1. **Documente sus conversiones:** Utilice descripciones de ruta descriptivas
2. **Pruebe a fondo:** Utilice la página de Prueba de Enrutamiento antes de la producción
3. **Monitoree los registros:** Verifique los registros INFO para transformaciones aplicadas
4. **Comience simple:** Agregue primero rutas de prefijo básicas, luego agregue reglas de indicadores
5. **Utilice comodines sabiamente:** Campos source_* vacíos coinciden con cualquier valor
6. **Preservar cuando sea posible:** Solo transforme lo que necesita cambiar
7. **Considere la especificidad:** Rutas más específicas anulan las menos específicas

---

Para más información, consulte:

- ← Volver a la Documentación Principal
- Guía STP - Documentación completa de enrutamiento
- Referencia de Configuración - Todas las opciones de configuración
- Guía de API - Documentación de la API REST

Enrutamiento de Títulos Globales M3UA STP

← Volver a la Documentación Principal

Descripción General

El M3UA STP admite el **enrutamiento de Títulos Globales (GT)** además del enrutamiento por Código de Punto (PC). El enrutamiento GT enruta mensajes SS7 basados en el Título Global del Parte Llamada en la capa SCCP, con un retroceso automático al enrutamiento por PC.

Cómo Funciona

Prioridad de Enrutamiento

1. **Enrutamiento GT (si está habilitado)**: Extraer GT Llamado de SCCP → Buscar en la tabla de enrutamiento GT
2. **Enrutamiento PC (retroceso)**: Extraer DPC de MTP3 → Buscar en la tabla de enrutamiento PC

Coincidencia de Prefijos GT

- Utiliza el algoritmo de **coincidencia de prefijo más largo**
- Ejemplo: GT "447123456789"
  - Coincide "447" (longitud 3) sobre "44" (longitud 2)
  - Enruta al par configurado para el prefijo "447"

Configuración

Habilitar Enrutamiento GT

En `config/runtime.exs`:

```
config :omniss7,
  enable_gt_routing: true,  # Establecer en false para usar solo
enrutamiento PC
```

Definir Rutas GT

```
config :omniss7,
  m3ua_gt_routes: [
    # Enrutar números de EE. UU. (que comienzan con 1234) al par 1
    %{gt_prefix: "1234", peer_id: 1, priority: 1, description:
"Números de EE. UU."},

    # Enrutar números del Reino Unido (que comienzan con 44) al
par 2
    %{gt_prefix: "44", peer_id: 2, priority: 1, description:
"Números del Reino Unido"},

    # Ruta más específica para móviles del Reino Unido (447xxx)
    %{gt_prefix: "447", peer_id: 2, priority: 1, description:
"Móviles del Reino Unido"},

    # Enrutar números australianos (que comienzan con 61) al par 3
    %{gt_prefix: "61", peer_id: 3, priority: 1, description:
"Números australianos"}
  ]
```

> Una configuración STP completa y lista para editar se envía en `config/stp_example_runtime.exs`. Consulte la Guía STP para saber cómo habilitarla.

Ejemplos de Uso

Ejemplo 1: Enrutamiento GT Básico

Mensaje: SCCP UDT con GT Llamado = "447700123456"

Decisión de Enrutamiento:

- GT extraído: "447700123456"
- Mejor coincidencia: "447" (móvil del Reino Unido) → par 2
- Mensaje enrutado al par 2

Ejemplo 2: GT No Encontrado, Retroceso a PC

Mensaje: SCCP UDT con GT Llamado = "331234567890", DPC = 100

Decisión de Enrutamiento:

- GT extraído: "331234567890"
- No se encontró ruta GT para "33"
- Retroceso al enrutamiento PC
- DPC 100 → par 1 (de la tabla de enrutamiento PC)
- Mensaje enrutado al par 1

Ejemplo 3: No Hay GT Presente

Mensaje: M3UA DATA sin SCCP, solo MTP3 con DPC = 200

Decisión de Enrutamiento:

- Sin capa SCCP, sin GT
- Usar solo enrutamiento PC
- DPC 200 → par 2
- Mensaje enrutado al par 2

Gestión en Tiempo de Ejecución

Estas funciones son expuestas por `M3UARouting` y pueden ser llamadas desde una consola `iex -S mix`.

Consultar Rutas GT

```
# Obtener todas las rutas GT
{:ok, routes} = M3UARouting.get_all_gt_routes()

# Cada registro es una tupla de 14 elementos. Los campos son:
# {:m3ua_gt_route, gt_prefix, source_ssn, peer_id, dest_ssn,
priority,
#  description, enabled, source_tt, dest_tt, source_npi, dest_npi,
#  source_nai, dest_nai}
#
# Ejemplo de salida (source_ssn y los campos de
reescritura/indicador se establecen en nil por defecto):
[
  {:m3ua_gt_route, "1234", nil, 1, nil, 1, "Números de EE. UU.",
true, nil, nil, nil, nil, nil, nil},
  {:m3ua_gt_route, "44", nil, 2, nil, 1, "Números del Reino
Unido", true, nil, nil, nil, nil, nil, nil},
  {:m3ua_gt_route, "447", nil, 2, nil, 1, "Móviles del Reino
Unido", true, nil, nil, nil, nil, nil, nil},
  {:m3ua_gt_route, "61", nil, 3, nil, 1, "Números australianos",
true, nil, nil, nil, nil, nil, nil}
]
```

Agregar Ruta GT Dinámicamente

`add_gt_route/3` toma el prefijo y el ID del par posicionalmente; todo lo demás (`priority`, `description`, `source_ssn`, `dest_ssn`, los emparejadores TT/NPI/NAI y reescrituras, y `enabled`) es una opción de palabra clave.

```
# Firma: add_gt_route(gt_prefix, peer_id, opts \ [])

# Agregar una ruta para números alemanes
M3UARouting.add_gt_route("49", 4, priority: 1, description:
"Números de Alemania")

# Agregar una ruta SMS que coincide con SSN 8 (SMSC) y reescribe
el SSN a 6 (HLR)
M3UARouting.add_gt_route("55", 4,
  source_ssn: 8,
  dest_ssn: 6,
  priority: 1,
  description: "Tráfico SMS a HLR"
)
```

Eliminar Ruta GT

```
# Eliminar ruta (gt_prefix, peer_id, source_ssn opcional)
M3UARouting.remove_gt_route("49", 4)
```

Probar Búsqueda GT

```
# Probar qué par manejaría este GT
# Firma: lookup_peer_by_gt(global_title, source_ssn \ nil, opts \
[])
{:ok, peer, rewrites} =
M3UARouting.lookup_peer_by_gt("447700123456")

# `peer` es el registro m3ua_peer de 13 elementos:
# {:m3ua_peer, peer_id, peer_name, protocol, role, local_ip,
local_port,
#  remote_ip, remote_port, routing_context, point_code,
network_indicator, status}
#
# `rewrites` es un MAP de las reescrituras de indicadores a
aplicar al reenviar.
# Los valores nil significan "mantener el original":
#   %{dest_ssn: nil, dest_tt: nil, dest_npi: nil, dest_nai: nil}
```

Cuando una ruta especifica reescrituras, aparecen en el mapa. Por ejemplo, una ruta `"55"` con `dest_ssn: 6`:

```
{:ok, peer, rewrites} =
M3UARouting.lookup_peer_by_gt("5512345678", 8)
# rewrites => %{dest_ssn: 6, dest_tt: nil, dest_npi: nil,
dest_nai: nil}
```

Una ruta con `peer_id: 0` (DROP) devuelve `{:error, :dropped}`; un GT no coincidente devuelve `{:error, :no_gt_route}`.

Registro

El STP registra decisiones de enrutamiento GT:

```
[info] Enrutando M3UA DATA: OPC=1000, DPC=200, GT
Llamado="447700123456" (desde el par 1)
[info] Enrutado por GT: 447700123456
[info] Mensaje enrutado del par 1 al par 2
```

Si falla el enrutamiento GT:

```
[info] Enrutando M3UA DATA: OPC=1000, DPC=200, GT
Llamado="331234567" (desde el par 1)
[info] No se encontró ruta GT para 331234567
[info] Mensaje enrutado del par 1 al par 2 (a través del
enrutamiento PC)
```

Soporte SCCP

Mensajes SCCP Soportados

Mensaje	Código	Soporte
UDT (Unitdata)	0x09	Completo: extracción de GT, reescritura de indicador, enrutamiento
UDTS (Unitdata Service)	0x0A	Completo: generado como un mensaje de retorno en caso de fallo de enrutamiento
XUDT (Extended Unitdata)	0x11	Completo: extracción de GT, reescritura de indicador, decremento de contador de saltos, mensaje de retorno
XUDTS (Extended Unitdata Service)	0x12	Completo: generado como un mensaje de retorno en caso de fallo de enrutamiento
LUDT (Long Unitdata)	0x13	Completo: extracción de GT, reescritura de indicador, decremento de contador de saltos, mensaje de retorno
LUDTS (Long Unitdata Service)	0x14	Completo: generado como un mensaje de retorno en caso de fallo de enrutamiento

XUDT y LUDT llevan un contador de saltos SCCP. El STP lo decrementa en cada salto de tránsito y, si llega a cero, descarta el mensaje y lo devuelve al originador (cuando se solicita retorno en caso de error). Consulte la Guía STP para más detalles.

Formatos GT Soportados

- **GTI 0-4**: Todos los tipos de indicadores GT
- **Codificación BCD**: Codificación estándar de dígitos de telefonía
- **Naturaleza de la Dirección**: Todos los tipos (Internacional, Nacional, etc.)

Rendimiento

- **Búsqueda GT**: O(n) donde n = número de rutas GT
- **Coincidencia de prefijos**: Optimizado con ordenamiento de coincidencia más larga primero
- **Recomendación**: Mantener el conteo de rutas GT < 1000 para un rendimiento óptimo

Solución de Problemas

GT No Está Siendo Extraído

Problema: Los registros muestran `Called GT=nil`

Soluciones:

1. Verifique que el tipo de mensaje SCCP sea UDT/XUDT/LUDT
2. Verifique que la Dirección del Parte Llamada lleve un Título Global
3. Habilite el registro de depuración para ver los detalles de análisis de SCCP

Par Incorrecto Seleccionado

Problema: Mensaje enrutado al par incorrecto

Soluciones:

1. Verifique la configuración del prefijo GT (la coincidencia más larga gana)
2. Verifique que `peer_id` coincida con los pares configurados
3. Verifique los valores de prioridad (menor = mayor prioridad)

Retroceso a PC No Funciona

Problema: Mensaje descartado cuando no se encuentra GT

Soluciones:

1. Verifique que `enable_gt_routing` sea `true`
2. Verifique que la tabla de enrutamiento PC tenga una ruta para el DPC
3. Asegúrese de que el par para el PC esté activo

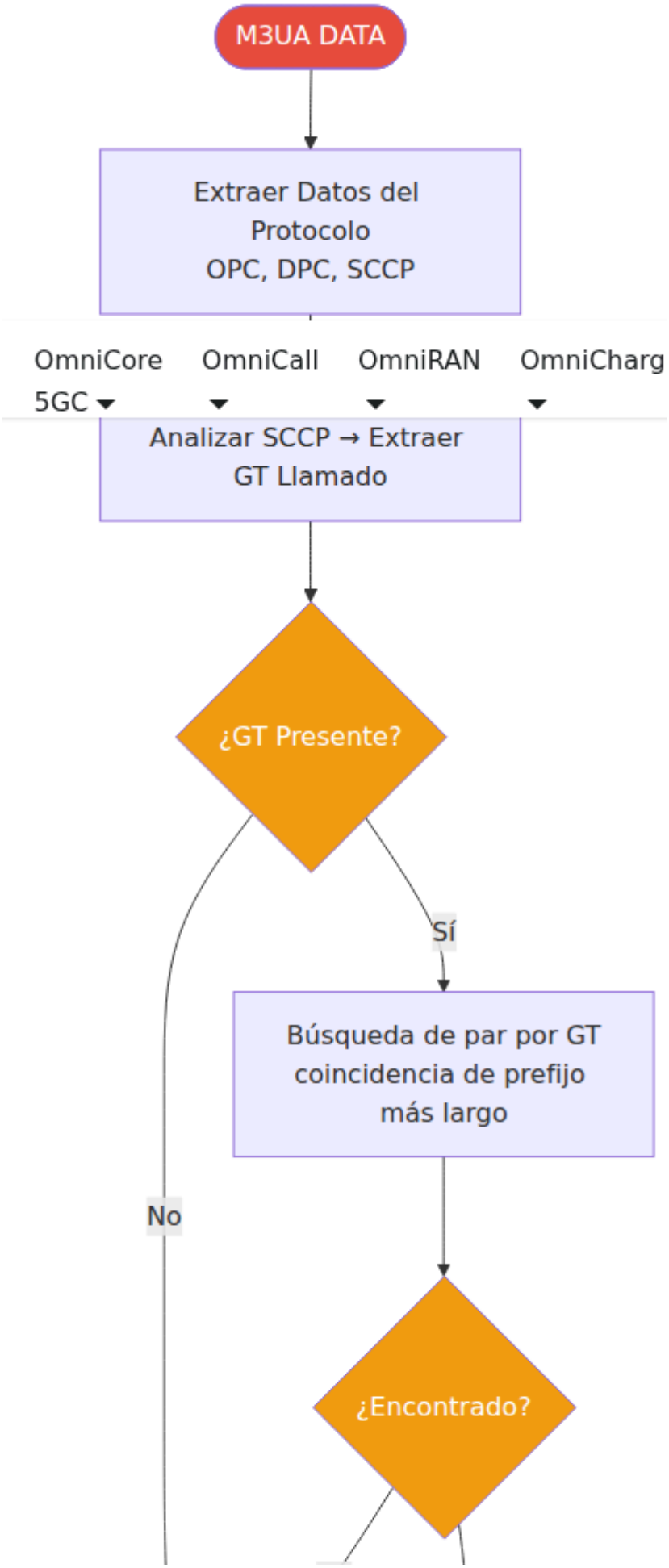
Mejores Prácticas

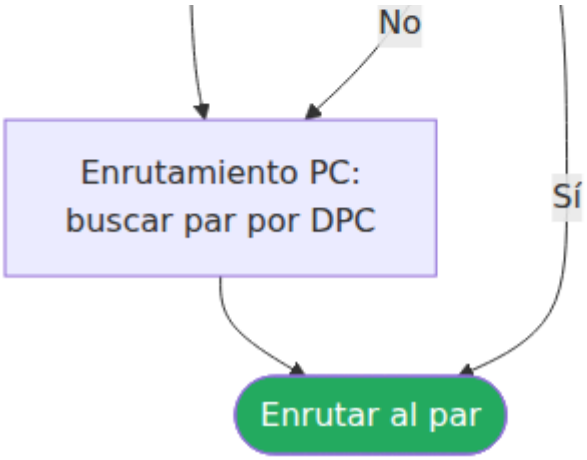
1. **Usar Prefijos Específicos**: Rutas más específicas anulan las menos específicas
   - Bueno: "447", "448", "449" para diferentes operadores del Reino Unido
   - Malo: Un solo "44" para todos los del Reino Unido
2. **Documentar Rutas**: Utilice el campo de descripción

   ```
   M3UARouting.add_gt_route("1234", 1, priority: 1,
     description: "ACME Corp - Costa Este de EE. UU.")
   ```

3. **Probar Rutas**: Utilice `lookup_peer_by_gt/1` para verificar
4. **Monitorear**: Verifique los registros para fallos de enrutamiento
5. **Retroceso**: Siempre configure rutas PC como respaldo

Arquitectura


M3UA DATA
Extraer Datos del Protocolo OPC, DPC, SCCP
OmniCore 5GC
OmniCall
OmniRAN
OmniCharg
Analizar SCCP → Extraer GT Llamado
¿GT Presente?
Sí
No
Búsqueda de par por GT coincidencia de prefijo más largo
¿Encontrado?



OmniSS7 por Omnitouch Network Services