Oracle® Communications Service Broker

Developer's Guide for GSM

Release 5.0

E20060-01

December 2010

ORACLE®

Oracle Communications Service Broker Developer's Guide for GSM, Release 5.0

E20060-01

Contents

Preface

This document provides detailed guidances for developing SIP applications and integrating these applications with the Oracle Communications Service Broker Next Generation - Intelligent Network (NG-IN) solution.

Audience

This document is intended for developers who want to develop SIP applications and integrate these applications with the Service Broker NG-IN solution.

This document assumes that the reader is already familiar with the following:

- Service Broker architecture and concepts
- Session Initiation Protocol (SIP) and the SIP-specific event notification
- CAP protocol specifications and procedures
- MAP protocol specifications and procedures

Conventions

The following convention is used to define nested structure of protocol messages:

<message>::<nested-parameter-1>:: ... <nested-parameter-n>

where:

- *<message>* is a message name
- *<nested-parameter-1>* is the first level nested parameter
- *<nested-parameter-n>* is the n-th level nested parameter

For example:

InitialDP:: CallingPartyNumber::NatureOfAddress describes the NatureOfAddress parameter of the CallingPartyNumber parameter of the InitialDP operation.

Reference Documents

The following documents provide additional information about Service Broker and relevant standards.

Oracle Communications Service Broker User Manuals

- *Oracle Communications Service Broker Concepts Guide*

- *Oracle Communications Service Broker System Administrator's Guide*

SIP Standards

- IETF RFC 3261, Session Initiation Protocol
- IETF RFC 3323, A Privacy Mechanism for the Session Initiation Protocol (SIP)
- IETF RFC 3325, Private Extensions to the Session Initiation Protocol
- IETF Internet Draft, Diversion Indication in SIP, August 25, 2004
- IETF Internet Draft, A Header to Deliver the Calling Party Category, October 21, 2005

Mobile Application Part (MAP) Standards

- ETSI TS 129 002 (3GPP TS 29.002) version 7.10.0, Mobile Application Part (MAP) specification
- ETSI TS 123 078 (3GPP TS 23.078) version 7.9.0, Customized Applications for Mobile network, Enhanced Logic (CAMEL) Phase X; Stage 2

Encoding Standards

- IETF RFC 3863, Presence Information Data Format (PIDF)
- ITU-T X.693, ASN.1 Encoding Rules, XML Encoding Rules (XER)

1

Principles of the Oracle Communications Service Broker NG-IN Solution

The following sections describe the principles of the Oracle Communications Service Broker NG-IN solution:

Introduction

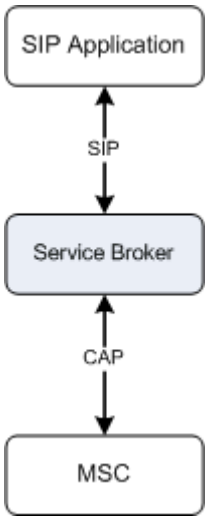
Service Broker NG-IN solution enables a SIP application to control an IN-enabled MSC or SSP in a legacy network.

With a Service Broker NG-IN solution, you can:

- Deploy new SIP applications and deliver them towards a legacy network
- Migrate legacy SCP-based applications to SIP-based applications and deliver them towards a legacy network.

Figure 1–1 shows a SIP application that controls an IN-enabled MSC in a legacy network.

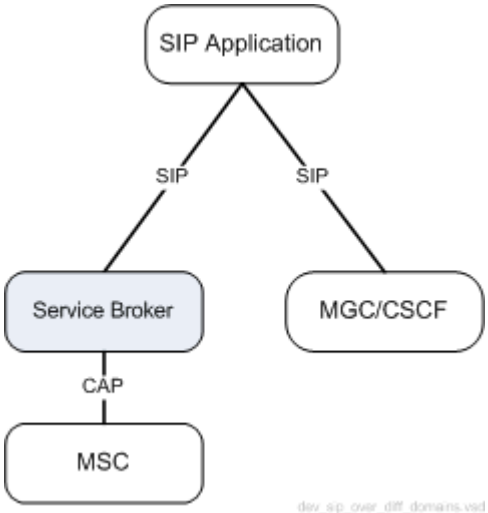
Figure 1–1 Architecture for Controlling a Legacy IN-Enabled MSC by a SIP Application



Service Broker provides SIP applications with a standard SIP interface to control IN-enabled MSCs. This enables a SIP application to control an MSC in a legacy network as it controls an MGC or CSCF in a SIP or IMS network. Furthermore, from the application developer's perspective, the application's control over an MSC does not require any network-specific customization.

Figure 1–2 shows a SIP application that provides call control to an MGC or CSCF in a SIP or IMS network, and to an IN-enabled MSC in a legacy network.

Figure 1–2 Architecture for controlling an MSC and MGC/CSCF by a SIP Application



Solution Architecture

The Service Broker NG-IN solution is composed of the following components:

- One or more SIP applications
- Service Broker

Figure 1–1 shows the Service Broker NG-IN solution architecture.

In an NG-IN solution, Service Broker has two external interfaces:

- Northbound SIP interface towards SIP applications
- Southbound IN interface towards the MSC

Session Control

Service Broker enables a SIP application to control an IN call through the SIP interface. An application may operate in a full call control mode or an initial call control mode acting as a SIP B2BUA or as a SIP Redirect Server accordingly.

Charging

Service Broker enables a SIP application to control an MSC for online and offline charging services. Charging operations are transferred from the application to Service Broker using SIP INFO messages. These messages carry an XML representation of the charging operation that needs to be performed.

For example, an application may send a SIP INFO message with a body that carries an XML representation of a CAP phase 4 FurnishChargingInformation operation. Upon receiving a SIP INFO, Service Broker sends a CAP FurnishCahrgingInformation towards the MSC.

User Interaction

Service Broker enables a SIP application to interact with a call party for providing service announcement to the call party with or without DTMF collection.

Based on application's instructions, Service Broker uses different media resources:

- MSC's internal resource
- External resource, that is gsmSRF
- MRF

When Service Broker uses internal or external resources, user interaction operations are transferred from the application to Service Broker using SIP INFO messages that carry an XML representation of the user interaction operation to be performed.

For example, an application may send a SIP INFO message with a body that carries the XML representation of CAP phase 4 PlayAnnouncement operation.

Multleg Control

Service Broker enables a SIP application to control individual parties in a call. For example, an application may create a new leg in an existing call or in a new call, connect two or more legs, split a leg out from the call, and more.

Multi-leg control is used by an application acting as a B2BUA to provide enhanced services, such as personalized ringback tone and click-to-dial.

Information Exchange through the SIP Interface

Service Broker exchanges information with the SIP application through the common SIP interface using two different mechanisms:

- Using SIP headers

- Using SIP message body

The following sections describe these two mechanisms.

Information Exchange using SIP Headers

To provide the application with the call related information received through the CAP interface, Service Broker uses the headers of the messages sent to the application.

For example, when Service Broker receives an InitialDP operation through the CAP interface, Service Broker sends a SIP INVITE message to the application and sets the Request-URI to the called party address as received in the CAP InitialDP operation. In the other direction, Service Broker uses the headers of the messages received from the application to construct CAP operation and send it towards the MSC.

In addition, to exchange information with a SIP application, Service Broker uses SIP tokens. For example Service Broker uses the **noa** token to exchange the nature of address information of various call parties with the SIP application.

Information Exchange using SIP Body

Service Broker uses the SIP message body to exchange two types of information:

- IN parameters, which are not naturally transferred using the SIP headers. For example, Service Broker may use the SIP INVITE message body to propagate the IN BearerCapability parameter towards the application.

  IN parameters are exchanged using the SIP body in both directions: from Service Broker to the application and from the application to Service Broker.

- SDP, which contains call leg information.

2

Developing a SIP Call Control Application

The following sections describe how to develop a SIP call control application:

> **Note:** The following sections contain tables that specify the IN protocols for which the functionality described in that section is relevant.

Controlling a Call

When Service Broker invokes a call control application, the application can perform one of the following actions:

- Rejecting the call
- Allowing the call to continue leaving the call information unmodified
- Allowing the call to continue with the call information modified. The application performs this action by providing Service Broker with the call routing information. When Service Broker receives the call routing information, Service Broker propagates this information towards the MSC through the IN interface.

If the application decides to allow the call to continue, the application can do this in one of the following forms:

- Routing the call while retaining call control for the entire call duration. An application that retains call control for the entire call duration is known as a full call control application.
- Routing the call without retaining call control for the entire call duration. An application that routes the call to destination without retaining call control for the entire call duration is known as initial call control application.

Invoking a SIP Application

Table 2–1 Invoking a SIP Application: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

Acting as a standard SIP entity, Service Broker invokes a SIP application by sending a SIP INVITE message. The Service Broker sets the SIP INVITE content based on the information received in the CAP InitialDP operation.

Table 2–2 shows the content of the SIP INVITE message as set by Service Broker. The SIP Header column describes the SIP INVITE headers. The Source column includes the source of the information used by Service Broker to set the SIP INVITE message.

Table 2–2 Service Broker Mapping of CAP 4 InitialDP Operation to SIP INVITE Message

SIP Header	Source
Request-URI :: User part (see the note for Request-URI below the table)	InitialDP :: CalledPartyNumber OR InitialDP :: CalledPartyBCDNumber CalledPartyBCDNumber is used only when CalledPartyNumber is not included in InitialDP.
Request-URI :: Domain part	Service Broker configuration
To :: User part	Equals to the value set in RequestURI :: User part
To :: Domain part	Service Broker configuration
From :: User part	InitialDP :: AdditionalCallingPartyNumber If AdditionalCallingPartyNumber is not included in InitialDP, the From header is set from InitialDP :: CallingPartyNumber. If InitialDP :: CallingPartyNumber :: Address presentation restricted indicator is set to "presentation restricted": The From header is set as follows: From: "Anonymous" sip:anonymous@anonymous.invalid The Privacy header is set as follows: Privacy: id
From :: Domain part	Service Broker configuration
P-Asserted-Identity :: User part (see the note for P-Asserted-Identity below the table)	InitialDP :: CallingPartyNumber
P-Asserted-Identity :: Domain part	Service Broker configuration

Notes to Table 2-2:

- Request-URI

Service Broker sets the Request-URI with a **noa** token. For more information on the noa token, see "Exposing Nature of Address".

- P-Asserted-Identity

The Service Broker sets the P-Asserted-Identity with a noa token. For more information on the noa token, see "Exposing Nature of Address".

Exposing Nature of Address

Table 2–3 Exposing Nature of Address: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

Service Broker exposes CAP nature of address information towards the SIP application using a vendor specific token, named noa. The noa token carries the nature of address of various call parties as follows:

- Nature of address of the called party number provided by noa token in the Request-URI header of the SIP INVITE, which is sent to the application
- Nature of address of the calling party number provided by noa token in the P-Asserted-Identity header of the SIP INVITE, which is sent to the application

Table 2–4 describes how Service Broker sets the noa token for various nature of address values.

Table 2–4 NOA Token

Call party nature of address	NOA token content
subscriber number (national use)	subscriber
unknown (national use)	unknown
national (significant) number (national use)	national
International	noa token is not used

As shown in Table 2–4, Service Broker does not set the noa token for nature of address of type “International”. When the nature of address of one of the call parties is set to “International”, the user part in the corresponding SIP header is prefixed with “+”. For example:

- For the calling party nature of address of type “International”, Service Broker sets the P-Asserted-Identity in the SIP INVITE, which is sent to the application, as follows:

```
P-Asserted-Identity:
<sip:+972978880l9@domain:5060>
```

- For the calling party nature of address of type “national (significant) number”, Service Broker sets the P-Asserted-Identity in the SIP INVITE, which is sent to the application, as follows:

```
P-Asserted-Identity:
<sip:97888019@domain:5060;noa=national>
```

Developing an Initial Call Control Application

Table 2–5 Developing an Initial Call Control Application: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

To provide an initial call control, the application responds to the SIP INVITE message with a SIP 302 Moved Temporarily message.

An initial call control application can perform one of the following actions:

- Updating the called party number (that is to replace the number dialed by the calling party with a new number)
- Leaving the called party number unmodified

The following sections describe how to implement these two options.

Note: An initial call control application may also reject a call. This functionality is described in "Rejecting a Call".

Updating the Called Party Number

Table 2–6 Updating the Called Party Number: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

To update the called party number (that is to replace the number dialled by the calling party with a new number), the application sets the SIP 302 Moved Temporarily to the new destination address.

The new destination address is set in the user part of the Contact header. This makes Service Broker to respond to the CAP InitialDP with a CAP Connect operation.

Note: The application should set the user part with digits only.

Figure 2–1 shows the high level architecture for an initial call control application that updates the called party number.

Figure 2–1 Architecture for Updating the Called Party Number by an Initial Call Control Application over a SIP Network

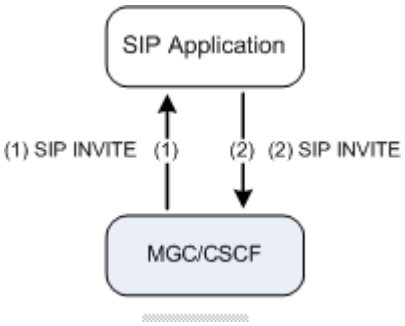


Figure 2–2 shows the same application as shown on Figure 2–1. However, on Figure 2–2, the application provides the same functionality over a CAP network using Service Broker.

Figure 2–2 Architecture for Updating the Called Party Number by an Initial Call Control Application over a CAP Network Using Service Broker

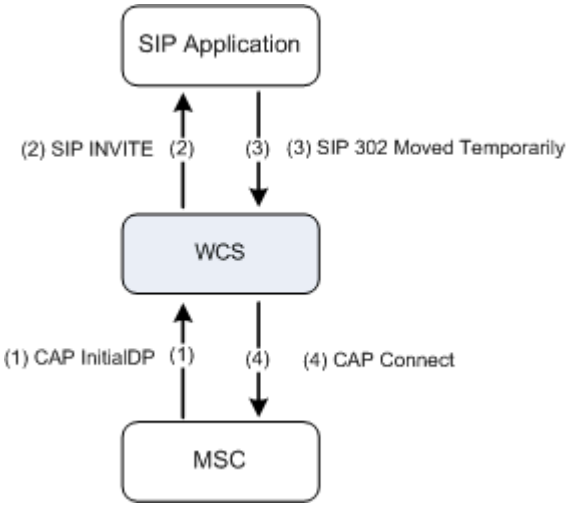
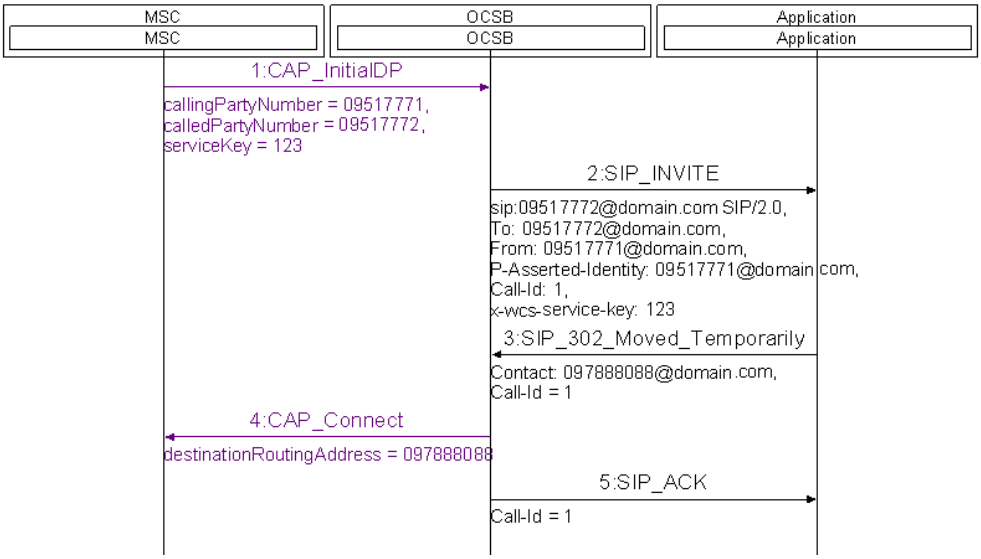


Figure 2–3 shows the detailed sequence diagram for an initial call control application that updates the called party number.

Figure 2–3 Initial Call Control Application Updates the Called Party Number



Creating CAP Connect

Service Broker creates the CAP Connect operation based on the information received in the SIP 302 Moved Temporarily response. Table 2–7 shows the content of the CAP Connect operation as set by Service Broker.

Table 2–7 Service Broker Maps SIP 302 Moved Temporarily to CAP Connect Operation

CAP Connect	Source
DestinationRoutingAddress :: AddressSignal	302 Moved Temporarily :: Contact :: user part
DestinationRoutingAddress :: NatureOfAddress	302 Moved Temporarily :: Contact :: noa Service Broker sets the NatureOfAddress in the Connect operation based on the noa token, as set by the application in the Contact header of SIP 302 Moved Temporarily. For more information on noa values, see Table 2–4.
DestinationRoutingAddress :: InternalNetworkNumberIndicator	Service Broker configuration
DestinationRoutingAddress :: NumberingPlanIndicator	Service Broker configuration

Leaving the Called Party Number Unmodified

Table 2–8 Leaving the Called Party Number Unmodified: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

To leave the called party number unmodified, the application sets the SIP 302 Moved Temporarily response to the address provided in the user part of the Request-URI

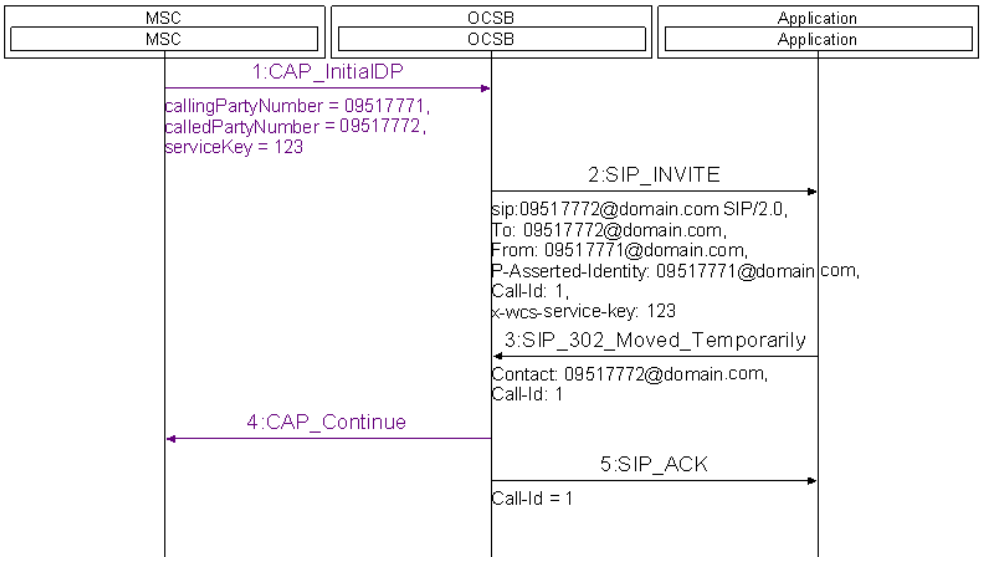
header of the SIP INVITE, which is sent by Service Broker. This address is set in the user part of the Contact header.

This action makes Service Broker to respond to a CAP InitialDP with a CAP Continue operation.

The Continue operation has no parameters.

Figure 2–4 shows the detailed sequence diagram for an initial call control application that leaves the called party number unmodified.

Figure 2–4 Initial Call Control Application Leaves the Called Party Number Unmodified



Developing a Full Call Control Application

Table 2–9 Developing a Full Call Control Application: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

To provide a full call control, the application implements a SIP B2BUA. The application receives the SIP INVITE message sent by Service Broker and creates a new SIP dialog by sending a new SIP INVITE towards Service Broker.

Service Broker receives the SIP INVITE and sends a CAP Continue or a CAP Connect operation accompanied by a CAP RequestReportBCSEvent operation.

The CAP RequestReportBCSEvent operation instructs the MSC to monitor the call for call related events (for example O_Busy or O_No_Answer) and send notifications to Service Broker when an event is detected.

Service Broker sets the specific events to be monitored in the CAP RequestReportBCSEvent operation as defined in the Service Broker configuration.

Note: Service Broker enables the application to specify events to be monitored, and by doing this, to overwrite the Service Broker configuration. For more information, see "Controlling the EDPs Arming".

The following sections describe various call control capabilities. To provide each of the call control capabilities defined below, the application must follow relevant instructions described in the following sections.

Handling the SDP

Table 2–10 Handling the SDP: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

A full control application needs to propagate the SDP which is provided by Service Broker, back-to-back. Figure 2–5 shows how the SDP is handled during the call initiation phase.

Figure 2–5 Architecture for Handling an SDP (Call Initiation Phase)

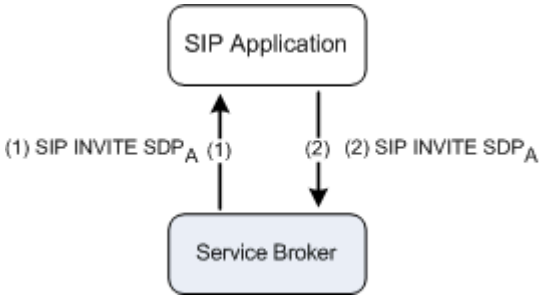
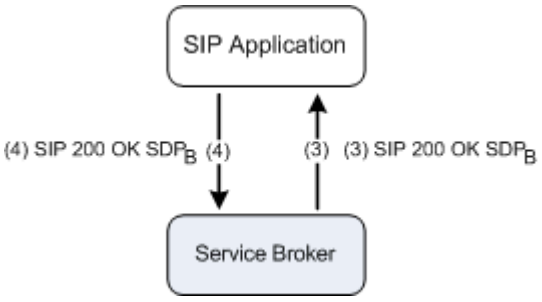


Figure 2–6 shows how the SDP is handled during the call answering phase.

Figure 2–6 Architecture for Handling an SDP (Call Answering Phase)



Handling the SIP Route Header

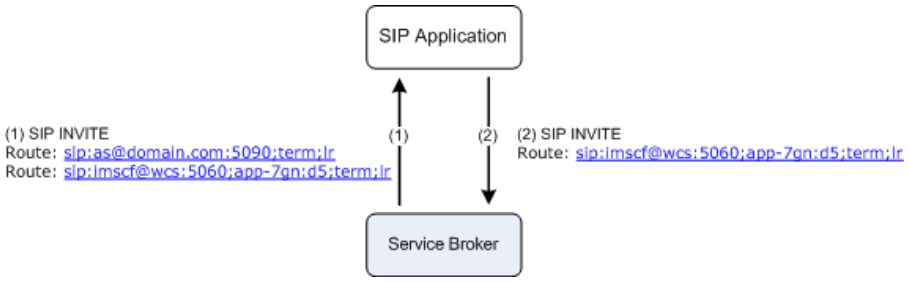
Table 2–11 Handling the SIP Route Header: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

The SIP Route header is defined in RFC 3323. A SIP full call control application implemented over the Service Broker has to follow the loose-routing mechanism defined in RFC 3261.

Figure 2–7 demonstrates the loose-routing mechanism.

Figure 2–7 Architecture for the SIP Loose Routing Mechanism



Updating the Called Party Number

Table 2–12 Updating the Called Party Number: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

To update the called party number, application sets the SIP INVITE which is sent to Service Broker, to a new destination address. The new destination address is set in the user part of the RequestURI header. This makes Service Broker to respond to the CAP InitialDP with a CAP Connect operation.

Figure 2–8 shows the high level architecture for a full control application that updates the called party number.

Figure 2–8 Architecture for Updating the Called Party Number by a Full Call Control Application over a SIP Network

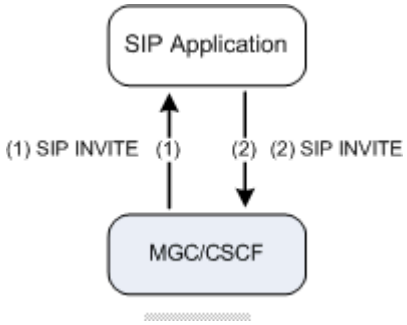


Figure 2–9 shows the same application as shown on Figure 2–8. However, on Figure 2–9, the application provides the same functionality over a CAP network using Service Broker.

Figure 2–9 Architecture for Updating the Called Party Number by a Full Call Control Application over a CAP Network Using Service Broker

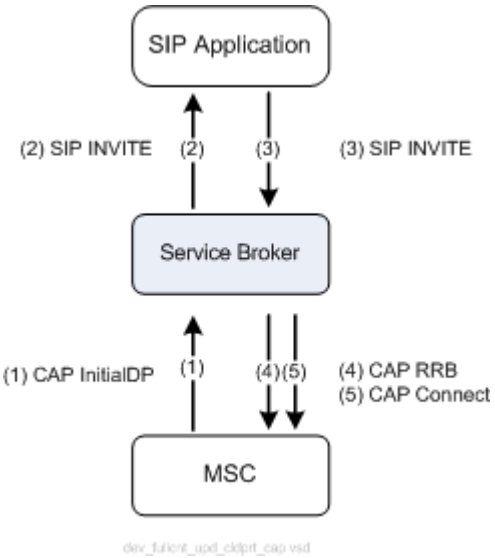


Figure 2–10 and Figure 2–11 show the detailed sequence diagram for a full control application that updates the called party number.

Figure 2–10 Full Call Control Application Updates the Called Party Number

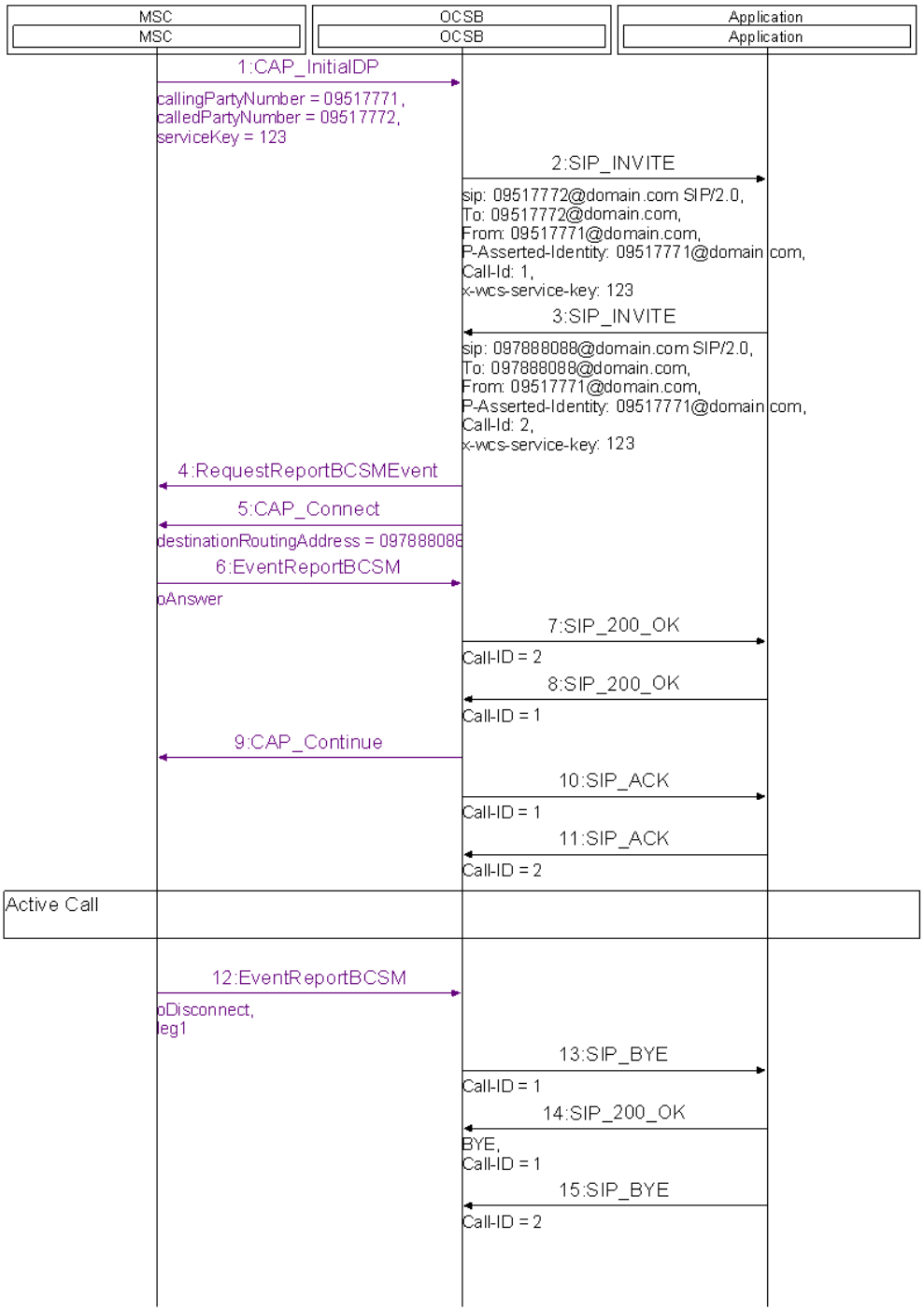
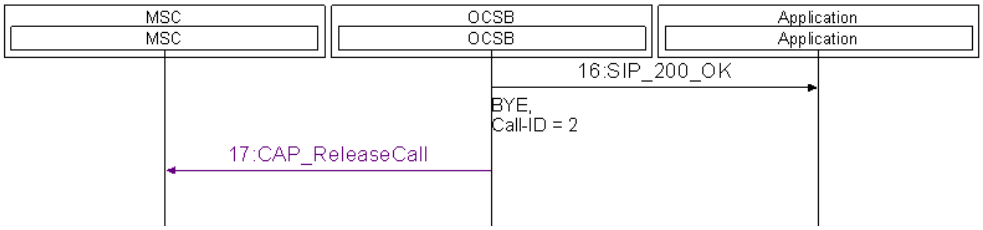


Figure 2–11 Full Call Control Application Updates the Called Party Number (cont'd)



Service Broker creates the CAP Connect operation based on the information received in the SIP INVITE, which is sent by the application. Table 2–13 shows the content of the CAP Connect operation as set by Service Broker.

Table 2–13 Service Broker Maps SIP INVITE to CAP Connect Operation

CAP Connect	Source
DestinationRoutingAddress :: AddressSignal	INVITE :: Request-URI :: user part
DestinationRoutingAddress :: NatureOfAddress indicator	INVITE :: Request-URI :: noa token For more information on the noa token, see "Exposing Nature of Address" and "Updating the Nature of Address".
DestinationRoutingAddress :: InternalNetwork NumberIndicator	Service Broker configuration
DestinationRoutingAddress :: NumberingPlan Indicator	Service Broker configuration

Leaving the Called Party Number Unmodified

Table 2–14 Leaving the Called Party Number Unmodified: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

To leave the called party number unmodified, the application sets the SIP INVITE, which is sent to Service Broker, to the address provided in the SIP INVITE message received from Service Broker.

This procedure is done by copying the user part of the Request-URI of the received SIP INVITE and pasting it into the SIP INVITE sent to Service Broker. This makes Service Broker to respond to InitialDP with a CAP Continue operation.

Updating the Nature of Address

Table 2–15 Updating the Nature of Address: Applicable CAP Phases

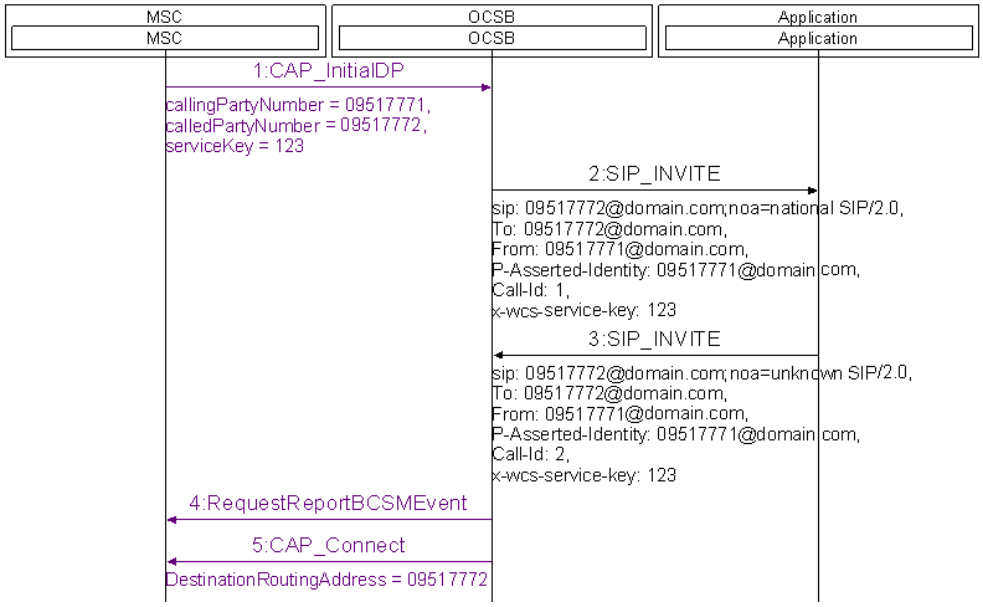
CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

To update the nature of address of a call party, the application sets the noa token in the SIP INVITE message, which is sent to Service Broker, to the required value as follows:

- To update the called party nature of address, the application sets the noa token in the Request-URI header to the required value.

Figure 2–12 shows an example in which the application updates the called party nature of address. This example assumes that the CalledPartyNumber in CAP InitialDP is set with NatureOfAddress of type “national”. The application updates the called party nature of address and sets it to "unknown". This causes Service Broker to set the DestinationRoutingAddress in the CAP Connect operation to NatureOfAddress of type “unknown”.

Figure 2–12 Application Updates the Called Party Nature of Address



Note: In the example shown on Figure 2–12, although the application does not update the called party number, Service Broker uses CAP Connect rather than CAP Continue. This is done because the Connect operation enables Service Broker to update the called party nature of address towards the MSC. CAP Connect is set to the called party number as received from the CAP InitialDP operation.

For more information on **tNOA** token, see "Invoking a SIP Application".

Controlling the EDPs Arming

Table 2–16 Controlling the EDPs Arming: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

As described in "Developing a Full Call Control Application", when Service Broker receives a SIP INVITE message, which is sent by a full call control application, Service Broker sends a CAP Continue or a CAP Connect operation accompanied by a CAP RequestReportBCSEvent operation.

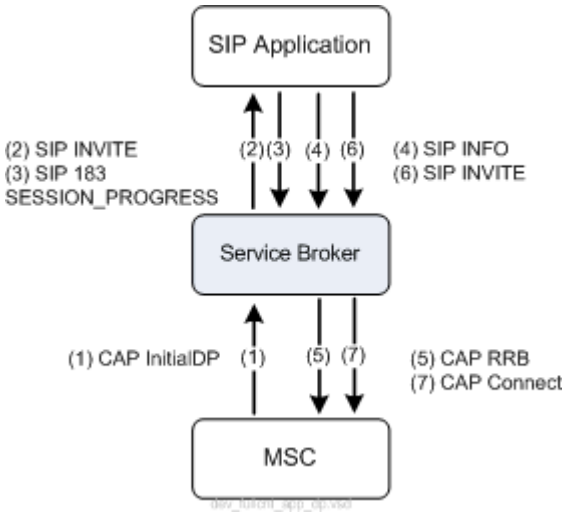
The specific events to be monitored are set in the CAP RequestReportBCSEvent operation as defined in the Service Broker configuration.

In some cases, it is required that an application dynamically controls the events that Service Broker arms for a given call, that is to define the CAP EDPs set by Service Broker in the CAP RRBCSM operation.

To control the events that Service Broker arms in the RequestReportBCSEvent operation, the application sends a SIP INFO message prior to the SIP INVITE. The SIP INFO is sent through the SIP dialog created by Service Broker and contains a XER representation of the CAP RequestReportBCSEvent operation.

Figure 2–13 shows the high level architecture for a full control application that controls the DPs armed by Service Broker.

Figure 2–13 Architecture for Controlling DPs by a Full Call Control Application



Receiving Call Events Notifications

Table 2–17 Receiving Call Events Notifications: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

Service Broker notifies a full call control application about encountered call-related events. For each call event encountered at the MSC and reported to Service Broker, Service Broker notifies the application using a corresponding SIP message as described in Table 2–18.

Notes to Table 2-18:

- RouteSelectFailure events are applicable for originating calls only.
- The table is applicable for both originating and terminating BCSM. For example, Service Broker uses SIP 486 Busy Here to notify the application about oCalledPartyBusy in an originating call and for tBusy in a terminating call.
- If a Disconnect event is reported by the calling party, Service Broker sends a SIP BYE message through the dialog created by Service Broker. If a Disconnect event is reported by the calling party, Service Broker sends a SIP BYE message through the dialog created by the application.
- For the DPs reported using SIP INFO, Service Broker sets the SIP INFO with a XER representation of the corresponding CAP EventReportBCSM operation.
- Table 2–18 provides the full EDP list supported in CAP 4. Earlier CAP phases support only part of the EDPs listed in the table.

Table 2–18 Event Notifications

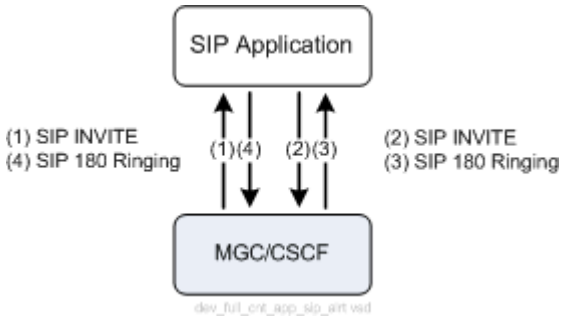
CAP event	SIP message
Route Select Failure	410 Gone
Busy	486 Busy Here
No Answer	480 Temporary Unavailable
Term Seized / Call Accepted	180 Ringing
Answer	200 OK
Disconnect	BYE
Abandon	CANCEL

To confirm notification and enable Service Broker to instruct the MSC to continue call processing at event notification, the application propagates the received SIP message back-to-back.

Note:: Call processing is suspended by the MSC when an event armed as EDP-R is encountered. When EDP-R is reported to Service Broker, MSC requests Service Broker instructions for call processing.

Figure 2–14 shows a full control application in the call initiation process. When the called party is alerted, the application receives a SIP 180 Ringing message and propagate it back-to-back.

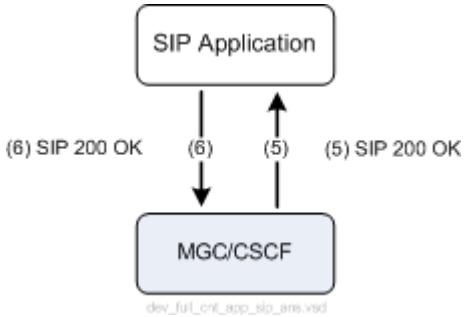
Figure 2–14 Architecture for Initiating a Call over a SIP Network (Alerting Phase)



When the called party answers the call, the application receives a SIP 200 OK and again propagates this message back-to-back towards the initiating side.

Figure 2–15 shows a full control application in the call answering phase.

Figure 2–15 Architecture for Initiating a Call over a SIP Network (Answering Phase)



Finally, when the called party (or in another scenario, the calling party) disconnects the call, the application receives a SIP BYE and propagates it towards the initiating side as shown on Figure 2–16.

Figure 2–16 Architecture for Initiating a Call over a SIP Network (Disconnecting Phase)

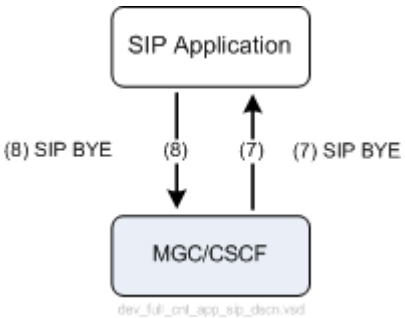


Figure 2–17, Figure 2–18, and Figure 2–19 show the same application as shown on Figure 2–14 and Figure 2–16. However, the application below provides the same functionality over a CAP network using Service Broker.

Figure 2–17 Architecture for Initiating a Call by a Full Control Application over a CAP Network Using Service Broker (Alerting Phase)

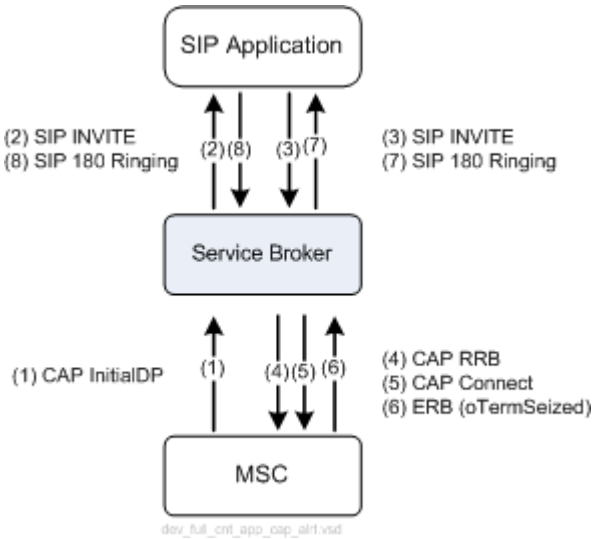


Figure 2–18 Architecture for Initiating a Call by a Full Control Application over a CAP Network Using Service Broker (Answering Phase)

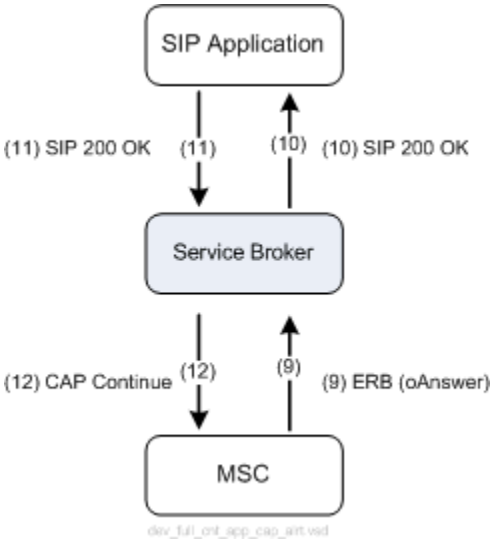
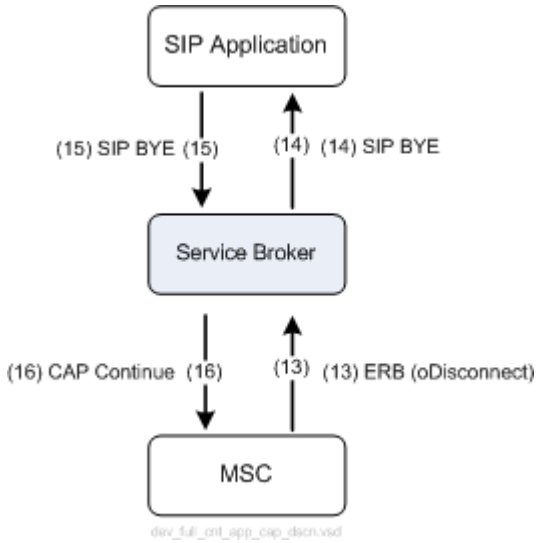


Figure 2–19 Architecture for Initiating a Call by a Full Control Application over a CAP Network Using Service Broker (Disconnecting Phase)



Terminating a Call

Table 2–19 Terminating a Call: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

To terminate a call, the application sends a SIP BYE request towards Service Broker. The BYE request is sent on both active dialogs, that is the dialog created by Service Broker and the dialog created by the application.

Service Broker uses the BYE request to terminate the CAP dialog towards MSC using a CAP ReleaseCall operation.

Figure 2–20 shows architecture for a full control application terminating a call.

Figure 2–20 Architecture for Terminating a Call over a SIP Network

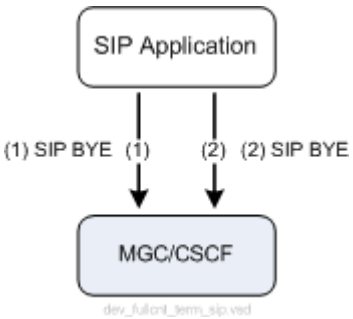
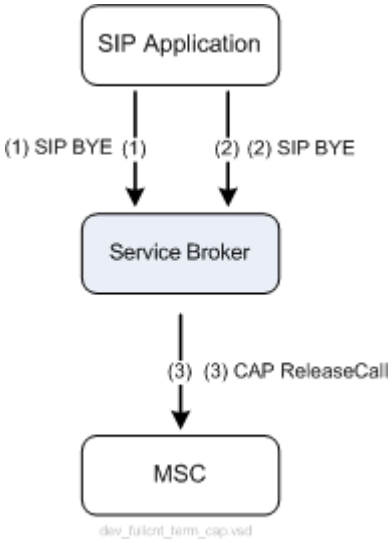


Figure 2–21 shows the same application as shown on Figure 2–20. However, on Figure 2–21, the application provides the same functionality over a CAP network using Service Broker.

Figure 2–21 Architecture for Terminating a Call over a CAP Network Using Service Broker



Service Broker Error Responses

Service Broker may respond towards the application with a SIP error in case an application request cannot be fulfilled or in other error cases as defined in Table 2–20.

Table 2–20 SIP Errors

SIP Error	Description
405 Method Not Allowed	Sent by Service Broker in case the application requests a call control operation which is not legal in the current moment on the CAP interface
403 Forbidden	Sent by Service Broker in case the application requests a call control operation which is not supported by the specific CAP interface
415 Unsupported Media Type	Sent by Service Broker in case the application provides a non-supported SDP

Rejecting a Call

Table 2–21 Rejecting a Call: Applicable CAP Phases

CAP 1	CAP 2	CAP 3	CAP 4
YES	YES	YES	YES

To reject a call, the application responds to the SIP INVITE with a SIP error response (for example, SIP 404 Not Found). When Service Broker receives the SIP error response, Service Broker performs one of the following actions:

- Instructs the MSC to terminate the call by sending a CAP ReleaseCall operation
- Instructs the MSC to allow the call to continue by sending a CAP Continue operation

Service Broker determines the action to be performed based on its configuration.

Figure 2–22 shows the high level architecture for a full control application that terminates a call.

> **Note:** The figures below shows an example in which the application uses SIP 404 Not Found to reject the call. In practice, applications are not limited to a specific SIP error response.

Figure 2–22 Architecture for Rejecting a Call over a SIP Network

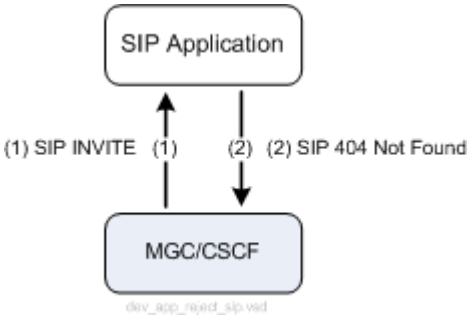
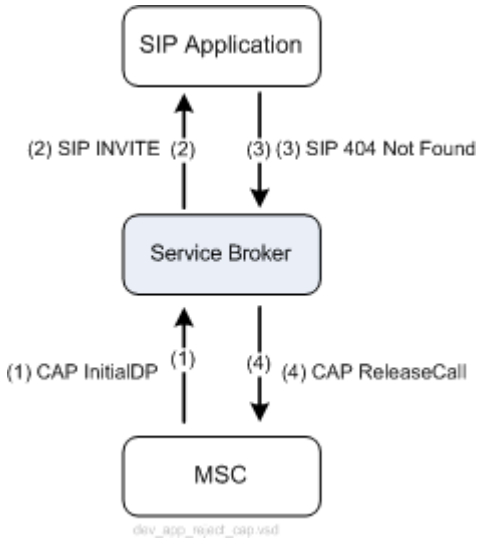


Figure 2–23 shows the same application as shown on Figure 2–22. However, on Figure 2–23, the application provides the same functionality over a CAP network using Service Broker.

Figure 2–23 Architecture for Rejecting a Call over a CAP Network Using Service Broker



Controlling the CAP Release Cause

Service Broker uses the SIP error response sent by the application to set the cause parameter in the CAP ReleaseCall operation.

To instruct the Service Broker to set the ReleaseCall operation to a specific cause value, the application uses the corresponding SIP error response as defined in Table 2–22.

Table 2–22 Release Cause

CAP Cause	SIP Response
31, normal unspecified	400 to 479
19, no answer from user	480
31, normal unspecified	481 to 485
17, user busy	486
31, normal unspecified	487 to 699
31, normal unspecified	Any other SIP error response

3

Understanding the Service Broker NG-IN Solution for Presence and Subscriber Status Applications

The following sections describe the principles of the Oracle Communications Service Broker NG-IN solution for Presence and Subscriber Status eXtensions applications:

Introduction

The Service Broker NG-IN solution enables a SIP application to access SS7 network entities that communicate using the MAP protocol. The solution currently supports interaction with HLRs and VLRs.

With the Service Broker NG-IN solution for PSX applications, your SIP application can access a mobile network to perform the following actions:

- Obtain a mobile subscriber's state, location, and service subscription information from an HLR
- Modify mobile subscriber's subscription information in an HLR and VLR

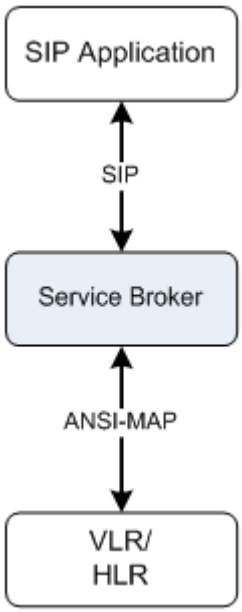
Solution Architecture

The Service Broker NG-IN solution for PSX applications consists of the following components:

- One or more SIP applications
- Service Broker
- MAP network entity

Figure 3–1 shows a SIP application that interacts with an HLR or VLR in a legacy mobile network.

Figure 3–1 Architecture for Interacting with MAP Network Entities



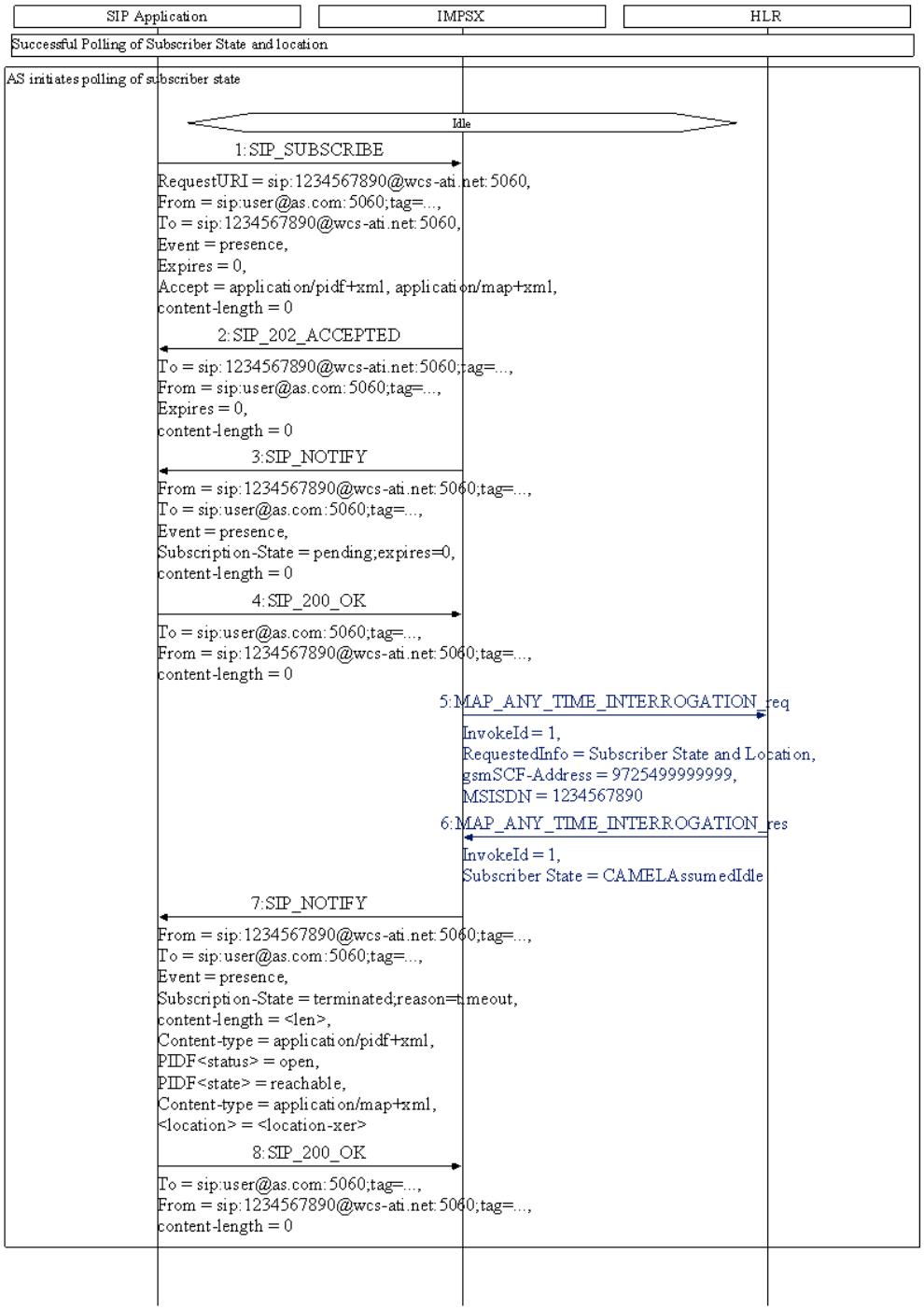
The application-facing side of Service Broker provides a SIP application with a standard SIP interface. The interface is based on the SIP SUBSCRIBE and SIP NOTIFY messages. Implementing the interaction between a SIP application and a network entity does not require any network-specific customization.

SIP SUBSCRIBE and SIP NOTIFY Interface

A SIP application interacts with Service Broker through a standard SIP interface using the subscribe and notify mechanism. Figure 3–2 shows a typical call flow in the solution:

1. The SIP application subscribes to Service Broker for a specific type of operation, such as obtaining the subscriber's status. The application performs the subscription by sending a SIP SUBSCRIBE message to Service Broker.
2. The SIP SUBSCRIBE message triggers Service Broker to perform an appropriate operation on the SS7 network entity.
3. After Service Broker received a response from the entity, Service Broker sends the application either a SIP NOTIFY message or a failure response. In both cases, Service Broker terminates the subscription by
   - Setting the Subscription-State header of the SIP NOTIFY message to "terminated"
   - Setting the "reason" token of the Subscription-State header of the SIP NOTIFY message to "timeout"

Figure 3–2 Basic Interrogation Call Flow



Exchanging Information Through the SIP Interface

Service Broker exchanges information with the SIP application through the common SIP interface using two different mechanisms:

- SIP headers

  To provide Service Broker with information required to trigger specific MAP operations, the SIP application uses SIP headers.

Service Broker also supports several SIP header tokens. For example, Service Broker supports the **`requested-info`** token to allow the SIP application to specify which information it wishes to obtain.

- SIP message body

  Service Broker uses the SIP message body to exchange the following two types of information:

  - Accept XER or BER encoded MAP operation arguments that need to be sent towards SS7 entities. For example, Service Broker uses the SIP SUBSCRIBE message body to accept MAP ANY-TIME-SUBSCRIPTION-INTERROGATION argument, and further construct the argument before sending it towards the SS7 entity.
  - Pass XER or BER encoded MAP operation results to the SIP application.

4

Developing a Presence and Subscriber Status SIP Application

The following sections describe how to develop a Presence and Subscriber Status eXtensions SIP application in the Oracle Communications Service Broker NG-IN solution:

- Understanding Common SIP Interface Concepts
- Obtaining Subscriber's State and Location
- Obtaining Mobile Subscriber's Subscription Information
- Modifying Mobile Subscriber's Information

Understanding Common SIP Interface Concepts

Using the NG-IN solution, SIP applications can obtain mobile subscriber's information that is stored in SS7 network entities such as HLR. To obtain subscriber's information, a SIP application has trigger a SIP SUBSCRIBE message to Service Broker and specify inside the information that it needs. Service Broker returns the subscriber's information in a SIP NOTIFY message.

The following sections describe common parts of the SIP SUBSCRIBE and NOTIFY interface, such as headers and SIP errors, that the SIP application generates every time it communicates with Service Broker, regardless of the type of operation the SIP application requests to perform.

Specifying the Address of an SS7 Entity

SIP applications can instruct Service Broker which SS7 entity to connect using the domain part of the To header. An application can set the domain part using one of the following methods:

- Setting an alias

  When a SIP application needs to communicate with an SS7 entity whose SCCP address is configured in the SS7 SSU, the application sets an alias that refers to this address.

  Service Broker uses this alias to resolve the preconfigured SCCP address (that is point code or GT address). For example, if you set the To header to **sip:1234567890@hlr01**, then Service Broker will resolve hlr01 to a real SCCP address, based on the SCCP addresses configured in the SS7 SSU.

Note that if you specify an alias that resolves to a dynamic GT address, then you must also specify the GT digits, using the 'gtaddr' token. For example, **sip:1234567890@hlr01;gtaddr=972543349098**

- Using an IM-PSX SIP domain

  When a SIP application needs to communicate with an SS7 entity whose alias is preconfigured in IM-PSX, in the **PsxSipDomain** parameter, the application sets the IM-PSX SIP domain.

  When you set the IM-PSX SIP domain in the domain part of the To header, for example, **sip:1234567890@ocsb-psx.net**, Service Broker uses the alias configured in the **DefaultSs7EntityAlias** parameter to resolve an SCCP address that is already configured in the SS7 SSU.

- Using the Anonymous string

  If a SIP application needs to communicate with an entity whose SS7 address is not configured in Service Broker, the application constructs the domain part of the To header using the "Anonymous" string.

  Setting the "Anonymous" string means the application uses additional tokens in the To header to specify an SCCP address.

  The "Anonymous" string requires an application to set the tokens described in Table 4–1.

Table 4–1 Tokens Required to Specify an SCCP Address

Token	Type	Description
gtaddr	STRING	Stands for Global Title Address. The parameter contains digits.
nai	STRING	Stands for Nature of Address Indicator. Possible values: ■ unknown ■ subscriberNumber ■ nationalReserved ■ nationalSignificant ■ international
np	STRING	Stands for Numbering Plan. Possible values: ■ unknown ■ isdn ■ generic ■ data ■ telex ■ maritimeMobile ■ landMobile ■ isdnMobile
netind	STRING	Stands for Network Indicator. Possible values: ■ national ■ international

Table 4–1 (Cont.) Tokens Required to Specify an SCCP Address

Token	Type	Description
tt	BYTE	Stands for Translation Type.
spc	Up to 24 bit decimal	Stands for Signaling Point Code.
ssn	BYTE	Stands for Subsystem Number.

An application must set the tokens described in Table 4–1 as follows:

- The **netind** token is always required.
- One of the following tokens is required:
  - **gtaddr**. If an application uses the **gtaddr** token, to allow Service Broker to build the global title indicator, the application must set at least one of the following tokens: **tt** or **nai**.
  - **spc** accompanied by the **ssn** token

For example, the following combinations of tokens are considered valid:

- **netind**, **gtaddr**, **nai**, **np**, **tt**
- **netind**, **gtaddr**, **tt**
- **netind**, **gtaddr**, **tt**, **np**
- **netind**, **gtaddr**, **nai**
- **netind**, **spc**, **ssn**
- **netind**, **gtaddr**, **nai**, **np**, **tt**, **spc**, **ssn**

Specifying the Identity of a Mobile Subscriber

SIP applications specify the mobile subscriber whose information is required by using the domain part of the RequestURI.

SIP applications should set the RequestURI as follows:

- Set the mobile subscriber's MSISDN in the user part
- Set the IM-PSX address, as configured in **PsxSipDomain**, in the domain part.

A SIP application can use the **noa** token, to specify the MSISDN nature of address that is later used on the MAP interface. The possible **noa** token values are:

- subscriber
- unknown
- national

To specify that the MSISDN nature of address is 'international', the SIP application must set the MSISDN in the RequestURI user part in a global format, with a leading '+' sign.

For example, a SIP application can set the RequestURI to:

- sip:1234567890@ocsb-psx.net:5060;noa=national
- sip:+972540987610@ocsb-psx.net:5060;

SIP NOTIFY Message Body Formats

Service Broker uses the SIP NOTIFY message body to pass a MAP result to the SIP application. Service Broker passes the MAP result in one of the following formats:

- "application/map-phase3+xml"

  The SIP NOTIFY message body contains a full MAP operation result, encoded in XER.

- "application/map-phase3+ber"

  The SIP NOTIFY message body contains a full MAP operation result, encoded in BER.

- "application/pidf+xml"

  The SIP NOTIFY message body contains the mobile subscriber's state and location in PIDF format.

Specifying Supported SIP NOTIFY Message Body Formats

SIP applications use the Accept header to specify the SIP NOTIFY message body formats that the application supports. The value of the Accept header must be one that is also supported by the Service Broker, that is values configured under the IM-PSX AcceptHeadersMBean.

For example, "application/map-phase3+xml" and "application/pidf+xml". See "SIP NOTIFY Message Body Formats" for more information.

The Accept header is optional. If not specified, Service Broker assumes that the SIP application supports the body formats specified in the IM-PSX AcceptHeadersMBean.

Setting the Expires Header

SIP applications must always set the Expires header to zero. If an application does not set the Expires header, Service Broker assumes this header is set to zero.

Handling SIP Errors

Service Broker can return all standard SIP errors. Table 4–2 provides additional interpretation of some standard SIP errors specifically for Presence and Subscriber eXtensions applications.

Table 4–2 SIP Errors

Error	Description
400 Bad Request	The application sets the value of the Expires header to a value other than zero.
403 Forbidden	The application sets the value of the domain part in the requestURI which is different from the PsxSipDomain.
415 Unsupported Media Type	The application sets the value of the Accept header to a format which Service Broker does not support.
489 Bad Event	This error may indicate one of the following problems: ■ The application sets the Event header with an unsupported value ■ The application does not set the Event header at all ■ The application sets the **requested-info** token with an unsupported value (see "Event Header")

Table 4–2 (Cont.) SIP Errors

Error	Description
500 Server Internal Error	Unexpected internal error has occurred in Service Broker.
603 Declined	There's a mismatch between the Accept header value and the **requested-info** token value. For example, if the application requests mobile subscriber's location (sets the **requested-info** token to 'Mobile-location'), but does not support XER format (does not set "application/map-phase3+xml" in the Accept header) that is required to receive the mobile subscriber's location.

Obtaining Subscriber's State and Location

Using the NG-IN solution, SIP applications can obtain a mobile subscriber's state and location that is stored in a network's HLR. Such information includes:

- Subscriber's state. For example, reachable or busy.
- Subscriber's location. For example, geographical location and VLR number.
- Other subscribe's information. For example, extensions information.

The ability to obtain a mobile subscriber's state and location information is based on and the MAP ANY-TIME-INTERROGATION operation. SIP applications can request Service Broker to obtain a mobile subscriber's state and location information by using the SIP SUBSCRIBE message. Service Broker returns the state and location information inside the SIP NOTIFY message body.

The following sections describe how a SIP application needs to generate a SIP SUBSCRIBE message to Service Broker in order to obtain a subscriber's state and location, and then process the SIP NOTIFY message which Service Broker sends back to the SIP application.

Generating a SIP SUBSCRIBE Message

The following SIP headers must be set in the SIP SUBSCRIBE message:

Common SIP Headers

- RequestURI

  For more information, see "Specifying the Identity of a Mobile Subscriber"
- To

  For more information, see "Specifying the Address of an SS7 Entity"
- Accept

  For more information, see "SIP NOTIFY Message Body Formats"
- Expires

  For more information, see "Setting the Expires Header"

Event Header

The SIP application must set the Event header to 'Presence'. In addition, the SIP application may define the **requested-info** token to specify the information that Service Broker needs to request from an HLR. For example, the application may request information only about the subscriber state without information about its location.

The SIP application can set **requested-info** token to one of the following values:

- Mobile-state
- Mobile-location

If the SIP application does not specify the **requested-info** token, Service Broker returns both the mobile subscriber's state and location.

Processing a SIP NOTIFY Request

Service Broker returns a mobile subscriber's state and location information inside a SIP NOTIFY message body. Depending on the information that the SIP application has requested from Service Broker, the SIP NOTIFY message body may contain either one of the following or both:

- Subscriber's state - provided in PIDF format
- Subscriber's location and any other information - provided in the XER or BER format

This section describes important SIP headers and the SIP message body that a SIP application receives from Service Broker.

Subscription-State Header

The Subscription-State header value is always 'terminated'.

To explain the reason why a subscriber's state and location cannot be returned, the SIP application uses the **reason** token. Table 4–3 lists the possible values of the **reason** token:

Table 4–3 Possible Values of the Reason Token

Token Value	Description
timeout	Interrogation terminated successfully. The result is available in the SIP NOTIFY message body.
map_unknownsubscriber	HLR does not recognize the subscriber whose subscription information has been requested.
map_datamissing	HLR stated that the some data is missing in the MAP operation request.
map_unexpecteddatavalue	HLR found unexpected data in the MAP operation request.
map_systemfailure	There is a problem to connect the HLR.
unknown	Unexpected internal Service Broker error occurred.
addressresolutionfailure	Service Broker failed to resolve the SCCP address alias, that is the SS7 entity address.
map_atinotallowed	HLR does not permit interrogation using the MAP-ANY-TIME-INTERROGATION operation.
map_timeout	HLR does not respond.

Content-Type Header

The Content-Type header specifies the format of the SIP NOTIFY message body, as follows:

- "application/map-phase3+xml" - the body contains the full MAP-ANY-TIME-INTERROGATION operation result structure encoded in the XER format.
- "application/map-phase3+ber" - the body contains the full MAP-ANY-TIME-INTERROGATION operation result encoded in the BER format.
- "application/pidf+xml" - the body contains subscriber's state encoded in PIDF format.
- "multipart/mixed; boundary='frontier'" - the body contains multiple formats, both the subscriber's state encoded in PIDF and the full MAP-ANY-TIME-INTERROGATION operation result structure encoded in XER.

SIP Message Body

The SIP NOTIFY message body contains information about subscriber's state and location as they were requested by the SIP application in the Event header and **requested-info** token of the SIP SUBSCRIBE message (for more information, see "Event Header").

The information requested by the SIP application is delivered in the message body as follows:

- Subscriber's state is provided in an XML according to the PIDF schema. For more information, see "Subscriber's State".
- Subscriber's location and additional subscriber information is provided in the XER or BER format. For more information, see "Other Subscriber Information".

Subscriber's State

Subscriber's state is provided in an XML, according to the PIDF schema.

The following example shows how information about the subscriber's state is encoded:

```
<?xml version="1.0" encoding="UTF-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf "
xmlns:ts="urn:ietf:params:xml:ns:pidf:terminal-status"
entity="pres:123456789@psx-ocsb.net">
   <tuple id="sg89ae">
      <status>
         <basic>open</basic>
            <ts:state>reachable</ts:state>
      </status>
      <timestamp>2008-04-01T18:08:20Z</timestamp>
   </tuple>
</presence>
```

Table 4–4 explains the PIDF elements and attributes.

Table 4–4 PIDF Elements

Element	Description
<presence>	The root element. The element includes the <entity> attribute and 0 or more <tuple> elements. The value of the entity attribute contains the value of the RequestURI received on the SIP SUBSCRIBE message with 'pres' uri-scheme.

Table 4–4 (Cont.) PIDF Elements

Element	Description
<tuple>	Contains the mobile subscriber's state information that consists of a mandatory <status> element accompanied by the optional <timestamp> element.
<status>	Contains one optional <basic> element accompanied by the <state> element.
<basic>	Specifies a subscriber's availability for communications. Possible values: ■ Open ■ Close
<ts:state>	Extension element that specifies the subscriber's state. Possible values: ■ Reachable ■ Unreachable ■ Busy ■ Unknown The value of <ts:state> correlates with the value of <basic> as follows: ■ When <basic> contains "Open", <ts:state> contains "Reachable". ■ When <basic> contains "Close", <ts:state> may contain "Unreachable", "Busy", or "Unknown".
<timestamp>	Specifies the date and time when the presence information was created.

Other Subscriber Information

Subscriber's location and other information is provided in the XER or BERformat. In this case, the SIP NOTIFY message body contains the full MAP operation result structure.

Obtaining Mobile Subscriber's Subscription Information

Using the NG-IN solution, SIP applications can obtain mobile subscriber's subscription information that is stored in an HLR. Such information includes:

- Subscriber's basic information, for example, IMSI
- Subscriber's service information, for example, indication on services that are invoked for incoming and outgoing calls, mobility changes, incoming and outgoing SMS.

The ability to modify this information is based on the following MAP operations:

- MAP-ANY-TIME-SUBSCRIPTION-INTERROGATION
- MAP-SEND-IMSI

The ability to obtain subscription information is based on MAP operations. To obtain subscription information, a SIP application has to construct a XER or BER representation of the MAP operation request, and pass it to Service Broker inside the SIP SUBSCRIBE message body. Service Broker returns the subscription information in the SIP NOTIFY message body.

The following sections specify SIP interface requirements for subscription information interrogation, in addition to the common requirements specified at "Understanding Common SIP Interface Concepts".

Generating a SIP SUBSCRIBE Message

The following SIP headers must be set in the SIP SUBSCRIBE message:

Common SIP Headers

- RequestURI

  For more information, see "Specifying the Identity of a Mobile Subscriber"

- To

  For more information, see "Specifying the Address of an SS7 Entity"

- Accept

  For more information, see "SIP NOTIFY Message Body Formats"

- Expires

  For more information, see "Setting the Expires Header"

Event Header

SIP applications must set the Event header to 'SubQuery'.

Content-Type Header

SIP applications use the Content-Type header to specify the SIP SUBSCRIBE message body format, that is the MAP operation encoding format. Possible values are:

- "application/map-phase3+xml" - when the MAP operation request inside the SIP SUBSCRIBE message body is encoded in XER
- "application/map-phase3+ber" - when the MAP operation request inside the SIP SUBSCRIBE message body is encoded in BER

The SIP application should use two additional tokens to provide an indication of the MAP operation encoded inside the SIP message body:

- **op**, which defines the MAP operation code. You can set this token to one of the following values depending on the operation you want to trigger:
  - 58, when you want to trigger MAP-SEND-IMSI
  - 62, when you want to trigger MAP-ANY-TIME-SUBSCRIPTION-INTERROGATION
- **dir**, which defines the MAP operation direction. Set this token to "invoke".

For example:

Content-Type: "application/map-phase3+xml; op=62; dir=invoke"

Processing the SIP NOTIFY Message

Service Broker returns a mobile subscriber's subscription information inside a SIP NOTIFY message body. Service Broker passes the MAP operation result, as was received from the HLR, encoded in XER or BER. The SIP application can request a preferable format, using the Accept header. See "Specifying Supported SIP NOTIFY Message Body Formats" for more information.

The following SIP headers can provide additional information to the MAP operation result in the SIP message body.

Subscription-State Header

The Subscription-State header value is always 'terminated'.

In case of a problem, the SIP application can use the **reason** token to identify the failure reason. Table 4–5 lists the possible **reason** token values:

Table 4–5 Possible Values of the Reason Token

Token Value	Description
timeout	Interrogation terminated successfully. The result is available in the SIP NOTIFY message body.
map_unknownsubscriber	HLR does not recognize the subscriber whose subscription information has been requested.
map_datamissing	HLR stated that the some data is missing in the MAP operation request.
map_unexpecteddatavalue	HLR found unexpected data in the MAP operation request.
map_systemfailure	There is a problem to connect the HLR.
unknown	Unexpected internal Service Broker error occurred. An application may also receive this error if the XER or BER provided in the SIP SUBSCRIBE message body cannot be decoded to a MAP message.
addressresolutionfailure	Service Broker failed to resolve the SCCP address alias, that is the SS7 entity address.
map_atsinotallowed	HLR does not permit interrogation using the MAP-ANY-TIME-SUBSCRIPTION-INTERROGATION operation.
map_bearerservicenotprovisioned	The bearer service for which information is requested was not provisioned in the HLR.
map_teleservicenotprovisioned	The teleservice for which information is requested, was not provisioned.
map_illegalssoperation	Illegal supplementary service operation.
map_ssnotavailable	Supplementary service is not available.
map_informationnotavailable	The requested information is not available.

Modifying Mobile Subscriber's Information

Using the NG-IN solution, SIP applications can modify subscriber's data in an HLR or VLR. The ability to modify this information is based on the following MAP operations:

- MAP-ANY-TIME-MODIFICATION
- MAP-INSERT-SUBSCRIBER-DATA

SIP applications can request Service Broker to modify subscription information using the SIP SUBSCRIBE message. Service Broker returns the result of the modification operation inside the SIP NOTIFY message body.

The following sections describe how a SIP application needs to generate a SIP SUBSCRIBE message to Service Broker in order to modify subscriber's data, and then

process the SIP NOTIFY message which Service Broker sends back to the SIP application.

Generating a SIP Subscribe Message

The following sections describe the SIP headers that must be set in the SIP SUBSCRIBE message.

Common SIP Headers

- RequestURI

  For more information, see "Specifying the Identity of a Mobile Subscriber".

- To

  For more information, see "Specifying the Address of an SS7 Entity".

- Accept

  For more information, see "SIP NOTIFY Message Body Formats".

- Expires

  For more information, see "Setting the Expires Header".

Event Header

The SIP application must set the Event header to 'SubUpdate'.

Content-Type Header

SIP applications use the Content-Type header to specify the SIP SUBSCRIBE message body format, that is the MAP operation encoding format. The header can contain one of the following values:

- "application/map-phase3+xml", when the MAP operation request inside the SIP SUBSCRIBE message body is encoded in XER
- "application/map-phase3+ber", when the MAP operation request inside the SIP SUBSCRIBE message body is encoded in BER

The SIP application can use two additional tokens to provide an indication of the MAP operation encoded inside the SIP message body:

- **op**, which defines the MAP operation code. You can set this token to one of the following values depending on the operation you want to trigger:
  - 7, when you want to trigger MAP-INSERT-SUBSCRIBER-DATA
  - 65, when you want to trigger MAP-ANY-TIME-MODIFICATION
- **dir**, which defines the MAP operation direction. Set this token to "invoke".

For example:

Content-Type: "application/map-phase3+xml; op=65; dir=invoke"

If the body is empty, Service Broker returns the SIP error 400 'Bad request'.

Processing a SIP Notify Message

Service Broker returns a result of a modification operation inside a SIP NOTIFY message body. This section describes the SIP headers and the SIP message body that a SIP application receives from Service Broker.

Subscription-State Header

The Subscription-State header value is always 'terminated'. If a problem occurs, the SIP application can set the **reason** token to the values described in Table 4–6.

Table 4–6 Possible Values of the Reason Token

Token Value	Description
addressresolutionfailure	Service Broker failed to resolve the SCCP address alias, that is the SS7 entity address.
map_bearerservicenotprovisioned	The bearer service for which information is requested was not provisioned in the HLR.
map_callbarred	Operator determined barring or supplementary service barring management is active
map_datamissing	HLR stated that the some data is missing in the MAP operation request.
map_illegalssoperation	Illegal supplementary service operation.
map_informationnotavailable	The requested information is not available.
map_sserrorstatus	Supplementary service error status
map_ssincompatibility	Supplementary service incompatibility
map_sssubscriptionviloation	Supplementary service subscription violation
map_teleservicenotprovisioned	The teleservice for which information is requested, was not provisioned.
map_atmnotallowed	MAP ANY-TIME-MODIFICATION operation is not allowed by the HLR
map_unexpecteddatavalue	HLR found unexpected data in the MAP operation request.
map_unknownsubscriber	HLR does not recognize the subscriber whose subscription information was requested.
timeout	Interrogation terminated successfully. The result is available in the SIP NOTIFY message body.
unknown	Unexpected internal Service Broker error occurred. An application may also receive this error if the XER or BER provided in the SIP SUBSCRIBE message body cannot be decoded to a MAP message.

Content-Type Header

The Content-Type header specifies the format of the SIP NOTIFY message body, as follows:

- "application/map-phase3+xml", when the body contains the full modification operation result structure encoded in the XER format
- "application/map-phase3+ber", when the body contains the full modification operation result structure encoded in the BER format