Oracle® Communications Subscriber Data Management

Subscriber Provisioning User Guide

Release 9.3

910-6869-001 Revision B

October 2014

Table of Contents

Chapter 1: Introduction.....8

About this document.....9

Scope and audience.....9

Document organization.....9

Documentation Admonishments.....10

Related publications.....11

My Oracle Support (MOS).....11

Emergency Response.....12

Locate Product Documentation on the Oracle Technology Network Site.....12

Chapter 2: Getting Started.....13

Safety Warnings and Cautions.....14

Electrostatic Discharge (ESD).....14

Accessing the System.....14

- Establish serial connection.....14
- Establish Secure Shell (SSh) connection.....15
- System login.....16

Chapter 3: User Interfaces.....20

Subscriber Provisioning Using XML Templates.....21

- Using TCP socket for provisioning in direct mode.....21
- Using SOAP to provision in direct mode.....23
- Bulk and Template Provisioning.....23

Creating and Managing Users for the User Interfaces.....31

- User management using CLI.....32
- User management using WebCI.....40

Creating and Managing Users for Notifications.....51

- Notification Management Using CLI.....51
- Notification Management Using WebCI.....56

Chapter 4: Provisioning subscribers for Mobile Number Portability (WebCI).....62

Provisioning MNP DB Mismatch Error Handling feature....63
Add MNP subscriber....63
Display MNP MSISDN profile....64

Chapter 5: Examples of XML Templates for Subscriber Provisioning....66

Introduction....67
Subscriber provisioning sequence....67
Subscription provisioning template....68
SIM card provisioning template....69
HLR subscriber profile provisioning template....70
SIP Subscriber Provisioning....71
HSS subscriber provisioning file....72
SLF subscriber provisioning file....80
DNS Enum subscriber provisioning file....80
AAA subscriber provisioning file....81
LTE-HSS subscriber provisioning templates....83
Update request template....85
Delete XML request template....86
Select XML request file....86
Request-Reply XML file....87

Chapter 6: Converting an original XML request file to an XML template file....88

About XML request files and XML template files....89
About XML long and short formats....89
Comparison of XML request files and XML template files....89
Create XML template request....91
Create command template....94
Create invoking file....96

Chapter 7: Subscriber Profile Repository (SPR)....99

Subscription Profile Repository (SPR)....100
SPR subscriber provisioning (XML)....100
SPR subscriber provisioning (XML-REST)....103
SPR Subscriber Quota provisioning (XML)....105
SPR Subscriber Quota Provisioning (XML-REST)....110
SPR subscriber DynamicQuota provisioning (XML)....113

SPR subscriber DynamicQuota provisioning (XML-REST)....115
SPR subscriber State provisioning (XML)....116
SPR subscriber State provisioning (XML-REST)....118
SPR Pool provisioning (XML)....118
SPR Pool provisioning (XML-REST)....122

Glossary....125

List of Figures

Figure 1: Configure SSh client (PuTTY)........16

Figure 2: User Manager Window........41

Figure 3: User Provisioning Window to Modify a User........43

Figure 4: Group Provisioning Window to Modify a Group........45

Figure 5: UserAccessPrivileges provisioning window........47

Figure 6: Service Security Provisioning Window........50

Figure 7: ApplicationIdentity Provisioning Window to Create Applications........57

List of Tables

Table 1: Admonishments....11

Table 2: Pre-defined Users 17

Table 3: TCP connection commands....21

Table 4: Access Privileges....31

Table 5: User table operations per user interface....33

Table 6: Access privileges operations per user interface....37

Table 7: User table operations per user interface....41

Table 8: Access privileges operations per user interface....46

Table 9: UserApplicationMap table....54

Table 10: UserApplicationMap table....59

Table 11: Conversion summary....90

Chapter 1

Introduction

Topics:

- *About this document.....9*
- *Scope and audience.....9*
- *Document organization.....9*
- *Documentation Admonishments.....10*
- *Related publications.....11*
- *My Oracle Support (MOS).....11*
- *Emergency Response.....12*
- *Locate Product Documentation on the Oracle Technology Network Site.....12*

This chapter provides general information about manual organization, the scope of this manual, its targeted audience, how to get technical assistance, and how to locate customer documentation on the Customer Support site.

About this document

This document describes how to provision subscribers using the XML interface. Procedures demonstrate how to use the SOAP protocol, the Command File Loader, Command Template Loader, and Command Template Viewer tools. It also provides the order in which subscribers must be provisioned and gives examples of how XML files with and without templates should be written for subscriber provisioning of each application.

Scope and audience

This manual contains the standard user interface descriptions but focuses on using how to use the Command File Loader, Command Template Loader, and Command Viewer to perform bulk provisioning with XML templates. This document also explains how to convert a standard XML request file of type Insert, Update, or Delete to a template Request file and the related Command and Invoking files.

This document is intended for operators that are responsible for and qualified to provision the Subscriber Data Management system.

Document organization

This document is organized into the following chapters:

- *Introduction* contains general information about this document, how to contact the Tekelec Customer Care Center, and how to locate the customer documentation on the Customer Support site.
- *Getting Started* introduces the safety notes such as Warnings and Cautions; warns about potential electrostatic discharge (ESD) when handling hardware, and explains how to connect to the system and log in for the first time.
- *User Interfaces* explains how to use the Command File Loader, Command Template Loader, and Command Template Viewer to provision the Subscriber Data Management system with XML templates. This chapter also describes how to create and manage users for the CLI and WebCI interfaces and for notifications.
- *Provisioning subscribers for Mobile Number Portability (WebCI)* contains procedures for provisioning subscribers for Mobile Number Portability using the WebCI.
- *Examples of XML Templates for Subscriber Provisioning* provides examples of XML templates for the major SDM applications.
- *Converting an original XML request file to an XML template file* explains in detail how to use a standard XML request and convert it to a Request template with the Invoking files, which creates the subscription ID.

About links and references

Information within the same document is linked and can be reached by clicking the hyperlink.

To follow references pointing outside of the document, use these guidelines:

General:

- Locate the referenced section in the Table of Content of the referenced document.
- Locate the same section name in the referenced document.
- Place the PDF files in one folder or on a disc and use the powerful Adobe PDF search functions to locate related information in one or more documents simultaneously.

Alarms

- *SDM Alarms Dictionary*

Product, features, concepts

- *SDM Product Description*

Monitoring, maintenance, or troubleshooting:

- Procedures: *Monitoring, Maintenance, Troubleshooting User Guide*
- Entities: *Monitoring, Maintenance, Troubleshooting Reference Manual*

Subscriber provisioning:

- Procedures: *Subscriber Provisioning User Guide*
- Entities: *Subscriber Provisioning Reference Manual*

System configuration:

- Procedures: *System Configuration User Guide*
- Entities: *System Configuration Reference Manual*

User Interfaces:

- *User guides*
  - How to use the user interface
  - How to set up users (permissions, groups, services)
- *Reference manuals*
  - About user interfaces
  - Entities for setting up users

To determine the components of the complete documentation set delivered with the software, refer to the *SDM Documentation Roadmap* delivered with each documentation set.

Documentation Admonishments

Admonishments are icons and text throughout this manual that alert the reader to assure personal safety, to minimize possible service interruptions, and to warn of the potential for equipment damage.

Table 1: Admonishments

Icon	**Description**
DANGER	**Danger**: (This icon and text indicate the possibility of *personal injury*.)
WARNING	**Warning**: (This icon and text indicate the possibility of *equipment damage*.)
CAUTION	**Caution**: (This icon and text indicate the possibility of *service interruption*.)
TOPPLE	**Topple**: (This icon and text indicate the possibility of *personal injury* and *equipment damage*.)

Related publications

For a detailed description of the available SDM documentation, refer to the *SDM Documentation Roadmap* included with your SDM documentation set.

My Oracle Support (MOS)

MOS (*https://support.oracle.com*) is your initial point of contact for all product support and training needs. A representative at Customer Access Support (CAS) can assist you with MOS registration.

Call the CAS main number at **1-800-223-1711** (toll-free in the US), or call the Oracle Support hotline for your local country from the list at *http://www.oracle.com/us/support/contact/index.html*. When calling, make the selections in the sequence shown below on the Support telephone menu:

1. Select **2** for New Service Request
2. Select **3** for Hardware, Networking and Solaris Operating System Support
3. Select **2** for Non-technical issue

You will be connected to a live agent who can assist you with MOS registration and provide Support Identifiers. Simply mention you are a Tekelec Customer new to MOS.

MOS is available 24 hours a day, 7 days a week, 365 days a year.

Emergency Response

In the event of a critical service situation, emergency response is offered by the Customer Access Support (CAS) main number at **1-800-223-1711** (toll-free in the US), or by calling the Oracle Support hotline for your local country from the list at *http://www.oracle.com/us/support/contact/index.html*. The emergency response provides immediate coverage, automatic escalation, and other features to ensure that the critical situation is resolved as rapidly as possible.

A critical situation is defined as a problem with the installed equipment that severely affects service, traffic, or maintenance capabilities, and requires immediate corrective action. Critical situations affect service and/or system operation resulting in one or several of these situations:

- A total system failure that results in loss of all transaction processing capability
- Significant reduction in system capacity or traffic handling capability
- Loss of the system's ability to perform automatic system reconfiguration
- Inability to restart a processor or the system
- Corruption of system databases that requires service affecting corrective actions
- Loss of access for maintenance or recovery operations
- Loss of the system ability to provide any required critical or major trouble notification

Any other problem severely affecting service, capacity/traffic, billing, and maintenance capabilities may be defined as critical by prior discussion and agreement with Oracle.

Locate Product Documentation on the Oracle Technology Network Site

Oracle customer documentation is available on the web at the Oracle Technology Network (OTN) site, *http://docs.oracle.com*. You do not have to register to access these documents. Viewing these files requires Adobe Acrobat Reader, which can be downloaded at *www.adobe.com*.

1. Log into the Oracle Technology Network site at *http://docs.oracle.com*.
2. Under **Applications**, click the link for **Communications**.
   The **Oracle Communications Documentation** window opens with Tekelec shown near the top.
3. Click **Oracle Communications Documentation for Tekelec Products**.
4. Navigate to your Product and then the Release Number, and click the **View** link (the **Download** link will retrieve the entire documentation set).
5. To download a file to your location, right-click the PDF link and select **Save Target As**.

Chapter 2
Getting Started

Topics:

- *Safety Warnings and Cautions.....14*
- *Electrostatic Discharge (ESD).....14*
- *Accessing the System.....14*

This chapter contains information regarding safety precautions, accessing the system, and logging in for the first time.

Safety Warnings and Cautions

It is important to read this section before attempting any of the hardware installation and maintenance procedures in this guide.

Only trained and qualified personnel should install, activate, and maintain the systems.

Warning:

- During installation, ensure the hardware being worked on is disconnected from the power supply until it is ready to be connected to a power source.
- Always turn OFF all power supplies and unplug all power and external cables before opening, installing, or removing a Tekelec hardware shelf.
- Do not wear loose clothing, jewelry (including rings and chains), or other items that might become trapped in the chassis.

Electrostatic Discharge (ESD)

The Tekelec Subscriber Data Management system contains electrical components which can be damaged by static electricity. Electrostatic discharge (ESD) damage occurs when electronic blades or components are improperly handled, which can result in complete or intermittent system failures. The following can help avoid ESD damage:

Caution: To prevent accidental damage that can be caused by static discharge, always use a grounding wrist strap or other static dissipating device while handling the equipment. Connect the wrist strap to the ESD jack located at the front top right corner of the chassis.

Do not touch components on the blades. Handle the blades only by their edges, face plates or extractor levers. When inserting or removing blades, do not touch any of the components.

Always place the blades with the component side up on an antistatic surface or in a static shielding bag.

Accessing the System

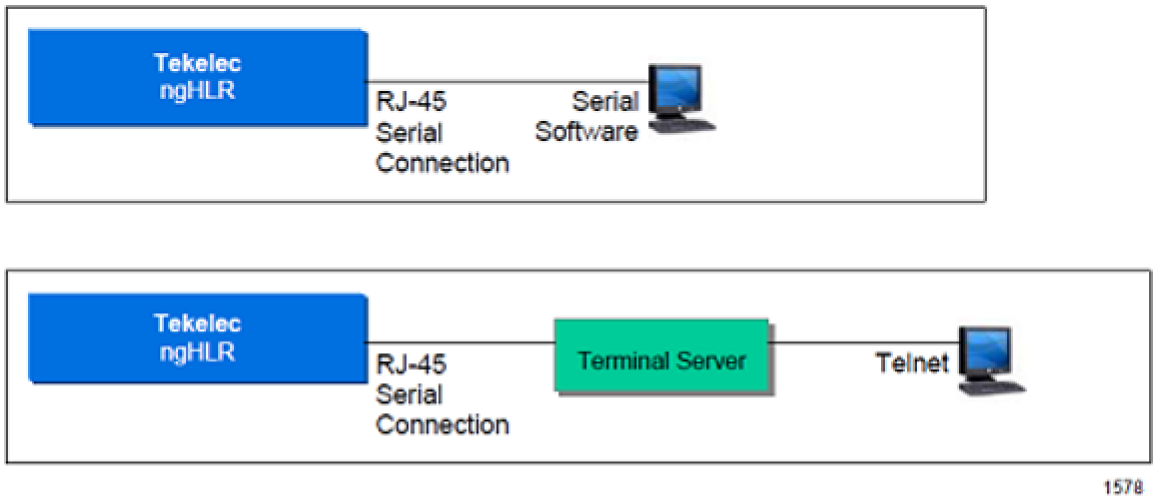
The Operating System and Tekelec Subscriber Data Management software are installed on the system prior to delivery. There are two ways to access the system: SSH client and serial connection.

Establish serial connection

Prerequisites:

- Terminal device with terminal emulation program
- Null-modem serial cable

1. Connect one end of cable to serial console connector on faceplate of Single Board Computer.
2. Connect other end of cable to PC or other terminal device running a terminal emulation program. Or create a Telnet connection via a Terminal server.

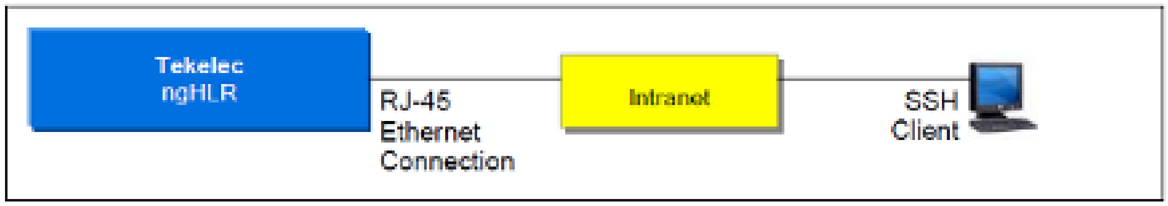


Establish Secure Shell (SSh) connection

Prerequisites

- *Configure SSh client (PuTTY)* (for example, OpenSSH, Cygwin, PuttY)
- Standard CAT 5 Ethernet cable

1. Connect one end of the cable to any one of the three Ethernet RJ-45 ports located on the front faceplate of the switch module.

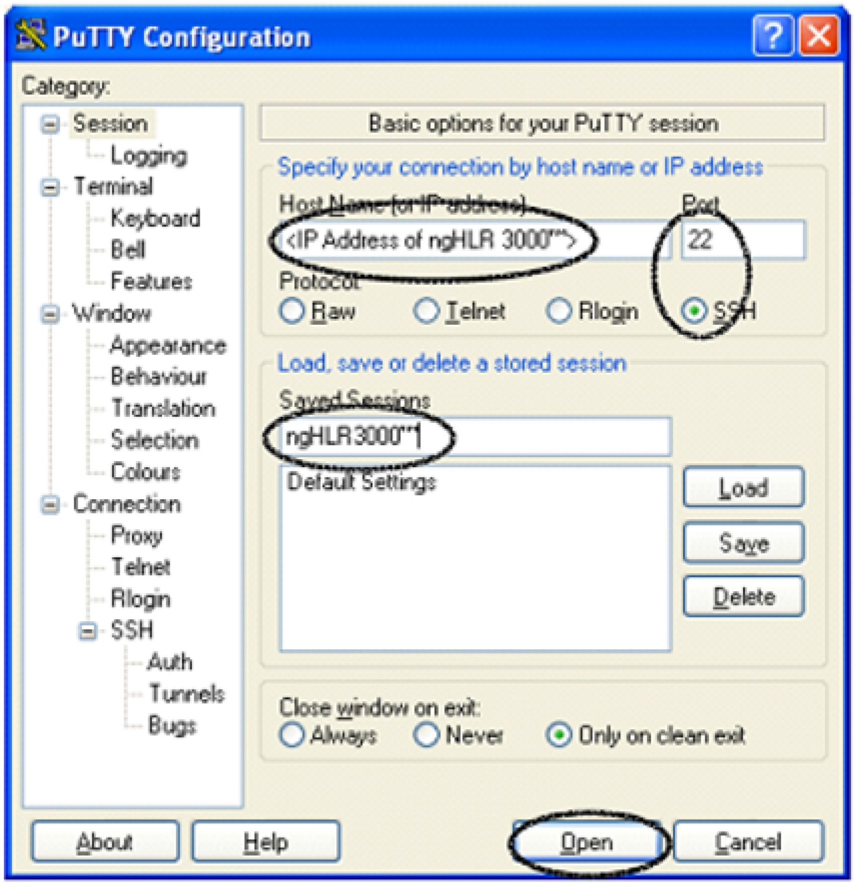


2. Connect other end of cable to PC or other terminal device running a terminal emulation program.
3. Start SSh client.

Configure SSh client (PuTTY)

When using PuTTY as the SSh client and connecting for the first time, install and configure PuTTY.

1. Locate the SDM software CD-ROM, which includes a version of PuTTY for Windows.
2. Copy the PuTTY directory to the system and run PUTTY.EXE.



Host Name (or IP address): <IP Address of ngHLR 3000"> Port: 22

Protocol: Raw Telnet Rlogin [✓ SSH]

Saved Sessions: ngHLR3000"

Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

Figure 1: Configure SSh client (PuTTY)

3. Enter <**`IP Address of the SDM`**> and click SSH protocol.
4. Create session name.
5. Click **Open**

System login

You can access the system by entering a valid user ID and password to the System Controller (SC).

The User Security Management feature introduces Groups, in which users are categorized following their system use and to which access privileges are associated. Only the administrator has all the access privileges and permissions on the system.

At system installation, one default user is predefined for each of the following six predefined groups: user, operation, surveillance, admin, batch and simprov. Each default user is part of a group that has the same name as the user name (example: user admin is part of group admin). The password for each user is the user name by default. A user can be a member of only one user group. The table below displays the users pre-defined at installation, their UserName and UserPasswd and the name of the Group they are associated with.

Table 2: Pre-defined Users

User	Default UserName	Default UserPasswd	Group Name
Administrator	admin	admin	Admin
Surveillance	surveil	surveil	Surveillance
User	user	user	User
Batch	batch	batch	Batch
Operation	operation	operation	Operation
Sim provisioning	simprov	simprov	Simprov

The following groups are pre-defined in the system and categorized based on their system use:

- User: users are responsible for the system configuration and the subscriber provisioning. Typical use is through the WebCi and the Tekelec CLI. On rare occasions, they might also need to log in to the system to access the Tekelec CLI and the Command File Loader services.
- Operation: Operation regroups users responsible for the system operation and maintenance. Typical use is through the WebCi and the Tekelec CLI.
- Surveillance: Surveillance user are the groups involved in managing alarms produced by the system. Typical use is through an external network monitoring system (e.g., HP OpenView) and the Tekelec WebCi.
- Admin: The Administrators are responsible for a set of tasks that requires super user privileges. Their typical use is through the Unix Console
- Simprov: Simprov regroups users that are in charge of provisioning SIM cards.

Only the administrator of the system, already defined in the admin user group, can add users and associate them to one of the ten customizable groups, change its password and provision the groups by editing the services and permissions bind to them, all through the Tekelec CLI or WebCI.

Each group may contain several users and are categorized based on their system use.

Each Group has different access privileges assigned for specific services.

To view the access privileges predefined in the system for each Group, please refer to section 8.3 *User Management* of the *Reference Manual*.

The admin user is for the client set up administrator to access the SDM. Only the administrator of the system can manage the system's blades as well as enter the Tekelec CLI and manage all the Tekelec applications and their services. The administrator can perform a set of tasks that requires super user privileges. The administrator is the only one that can perform anything through the Unix Console.

Log in for the first time

1. Log in with the default username and password of a predefined group.

   Note: The system administrator has superuser permissions and should always be the first person to log in to change and assign passwords.

For example, as administrator, log in as shown below and press **Enter**.

```
login as: admin
password: admin
```

As user, log in as shown below:

```
login as: user
password: user
```

2. At the system prompt, start a CLI session to change the password. Type **`cli`** and press **Enter**.

```
[UserName@system UserName] $ cli
```

3. Go to the Oamp subsystem to change the password; type

```
:> Oamp[]
```

4. Continue to User Management; type

```
Oamp[]> SecurityManager[]
```

5. Specify the user to be modified (e.g., UserName=user2). Type

```
Oamp[]:SecurityManager[]> User [UserName=user2]
```

6. Change the password by using the modify operation and entering the new password. Type

```
Oamp[]:SecurityManager[]> User [UserName=user2]> modify .
Password=Xseries4users]
```

The following message displays:

```
Warning, you are about to modify this instance(s) permanently, Proceed with
modify? (y/[n]):
```

7. Type **y** if you wish to continue or **n** to cancel.
If you typed **y**, the following message displays:

```
Modified:1
```

Command help options

This option displays options available for built-in commands.

Help options show the operator the operations available to perform on the system.

From the directory where the command is stored, type the command name followed by **`-h`** or **`-help`** as shown with the commands below.

Help options are available for commands such as

- **`blueupdate.sh -help`**

- **`cfl -help`** (Command File Loader)
- **`ctl -h`** (Command Template Loader)
- **`CmdTemplateViewer -h`** (Command Template Viewer)

Note: The user must have access privileges to these interfaces and must have logged in successfully before these commands become available.

Blueupdatesh help options

```
/opt/blue/blueupdate.sh –help
blueupdate.sh[-u] [-s] [-k] [-d] [-t dir] [-i interface] [-r release] [-f
[<host:>]<filename>>] [<buildId>]
```

- **`-u:`** uninstall only
- **`-s:`** start software after successful installation
- **`-k:`** keep current database
- **`d:`** use debug load
- **`-t:`** download tarball to given dir but do not install
- **`-i:`** use specified interface
- **`-r:`** use specified release
- **`-f:`** use specified installation file

  `buildId` is ignored if **`-f`** is specified

CFL help options

View the different Command File Loader (CFL) options through this command:

`[UserName@system UserName] $ cfl -help`

CmdFileLoader options:

- **`[-c XmlConfigurationFileName]`** (default: default value)
- **`[-cmd XmlCommand]`** (i.e., submitted inline)
- **`[-d XmlCommandDirectoryName]`**
- **`[-f XmlCommandFileName]`**
- **`[-fo XmlOutputFileName]`** (default: console)

  The **`–fo <XmlOutputFileName.xml>`** tracks the results of the provisioning request, where **`<XmlOutputFileName>`** is the path followed by the name of the XML output file in which you wish the system replies be stored (i.e., `/tmp/template/Xmloutfile1.xml`).

  All system replies are stored in the output file (including error reply codes). Specifying the output file is optional and when no output file name is given, the output is sent automatically to the console by default.

- **`[-ip OampMgrIpAddress]`**
- **`[-p OampManagerPort]`** (default: 62001)
- **`[-reso]`** (produce result not encapsulated in xml and no other messages)
- **`[-trace]`** (traces for errors)
- **`[-user]`** (user name)
- **`[-validate]`** (validate input against the global schema)

Chapter 3

User Interfaces

Topics:

- *Subscriber Provisioning Using XML Templates.....21*
- *Creating and Managing Users for the User Interfaces.....31*
- *Creating and Managing Users for Notifications.....51*

This chapter provides the procedures on how to use the user interfaces that allow the operator to configure the system or provision subscribers.

For an overview of the Command Line Interface (CLI), its commands, the command convention, navigation, and command descriptions, refer to the *User Interfaces* section in the related reference manual.

Subscriber Provisioning Using XML Templates

The Tekelec Oamp Manager is a standard process running on the Tekelec SDM which supports external provisioning and configuration management. This can be done from the Oamp Manager which processes Extensible Markup Language (XML) requests to provision subscriber profiles. The system can process XML Template Invoking files through the following two modes: direct mode and batch mode.

- Direct mode (through a SOAP interface or directly through a TCP socket) accepts XML Template Invoking files, which are processed immediately by the Tekelec system. For more information on the SOAP interface, refer to the “Provisioning in direct mode through the SOAP interface” section of the *SDM Subscriber Provisioning - Reference Manual*.
- Batch mode (through the CFL tool) accepts a file containing XML Template Invoking files, which are then processed. This mode is useful when processing many subscribers (multiple requests) at the same time. For more information on the CFL tool, refer to the “Provisioning in batch mode through the Command File Loader” section of the *SDM Subscriber Provisioning - Reference Manual*.

Refer to the User Interface section of the *SDM Subscriber Provisioning - Reference Manual* for a more detailed description of each mode and for a description of all the types of XML requests supported and their format.

Using TCP socket for provisioning in direct mode

The system can be provisioned by sending XML files directly through a TCP connection.

1. Open a TCP connection on the active System Controller blade using the following setting:

   Table 3: TCP connection commands

Connection Properties	Value
Protocol	TCP/IP
SDM listen IP	Provisioning VIP (configured during SDM installation)
SDM listen port	62001

   Note: There is no prompt to be expected when opening the connection, you can proceed with step 2 right away.

2. Send an authentication request.

   Note: Only one authentication request is required when opening a connection.

   Refer to the “Authentication Operation” section of the *SDM Subscriber Provisioning - Reference Manual* for the format of an XML authentication request.

3. Send the XML files to provision the system.

   Refer to "Subscriber Provisioning using XML Templates" in the *SDM Subscriber Provisioning - Reference Manual* for detailed information on the XML files and templates.

Refer to *Examples of XML Templates for Subscriber Provisioning* for examples on how the XML files should be written.

For every XML request sent, a response will be received including error codes. This response will be in the same format as the request sent (see format described in "Provisioning in direct mode over a TCP socket" in the *SDM Subscriber Provisioning - Reference Manual*). For details on the XML system replies and error codes, refer to "System Replies and Error codes" in the *SDM Subscriber Provisioning - Reference Manual*.

Note: The TCP connection has an inactive period of 20 minutes. After 20 minutes of inactivity, the connection will automatically be closed.

Here are two hexadecimal dumps of what is sent over the TCP socket and what is received when authenticating with the system:

Sending the authentication request

```
0000:0000 00 00 01 49 3c 74 78 3e 3c 72 65 71 20 6e 61 6d ...I<tx><req nam
0000:0010 65 3d 22 6f 70 65 72 61 74 69 6f 6e 22 3e 3c 6f e="operation"><o
0000:0020 70 65 72 20 6e 61 6d 65 3d 22 52 65 71 75 65 73 per name="Reques
0000:0030 74 55 73 65 72 41 75 63 22 20 65 6e 74 3d 22 55 tUserAuc" ent="U
0000:0040 73 65 72 41 75 74 68 65 6e 74 69 63 61 74 69 6f serAuthenticatio
0000:0050 6e 22 20 6e 73 3d 22 62 6e 22 3e 3c 65 78 70 72 n" ns="bn"><expr
0000:0060 3e 3c 70 61 72 61 6d 20 6e 61 6d 65 3d 22 55 73 ><param name="Us
0000:0070 65 72 4e 61 6d 65 22 2f 3e 3c 6f 70 20 76 61 6c erName"/><op val
0000:0080 75 65 3d 22 3d 22 2f 3e 3c 76 61 6c 75 65 20 76 ue="="/><value v
0000:0090 61 6c 3d 22 61 64 6d 69 6e 22 2f 3e 3c 2f 65 78 al="admin"/></ex
0000:00a0 70 72 3e 3c 65 78 70 72 3e 3c 70 61 72 61 6d 20 pr><expr><param 
0000:00b0 6e 61 6d 65 3d 22 55 73 65 72 50 61 73 73 77 64 name="UserPasswd
0000:00c0 22 2f 3e 3c 6f 70 20 76 61 6c 75 65 3d 22 3d 22 "/><op value="="
0000:00d0 2f 3e 3c 76 61 6c 75 65 20 76 61 6c 3d 22 61 64 /><value val="ad
0000:00e0 6d 69 6e 22 2f 3e 3c 2f 65 78 70 72 3e 3c 65 78 min"/></expr><ex
0000:00f0 70 72 3e 3c 70 61 72 61 6d 20 6e 61 6d 65 3d 22 pr><param name="
0000:0100 49 6e 74 65 72 66 61 63 65 4d 6f 64 75 6c 65 49 InterfaceModuleI
0000:0110 64 22 2f 3e 3c 6f 70 20 76 61 6c 75 65 3d 22 3d d"/><op value="=
0000:0120 22 2f 3e 3c 76 61 6c 75 65 20 76 61 6c 3d 22 32 "/><value val="2
0000:0130 37 22 2f 3e 3c 2f 65 78 70 72 3e 3c 2f 6f 70 65 7"/></expr></ope
0000:0140 72 3e 3c 2f 72 65 71 3e 3c 2f 74 78 3e          r></req></tx>
```

Receiving the authentication reply

```
0000:0000 00 00 01 96 3c 74 78 20 6e 62 72 65 71 3d 22 31 ....<tx nbreq="1
0000:0010 22 3e 3c 72 65 71 20 6e 61 6d 65 3d 22 6f 70 65 "><req name="ope
0000:0020 72 61 74 69 6f 6e 22 20 76 65 72 3d 22 35 2e 32 ration" ver="5.2
0000:0030 2e 31 22 20 73 74 61 74 65 3d 22 70 72 6f 63 65 .1" state="proce
0000:0040 73 73 65 64 22 3e 3c 6f 70 65 72 20 6e 61 6d 65 ssed"><oper name
0000:0050 3d 22 52 65 71 75 65 73 74 55 73 65 72 41 75 63 ="RequestUserAuc
0000:0060 22 20 65 6e 74 3d 22 55 73 65 72 41 75 74 68 65 " ent="UserAuthe
0000:0070 6e 74 69 63 61 74 69 6f 6e 22 20 6e 73 3d 22 62 ntication" ns="b
0000:0080 6e 22 3e 3c 65 78 70 72 3e 3c 70 61 72 61 6d 20 n"><expr><param 
0000:0090 6e 61 6d 65 3d 22 55 73 65 72 4e 61 6d 65 22 20 name="UserName" 
0000:00a0 2f 3e 3c 6f 70 20 76 61 6c 75 65 3d 22 3d 22 2f /><op value="="/
0000:00b0 3e 3c 76 61 6c 75 65 20 76 61 6c 3d 22 61 64 6d ><value val="adm
0000:00c0 69 6e 22 2f 3e 3c 2f 65 78 70 72 3e 3c 65 78 70 in"/></expr><exp
0000:00d0 72 3e 3c 70 61 72 61 6d 20 6e 61 6d 65 3d 22 55 r><param name="U
0000:00e0 73 65 72 50 61 73 73 77 64 22 20 2f 3e 3c 6f 70 serPasswd" /><op
0000:00f0 20 76 61 6c 75 65 3d 22 3d 22 2f 3e 3c 76 61 6c  value="="/><val
0000:0100 75 65 20 76 61 6c 3d 22 61 64 6d 69 6e 22 2f 3e ue val="admin"/>
0000:0110 3c 2f 65 78 70 72 3e 3c 65 78 70 72 3e 3c 70 61 </expr><expr><pa
0000:0120 72 61 6d 20 6e 61 6d 65 3d 22 49 6e 74 65 72 66 ram name="Interf
```

```
0000:0130 61 63 65 4d 6f 64 75 6c 65 49 64 22 20 2f 3e 3c aceModuleId" /><
0000:0140 6f 70 20 76 61 6c 75 65 3d 22 3d 22 2f 3e 3c 76 op value="="/><v
0000:0150 61 6c 75 65 20 76 61 6c 3d 22 32 37 22 2f 3e 3c alue val="27"/><
0000:0160 2f 65 78 70 72 3e 3c 2f 6f 70 65 72 3e 3c 72 65 /expr></oper><re
0000:0170 73 20 65 72 72 6f 72 3d 22 30 22 20 61 66 66 65 s error="0" affe
0000:0180 63 74 65 64 3d 22 31 22 3e 3c 2f 72 65 73 3e 3c cted="1"></res><
0000:0190 2f 72 65 71 3e 3c 2f 74 78 3e                   /req></tx>
```

Using SOAP to provision in direct mode

XML requests are sent through XML interface messages over the Internet.

Template and Invoking files must have been completed and uploaded with the Command Template Loader.

1. Establish connection between SOAP application and SDM Web Server.

   Use either of these URLs:

   - http://<system controller IP address>:8080/axis/services/MessageService
   - http://<system controller IP address>:8443/axis/services/MessageService

2. Specify the destination IP address as the external SDM IP address.

   When sending messages, the destination IP address must be specified along with the content of the SOAP message. Moreover, the authentication information must be part of the header of the SOAP message.

   For a description of the SOAP messages, its format and content of header, as well as for examples, refer to the User Interface chapter of the *SDM Subscriber Provisioning - Reference Manual document*.

Bulk and Template Provisioning

The Tekelec OAM&P Manager provides provisioning and configuration management. The OAM&P manager accepts XML commands to handle bulk (multiple) requests as well as Template Invoking files.

- Bulk subscriber provisioning is done by first generating a file that contains XML requests (commands). XML provisioning files can only be written and stored in the /tmp folder. Then the commands stored in the file are processed by the command file loader tool (CmdFileLoader). Refer to the SDM Subscriber Provisioning - Reference Manual for details on the XML file and Request types.
- Subscriber provisioning by templates is done by generating a Template Invoking file that contains XML requests of type -Template‖, which means that are based on Templates. These Templates must have already been defined and loaded in the database using the Command Template Loader tool. Template files can only be written and stored in the /tmp folder. It is also possible to view the templates defined in the database with the Command Template Viewer tool. Refer to the SDM Subscriber Provisioning - Reference Manual for details on the XML Template Invoking file.

Preparing template files

Prior to provisioning the Subscriber Data Management system with templates and using the template tools, all templates must be written and validated.

1. Write all template files and invoking files necessary for provisioning.

   Note: Make sure to have an operation request for authentication, which is required to execute an invoking file with the Command File Loader.

   For XML file syntax and a description of what must be defined in each template, refer to the *SDM Subscriber Provisioning - Reference Manual*; for template examples, refer to the XML template section.

2. Open XML file(s) in an XML editor or reader of your choice to validate the basic XML syntax.

   Internet Explorer is a good example of an XML reader that can be used to perform the validation.

   The XML editor or reader will report any errors in the basic syntax. Correct any errors before continuing.
3. Save the template file(s) and the corresponding invoking file(s) in the `/tmp/templates/` directory, for example, `/tmp/templates/Xmlfile1.xml`.

The files are now ready to be uploaded with the Command Template Loader (CTL).

Running Command Template Loader

Prerequisites:

- User must belong to admin, batch, or operation group.
- Template files must be completed and ready for upload. See *Preparing template files*.

The Command Template Loader loads templates into the system database, which allows the operator to provision subscribers with an Invoking file that executes those templates.

1. *Establish Secure Shell (SSh) connection*.
2. Prior to running the Command Template Loader tool, you must first write the templates you wish to load into the system's database and then save them in the /tmp/templates/ directory

   The operator can then run the Command Template Loader (CTL) tool in order to load these templates and the template requests that constitute them.

3. At the prompt, type

   ```
   [UserName@system UserName] $ ctl -f <XmlInputFileName.xml>
   ```

   where **`XmlInputFileName`** is the path followed by the name of the Invoking file that is to be executed, for example, `/tmp/templates/Xmlfile1.xml`).

4. Load the templates one at a time in this order:
   a) Load the Template files, which contain the Command template and all template requests.
   b) Load the Invoking files.

      The Invoking file references the Template file and then provisions the subscribers with the specific values provided.

The template files are now ready to be viewed or executed by the Command File Loader (CFL)

CTL help options

Display the CmdTemplateLoader (CTL) help option by typing **`ctl -h`**.

These options are available to run the Command Template Loader tool:

- **`[-u]`** → username to access the database (this should be known at installation)

- `[-p]` → password to access the database(this should be known at installation)
- `[-f XmlInputFileName]` → the XML input file name where the templates is stored
- `[-d XmlInputDirectoryName]` → The directory where the <XmlInputFileName> can be found.
- `[-cmd XmlCommand]` → (i.e. submitted inline) It allow you to directly type your XML command instead of using a file.
- `[-del tp/tpr/all id/all]` → other command line params ignored. allow you to delete a template <tp> or a template request <tpr> or both <all> with the specific id.

Important: When performing the delete operation with the `all` option, templates and template requests are deleted without any warning or indication.

- `[-dbip]` → the ip address where the database is running
- `[-trace]` → enables the tool traces to view traces for errors)

Deleting a template from the database

Editing a template requires the deletion of the old template from the database and the reloading of the new template. For example, deleting template with template id =1 that refers to a template request with id=100 and a template request with id=101 requires the deletion of all template requests it refers to, one at a time, by entering the following:

* CmdTemplateLoader Execution time: 0 sec

- `[UserName@system UserName] $ ctl -del tpr 101`
- ** TemplateRequest id[100] has been deleted

*CmdTemplateLoader Execution time: 0 sec

Deleting all the template requests referenced by a Command Template deletes the latter. As a result, the template is deleted automatically.

Note: Keep in mind that executing an Invoking file that refers to the Command Template you just deleted will fail.

You can now load into the database the template file you have just edited by using the Command Template Loader.

1. At the prompt, type

   ```
   [UserName@system UserName] $ ctl -del tpr 100
   ```

   ** TemplateRequest id[100] has been deleted

   * CmdTemplateLoader Execution time: 0 sec

2. Type

   ```
   [UserName@system UserName] $ ctl -del tpr 101
   ```

   Deleting all the template requests referenced by a Command Template deletes the latter. As a result, the template is deleted automatically.

   Note: Executing an Invoking file that refers to a deleted Command Template will fail.

3. Reload the template. See *Running Command Template Loader*

Running Command File Loader

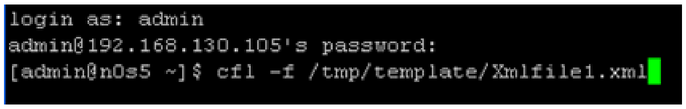
- User must belong to admin, batch, or operation group.
- Template files must have been uploaded. See *Running Command Template Loader*.

The Command File Loader provisions the subscribers in the database by executing the Invoking files, which reference the template requests.

1. *Establish Secure Shell (SSh) connection* to access the blade.
2. At the prompt, type

   `[UserName@system UserName] $ cfl -f <XmlInvokingFileName.xml>` to run the Command File Loader

   Where **`XmlInvokingFileName`** is the path followed by the name of the Invoking file that is to be executed, for example, `/tmp/templates/Xmlfile1.xml`).

   ```
   login as: admin
   admin@192.168.130.105's password:
   [admin@n0s5 ~]$ cfl -f /tmp/template/Xmlfile1.xml
   ```

   The Command File Loader accepts the Invoking file and then executes it by processing each XML request.

   Recommendations for using the Command File Loader

   There are a number of limitations associates with the Command File Loader tool. We recommend the following for optimal performance:

   - Check the input file for XML syntax errors. Errors in the input file can cause the tool to stop processing. The output file will list the transactions that were completed successfully. Failed transactions will need to be resubmitted.
   - Limit the size of the input file to 1.4 GB. Submit small input files in sequence. Large files increase parsing overhead and that consumes CPU resources. The output file is approximately 3 times the size of the input file.
   - Review each output log file. See section "Analyzing system reply" for more information.

CFL help options

View the different Command File Loader (CFL) options through this command:

`[UserName@system UserName] $ cfl -help`

CmdFileLoader options:

- **`[-c XmlConfigurationFileName]`** (default: default value)
- **`[-cmd XmlCommand]`** (i.e., submitted inline)
- **`[-d XmlCommandDirectoryName]`**
- **`[-f XmlCommandFileName]`**
- **`[-fo XmlOutputFileName]`** (default: console)

  The **`–fo <XmlOutputFileName.xml>`** tracks the results of the provisioning request, where **`<XmlOutputFileName>`** is the path followed by the name of the XML output file in which you wish the system replies be stored (i.e., `/tmp/template/Xmloutfile1.xml`).

All system replies are stored in the output file (including error reply codes). Specifying the output file is optional and when no output file name is given, the output is sent automatically to the console by default.

- **`[-ip OampMgrIpAddress]`**
- **`[-p OampManagerPort]`** (default: 62001)
- **`[-reso]`** (produce result not encapsulated in xml and no other messages)
- **`[-trace]`** (traces for errors)
- **`[-user]`** (user name)
- **`[-validate]`** (validate input against the global schema)

Analyzing system reply

To verify the provisioning request, view the output file for any errors. Refer to the "System Replies and Error Codes" section of the *SDM Subscriber Provisioning - Reference Manual* for a description of how to interpret the system replies and their error codes.

1. *Establish Secure Shell (SSh) connection*.
2. Perform one of these steps:
   - If you used the **`-fo`** option and saved the output file, you can view the output file by typing at the prompt

     `view /tmp/template/Xmloutfile1.xml` to view the templates in the database.

     Note: To exit View Mode, type **`q`**
   - If you didn't use the **`-fo`** option, the output data is sent automatically to the console and is only temporarily displayed. Search for the error tag by typing **`/error`**
3. Then search for the error tag by typing: **`/error`**

   a) Look for error responses:

      1. `error ="0"` implies no provisioning error occurred.
      2. error ="code #" implies an error with code # occurred. To view the meaning of the error code #, refer to the Database Error Notifications, HLR Error Notifications, and System Error Notifications in the *SDM Monitoring, Troubleshooting, Maintenance - Reference Manual*.

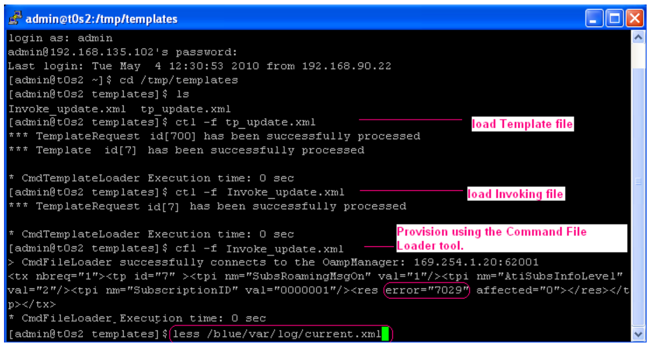
         Note: When provisioning with templates, if an error occurred in one of the template requests, the system returns a general error code that simply informs that an error occurred while processing XML database request (**`error# 7029`**).

   b) View the actual error by typing **`less /blue/var/log/sdmlog.xml`**.

      This command lists the logs recently generated. In one of the logs, you will see the xml file that was processed with the exact error code.

      Example:

      When provisioning the Invoking file with the Command File Loader, the error code 7029 is returned to inform that an error occurred while processing XML database request.

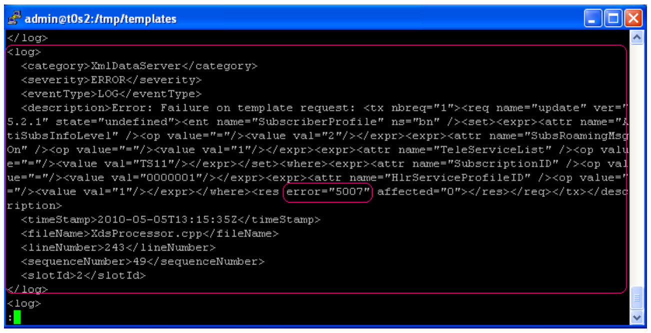

admin@t0s2:/tmp/templates
load Template file
load Invoking file
Provision using the Command File Loader tool.

```
login as: admin
admin@192.168.135.102's password:
Last login: Tue May  4 12:30:53 2010 from 192.168.90.22
[admin@t0s2 ~]$ cd /tmp/templates
[admin@t0s2 templates]$ ls
Invoke_update.xml  tp_update.xml
[admin@t0s2 templates]$ ctl -f tp_update.xml
*** TemplateRequest id[700] has been successfully processed
*** Template  id[7] has been successfully processed

* CmdTemplateLoader Execution time: 0 sec
[admin@t0s2 templates]$ ctl -f Invoke_update.xml
*** TemplateRequest id[7] has been successfully processed

* CmdTemplateLoader Execution time: 0 sec
[admin@t0s2 templates]$ cfl -f Invoke_update.xml
> CmdFileLoader successfully connects to the OampManager: 169.254.1.20:62001
<tx nbreq="1"><tp id="7" ><tpi nm="SubsRoamingMsgOn" val="1"/><tpi nm="AtiSubsInfoLevel"
val="2"/><tpi nm="SubscriptionID" val="0000001"/><res error="7029" affected="0"></res></t
p></tx>
* CmdFileLoader Execution time: 0 sec
[admin@t0s2 templates]$ less /blue/var/log/current.xml
```

c) Look for the log that contains in its description the template you just ran with the Command File Loader.



```
</log>
<log>
  <category>XmlDataServer</category>
  <severity>ERROR</severity>
  <eventType>LOG</eventType>
  <description>Error: Failure on template request: <tx nbreq="1"><req name="update" ver="
5.2.1" state="undefined"><ent name="SubscriberProfile" ns="bn" /><set><expr><attr name="A
tiSubsInfoLevel" /><op value="="/><value val="2"/></expr><expr><attr name="SubsRoamingMsg
On" /><op value="="/><value val="1"/></expr><expr><attr name="TeleServiceList" /><op valu
e="="/><value val="TS11"/></expr></set><where><expr><attr name="SubscriptionID" /><op val
ue="="/><value val="0000001"/></expr><expr><attr name="HlrServiceProfileID" /><op value="
="/><value val="1"/></expr></where><res error="5007" affected="0"></res></req></tx></desc
ription>
  <timeStamp>2010-05-05T13:15:35Z</timeStamp>
  <fileName>XdsProcessor.cpp</fileName>
  <lineNumber>243</lineNumber>
  <sequenceNumber>49</sequenceNumber>
  <slotId>2</slotId>
</log>
<log>
:
```

d) In the log file, another error code will indicate the problem. For a description of the error code returned, refer to the "Error Notifications" section in the *SDM Monitoring, Troubleshooting, Maintenance - Reference Manual*.

4. Enter **q** to exit the log list and return to the prompt.

WARNING

Warning: When using the Tekelec CLI, the `<ctrl> z` command does not send the process execution to background, as it typically would. Since there is no need to allow to run the Tekelec CLI in background, the Tekelec implementation intentionally interprets the `<ctrl> z` command as an "abort" message and suspends the ongoing command. Basically, the use of the `<ctrl> z` command cancels any change made by the ongoing command. In some situations, executing this command may produce a core dump of the Tekelec CLI processes. However, using the `<ctrl> z` command will not cause any service outage, nor will it cause data corruption. The same warning also applies for the use of the `<ctrl> z` command when using the Command File Loader.

XML file syntax using Command File Loader (CFL) for HLR subscriber provisioning

The CFL uses this template (short format) to bulk-provision Tekelec ngHLR subscribers. This Bulk Request file contains insert, update, and delete requests. The requests can be provided on a single line in the XML file. Due to margin limits, the update and delete requests are shown continuing onto subsequent lines as follows.

```
<file>
<ent name = "SubscriberProfile"
ns="bn"><Imsi>31091052100000</Imsi><MsIsdnAlertInd>15634210100</MsIsdnAlertInd></ent>
<ent name = "SubscriberProfile"
ns="bn"><Imsi>31091052100001</Imsi><MsIsdnAlertInd>15634210101</MsIsdnAlertInd></ent>
<ent name = "SubscriberProfile"
ns="bn"><Imsi>31091052100002</Imsi><MsIsdnAlertInd>15634210102</MsIsdnAlertInd></ent>
<ent name = "SubscriberProfile"
ns="bn"><Imsi>31091052100003</Imsi><MsIsdnAlertInd>15634210103</MsIsdnAlertInd></ent>
<ent name = "SubscriberProfile"
ns="bn"><Imsi>31091052100018</Imsi><MsIsdnAlertInd>15634210104</MsIsdnAlertInd></ent>
<ent name = "SubscriberProfile"
ns="bn"><Imsi>31091052100018</Imsi><MsIsdnAlertInd>15634210105</MsIsdnAlertInd></ent>
<ent name = "SubscriberProfile"
ns="bn"><Imsi>31091052100006</Imsi><MsIsdnAlertInd>15634210106</MsIsdnAlertInd></ent>
<req name="update"><ent name = "SubscriberProfile"
ns="bn"/><set><expr><attr name="MsIsdnAlertInd"/> <value val=
"15634210100"/></expr></set><where><expr><attr name="Imsi"/><op
value="="/><value val="0123456789012345"/> </expr></where></req>
<req name="delete"><ent name = "SubscriberProfile"
ns="bn"/><where><expr><attr name="Imsi"/><op value="="/><value val=
"0123456789099999"/></expr></where></req>
</file>
```

XML File Syntax Using Command File Loader for HSS Subscriber Provisioning

The following is an example (short format) of bulk subscriber provisioning for the HSS with a Bulk Request file with insert, update, and delete requests. The requests can be provided on a single line in the XML file. Due to margin limits, the update and delete requests are shown continuing onto subsequent lines as follows. For details on the HSS Subscriber Profile entities, attributes, and their values refer to the *SDM Subscriber Provisioning - Reference Manual*.

```
<file>
<ent name = "HssSubscription"
ns="bn"><SubscriptionID>sub-1</SubscriptionID><ChargingID>ChargingID-1</ChargingID></ent>
<ent name = "HssSubscription"
ns="bn"><SubscriptionID>sub-2</SubscriptionID><ChargingID>
ChargingID-1</ChargingID></ent>
<ent name = "HssSubscription"
ns="bn"><SubscriptionID>sub-3</SubscriptionID><ChargingID>
ChargingID-1</ChargingID></ent>
<ent name = "HssSubscription"
ns="bn"><SubscriptionID>sub-4</SubscriptionID><ChargingID>
ChargingID-2</ChargingID></ent>
<ent name = "HssSubscription"
ns="bn"><SubscriptionID>sub-5</SubscriptionID><ChargingID>
ChargingID-1</ChargingID></ent>
<ent name = "HssSubscription"
ns="bn"><SubscriptionID>sub-6</SubscriptionID><ChargingID>
ChargingID-2</ChargingID></ent>
<ent name = "HssSubscription"
ns="bn"><SubscriptionID>sub-7</SubscriptionID><ChargingID>
```

```
ChargingID-1</ChargingID></ent>
<req name="update"><ent name = "HssSubscription" ns="bn"/><set><expr><attr
 name="ChargingID"/> <value val=
"ChargingID-2"/></expr></set><where><expr><attr name="SubscriptionID"/><op
 value="="/><value val="sub-1"/> </expr></where></req>
<req name="delete"><ent name = "HssSubscription"
ns="bn"/><where><expr><attr name="SubscriptionID"/><op value="="/><value
 val= "sub-7"/></expr></where></req>
</file>
```

Running the Command Template Viewer

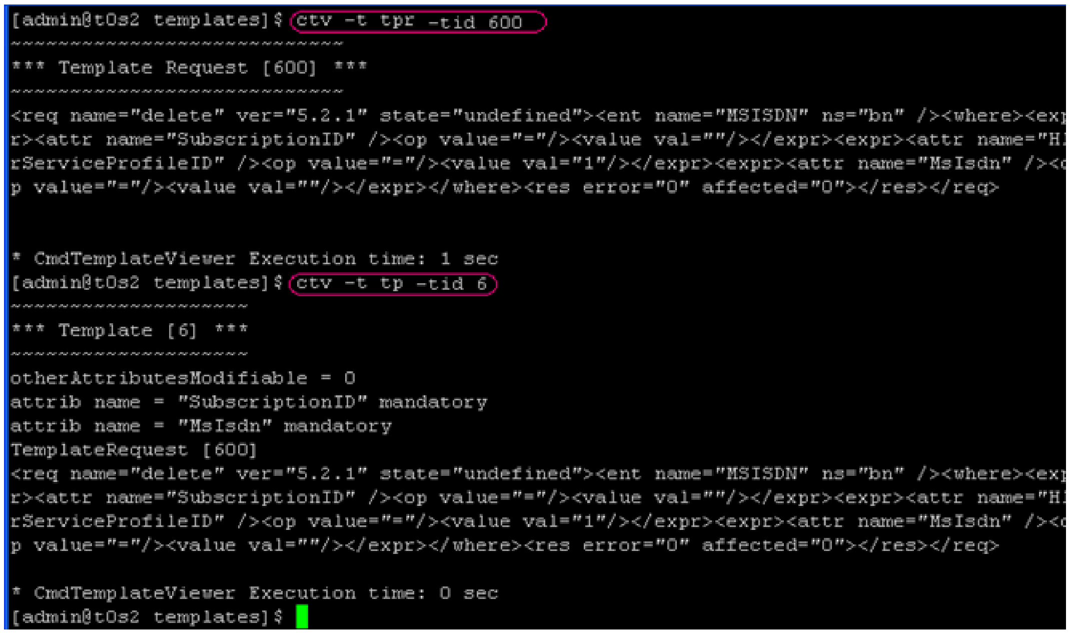
- User must belong to admin, batch, or operation group.
- Template files must have been uploaded. See *Running Command Template Loader*.

The Command Template Viewer (CTV) is used to view the templates currently loaded in the database.

1. *Establish Secure Shell (SSh) connection*.
2. Once the user is logged in the blade, in order to use the Command Template Viewer, the user must write the following:

   `[UserName@system UserName] $ ctv –t <tp/tpr> -id <template id>`
3. At the prompt, type

   `[UserName@system UserName] $ ctv –t <tp/tpr> -id <template id>` to view the templates in the database.



```
[admin@tOs2 templates]$ ctv -t tpr -tid 600
~~~~~~~~~~~~~~~~~~~~~~~~~~~~
*** Template Request [600] ***
~~~~~~~~~~~~~~~~~~~~~~~~~~~~
<req name="delete" ver="5.2.1" state="undefined"><ent name="MSISDN" ns="bn" /><where><ex
r><attr name="SubscriptionID" /><op value="="/><value val=""/></expr><expr><attr name="H
rServiceProfileID" /><op value="="/><value val="1"/></expr><expr><attr name="MsIsdn" /><
p value="="/><value val=""/></expr></where><res error="0" affected="0"></res></req>


* CmdTemplateViewer Execution time: 1 sec
[admin@tOs2 templates]$ ctv -t tp -tid 6
~~~~~~~~~~~~~~~~~~~~
*** Template [6] ***
~~~~~~~~~~~~~~~~~~~~
otherAttributesModifiable = 0
attrib name = "SubscriptionID" mandatory
attrib name = "MsIsdn" mandatory
TemplateRequest [600]
<req name="delete" ver="5.2.1" state="undefined"><ent name="MSISDN" ns="bn" /><where><ex
r><attr name="SubscriptionID" /><op value="="/><value val=""/></expr><expr><attr name="H
rServiceProfileID" /><op value="="/><value val="1"/></expr><expr><attr name="MsIsdn" /><
p value="="/><value val=""/></expr></where><res error="0" affected="0"></res></req>

* CmdTemplateViewer Execution time: 0 sec
[admin@tOs2 templates]$
```

The Command File Loader accepts the Invoking file and then executes it by processing each XML request.

CTV help options

Display the help options of the Command Template Viewer by typing **`CmdTemplateViewer -h`**

This command displays the options when running the Command Template Loader:

- **`[-u]`** → username to access the database (this should be known at installation)
- **`[-p]`** → password to access the database(this should be known at installation)
- **`[-t tp/tpr]`** → specifies if you want to view a complete template <tp> or a template request <tpr> (i.e. template/template request)
- **`[-tid (template id)]`** → the id of the target template or template request (digits only)
- **`[-dbip]`** → the ip address where the database is running

XML Template Invoking File Syntax for HLR Subscriber Provisioning

Once the Templates have been defined and loaded onto the database of the system using the Command Template Loader, it is possible to provision subscribers by creating Template Invoking file(s) that refer to those templates. The following shows an example of a Template Invoking file that refers to the template with an Id=1. For more information on the Template Invoking file fields and attributes, please refer to the *SDM Subscriber Provisioning - Reference Manual*. Refer to that same document for details on the HLR, SIP, HSS and AAA Subscriber Profile entities, attributes, and their values.

```
<tp id="1">
             <tpi nm="Imsi" val="310910421000100"/>
             <tpi nm="MsIsdn" val="15634210100"/>
             <tpi nm="MsIsdnAlertInd" val="15634210100"/>
             <tpi nm="DefaultFtn" val="15634213333"/>
             <tpi nm="SimId" val="234445666000"/>
</tp>
```

Creating and Managing Users for the User Interfaces

With the USM functionality, the SDM user interfaces (CLI, WebCI, XML interfaces) support multiple types of user accounts, which can be managed only by the administrator.

The administrator modifies the access privileges table as needed, for example, when an existing group requires modification.

Table 4: Access Privileges shows the predefined access privileges entries associated to the six predefined groups (user, operation, surveil, admin, batch, simprov):

Table 4: Access Privileges

Services/Group	User	Operation	Surveillance	Admin	Batch	Simprov
System	R	RWX	R	RWX		
OAMP	R	R	R	RWX	R	
Database		RWX		RWX		
HLR subscriber prov	RWX			RWX	RWX	

SIM provisioning	RWX			RWX	RWX	RWX
HLR configuration	RWX		R	RWX		
SS7 configuration	RWX		R	RWX		
SIP subscriber prov	RWX			RWX	RWX	
SIP configuration	RWX		R	RWX		
HSS subscriber prov	RWX			RWX	RWX	
HSS configuration	RWX		R	RWX		
ExternalService				RWX	RX	
Subscriber Provisioning	RWX			RWX	RWX	
Schema				RWX		
Policy				RWX		

R: Read (Display) W: Write (Add/Modify/Delete) X: eXecute (Access to entity own operations)

Each user belongs to a specific Group that has a specific access to the system's services (entities). Only the Admin Group has full access privileges. The admin user is able to:

- View and modify operational aspects of the system
- Add, delete, modify, and delete subscriber provisioning information
- View and acknowledge system alarms
- View system logs, and performance measurements

Changes made to the system configuration or subscriber provisioning data takes effect immediately. There is no rollback mechanism.

User management using CLI

User management procedures provision users, groups, access privileges, and services. The operator manages users through the CLI at the Security Manager level by performing the following procedures and using the indicated tables:

- Provision users in the User table
- Provision groups in the Group table
- Provision access privileges in the AccessPrivileges table
- Provision services in the Service table

Provisioning users in the User table

The User table defines users and includes user name, password, and group name. Users must be provisioned first in the User table. Users belonging to the Admin group can add, display, modify, and delete users. The exact operations depend on the user interface.

Table 5: User table operations per user interface

CLI	WebCI
Add user	Add user
Modify user	Modify user
Display user	Delete user

All users can change (modify) their passwords. The exact permissions per user group depend on the settings defined in *Creating and Managing Users for the User Interfaces*.

Create User from the CLI

The following CLI procedure shows how to create a user and give it a username and password. This procedure can only be done by the administrator and is recommended to be one of the first things to do once a CLI session is started for the first time. For details on the User parameters, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*. To perform the task in the following table, refer to the procedure for the step by step instructions.

1. Go to the Oamp subsystem by typing,

```
:> Oamp[]
```

2. Go to User Management by typing,

```
Oamp[]> SecurityManager[]
```

3. Add a user as shown below by specifying the UserName and UserPasswd you wish to attribute to the user (i.e. UserName:user2, UserPasswd: #Xseries4users)

```
Oamp[]:SecurityManager[]> add User [UserName=user2; UserPasswd=
#Xseries4users]
```

The following message will be displayed.

```
Added: 1
```

Display User from the CLI

The following CLI procedure displays the steps on how to view the User table, its UserName, UserId and GroupId. It is important to note that the UserPasswd is confidential and cannot be viewed.

1. Go to the Oamp subsystem by typing,

```
:> Oamp[]
```

2. Go to User Management by typing,

```
Oamp[]> SecurityManager[]
```

3. To display the entire User Table, go to User without specifying any attributes, as follows:

```
Oamp[]> SecurityManager[]> User[]
```

   To display the User Table only for specific users, go to User and specify the user's name (i.e. UserName: user):

```
Oamp[]> SecurityManager[]> User[UserName=user]
```

4. Display the User Table by typing:

```
Oamp[]> SecurityManager[]> User[]> display
or
Oamp[]> SecurityManager[]> User[UserName=user]> display
```

   Information similar to the following will be displayed if you displayed the entire User Table:

```
|  UserName|  UserPasswd       |  GroupName     |
-----------------------|-----------------------------
user         |                   |    user        |
operation    |                   |    operation   |
surveil      |                   |    surveil     |
admin        |                   |    admin       |
batch        |                   |    batch       |
simprov      |                   |    simprov     |
user2        |                   |    user        |
```

```
Displayed: 7
```

   Information similar to the following will be displayed if you only displayed a specific user:

```
|UserName     |UserPasswd          |GroupName|
------------------------------------
user            |                     |    user |
Displayed: 1
```

Modify user from the CLI

Each user can modify the user password by entering a new value of the password in the User[ ] entity.

Only the system administrator can modify:

- The password of each user.
- The group to which a user is associated by modifying the GroupName field.
- The UpgradeMode and the PersistOS (whether to store the user information in the Operating System)

This procedure describes how to modify any user information from the User[ ] entity. For details on User parameters, refer to the *User Management through CLI* section of the *SDM System Configuration - Reference Manual*.

1. Go to the Oamp subsystem by typing

```
:> Oamp[]
```

2. Go to User Management by typing

```
Oamp[]> SecurityManager[]
```

3. Specify the user for which you wish to modify information (i.e., UserName=user2). Type

```
Oamp[]:SecurityManager[]> User [UserName=user2]
```

4. Modify the user specified in the previous step by executing the modify command and specifying the fields you wish to modify and their new values. For example, type

```
Oamp[]:SecurityManager[]> User [UserName=user2]> modify .
Password=Xseries4users]
```

   The following message displays:

```
Warning, you are about to modify this instance(s) permanently, Proceed with
modify? (y/[n]):
```

5. Type **y** if you wish to continue or **n** if you wish to cancel
   If you typed **y**, the following message displays:

```
Modified:1
```

Provisioning groups in the Group table

The Group table defines a user group based on system use and common access privileges and permissions. Each group consists of a group name and the access granted for each service.

The Subscriber Data Management system pre-defines six groups with certain access privileges for each service: user, operation, surveil, admin, batch, and simprov. The user group can be displayed by all users, but only the administrator can display, add, modify, or delete access privileges of each service associated to the group. See also *Table 4: Access Privileges*.

Provisioning groups from the CLI

The following CLI procedure is a generic procedure on how to provision the Group entity. Follow this logic whether you wish to add/display/modify/delete an entry in the Group[ ] entity.

1. Go to the Oamp subsystem by typing,

```
:> Oamp[]
```

2. Go to User Management by typing,

```
Oamp[]> SecurityManager[]
```

3. From here, you can add/display/delete/modify an entry in the Group entity. Perform one of the following actions, as needed:
   - To display the Group entity, perform the following:

```
Oamp[]> SecurityManager[]> display Group[]
```

   - To display a specific entry in the Group entity, simply specify the GroupName of the group you wish to display, as follows:

```
Oamp[]> SecurityManager[]> display Group[GroupName=user]
```

   - To add an entry, perform the following (i.e.: GroupName: usergroup1):

```
Oamp[]> SecurityManager[]> add Group[GroupName=usergroup1]
```

   - To modify an entry, specify the GroupName of the group you wish to modify and perform the 'modify' command with the attributes you wish to modify and their new values (i.e.: modify the PersistOS from '0' (Off) to '1' (On):

```
Oamp[]> SecurityManager[]> Group[GroupName=usergroup1]> modify . PersistOS=1
```

   - To delete an entry, simply specify the GroupName of the group you wish to delete, as follows:

```
Oamp[]> SecurityManager[]> delete Group[GroupName=usergroup1]
```

4. Depending on the action taken in the previous step, the CLI will return one of the following messages:

```
This Command could potentially display a very large number of instances.
Proceed with display? (y/[n]):

or

Warning, you are about to modify this instance(s) permanently, Proceed with
modify? (y/[n]):
```

5. Type in '**y**' if you wish to continue or '**n**' if you wish to cancel.
6. If you typed '**y**', the result will be displayed.

Provisioning access privileges in the Access Privileges table

The Access Privileges table defines access privileges for a user group by making an association between a user group, a service, and access permissions. Each access privilege gives a single group access permission to a single service. Only the administrator can modify the AccessPrivileges table.

Each service has its own associated entities based on functionalities; see Table *Predefined services and associated entities* in the *SDM System Configuration Reference Manual*. All services are predefined in the system with these access permissions: Read, Write, and Execute. Services cannot be created or modified.

The Subscriber Data Management system pre-defines six groups with certain access privileges for each service: user, operation, surveil, admin, batch, and simprov. The user group can be displayed by all users, but only the administrator can display, add, modify, or delete access privileges of each service associated to the group. See also *Creating and Managing Users for the User Interfaces*.

These access privileges operations are available per user interface and associated table:

Table 6: Access privileges operations per user interface

CLI **SecurityAccessPrivileges table**	**WebCI** **UserAccessPrivileges table**
Display access privileges	Display access privileges
Modify access privileges	Modify access privileges
	Display All Groups

Displaying access privileges from the CLI

This CLI procedure displays the AccessPrivileges table with GroupId, ServiceId, and Permission.

1. Go to the Oamp subsystem by typing **`:> Oamp[]`**
2. Continue to User Management by typing **`SecurityManager[]`**
3. Continue to Group and specify the GroupName (i.e., GroupName: admin) to display its associated AccessPrivileges table. Type **`Group[GroupName=admin]`**
4. To display the entire AccessPrivileges table associated to the specified Group, type **`display SecurityAccessPrivileges []`**
   The complete syntax is shown below:

```
Oamp[]> SecurityManager[]>Group[GroupName=admin]> display SecurityAccessPrivileges
 []
```

   To display a specific entry of the SecurityAccessPrivileges table for the specified Group, specify the ServiceId, Permission, or both attributes, for example, type **`display SecurityAccessPrivileges [ServiceName=HlrConfig]`**

   - If you displayed the entire AccessPrivileges table, the system returns information similar to the one shown below:

```
ServiceName      | GroupName |   Permission   |
-------------------------------------
 Database        |  admin    |ReadWriteExecute|
 ExternalService |  admin    |ReadWriteExecute|
 HlrConfig       |  admin    |ReadWriteExecute|
 HlrSimProv      |  admin    |ReadWriteExecute|
 HlrSubsProv     |  admin    |ReadWriteExecute|
 HssConfig       |  admin    |ReadWriteExecute|
 Oamp            |  admin    |ReadWriteExecute|
 Policy          |  admin    |ReadWriteExecute|
 Schema          |  admin    |ReadWriteExecute|
 SipConfig       |  admin    |ReadWriteExecute|
 SipSubsProv     |  admin    |ReadWriteExecute|
 Ss7Config       |  admin    |ReadWriteExecute|
 subscriberProv  |  admin    |ReadWriteExecute|
 System          |  admin    |ReadWriteExecute|
 user            |HssSubsProv|ReadWriteExecute|

Displayed: 15
```

- If you displayed only a specific group, the system returns information similar to the one shown below:

```
GroupName: admin
ServiceName: HlrConfig
Permission: ReadWriteExecute
```

Modify AccessPrivileges from the CLI

This procedure describes how to modify the SecurityAccessPrivileges table. The only modifiable attribute of the SecurityAccessPrivileges table is the Permission attribute.

1. Go to the Oamp subsystem by typing

```
:> Oamp[]
```

2. Go to User Management by typing

```
Oamp[]> SecurityManager[]
```

3. Go to Group and specify the GroupName (i.e., GroupName: batch) of the Group for which you would like to modify the associated AccessPrivileges table.

```
Oamp[]> SecurityManager[]> Group[GroupName=batch]
```

4. Go to AccessPrivileges by typing

```
Oamp[]> SecurityManager[]> Group[GroupName=batch]> SecurityAccessPrivileges
[]
```

5. Modify the AccessPrivileges table by specifying the Permission attribute and providing its new value (i.e, Permission: ReadWriteExecute):

```
Oamp[]> SecurityManager[]> Group[GroupName=batch]> SecurityAccessPrivileges []>
 modify . Permission=ReadWriteExecute
```

The following warning displays:

```
Warning, you are about to modify this instance(s) permanently, Proceed with
modify? (y/[n]):
```

6. Type **y** to proceed.

   The following message displays:

```
Modified: 1
```

Provisioning services in the Service table

The Service table adds, displays, modifies, and deletes external services.

The Subscriber Data Management system pre-defines internal services and their associated entities; see Table *Predefined services and associated entities* in the *SDM System Configuration Reference Manual*. Any user can display these services within the Service table.

WARNING

Warning: Pre-defined services cannot be deleted by any user (including the system administrator) because deleting these internal services could impact the system.

The system administrator can define other services for external entities by adding them manually to the Global Schema. To assign external entities to a newly defined service, the system administrator must define the association when creating a new entity in the Global Schema (contact the Tekelec *My Oracle Support (MOS)* for assistance).

The system administrator can then

- modify the description given to these services
- delete the newly added services

Provisioning services from the CLI

This procedure describes how to provision the Service[ ] entity using the add, display, modify, or delete operation.

1. Go to the Oamp subsystem by typing

```
:> Oamp[]
```

2. Go to User Management by typing

```
Oamp[]> SecurityManager[]
```

3. Perform one of the following actions:

   - To display the Service entity, type **display**:

```
Oamp[]> SecurityManager[]> display Service[]
```

- To display a specific entry in the Service entity, type **`display`** and specify the ServiceName of the service to display:

```
Oamp[]> SecurityManager[]> display Service[ServiceName=HlrConfig]
```

- To add an entry, type **`add`** and specify the ServiceName with the ExternalService value, which regroups the ExternalServiceManager entity):

```
Oamp[]> SecurityManager[]> add Service[ServiceName=ExternalService]
```

- To modify an entry, type **`modify`** and specify the ServiceName of the group to modify:

```
Oamp[]> SecurityManager[]> Service[ServiceName=ExternalService]> modify .
Description=service regrouping external services
```

- To delete an entry, specify the ServiceName of the service to delete:

```
Oamp[]> SecurityManager[]> delete Service[ServiceName=ExternalService]
```

WARNING

Warning: The pre-defined services cannot be deleted by any user (including the system administrator) since these are internal services and a deletion could impact the system.

Depending on the action taken in the previous step, the CLI will return one of the following messages:

```
This Command could potentially display a very large number of instances.
Proceed with display? (y/[n]):

Warning, you are about to modify this instance(s) permanently, Proceed with
modify? (y/[n]):
```

4. Type **y** if you wish to continue; type **n** if you wish to cancel.
   If you typed **y**, the result displays.

User management using WebCI

User management procedures provision users, groups, access privileges, and services. The operator manages users through the WebCI in the User Manager by performing the following procedures and using the indicated windows or tables:

- View all User Management information in the User Manager window
- Provision users in the User table
- Provision groups in the Group table
- Provision access privileges in the AccessPrivileges table
- Provision services in the Service table

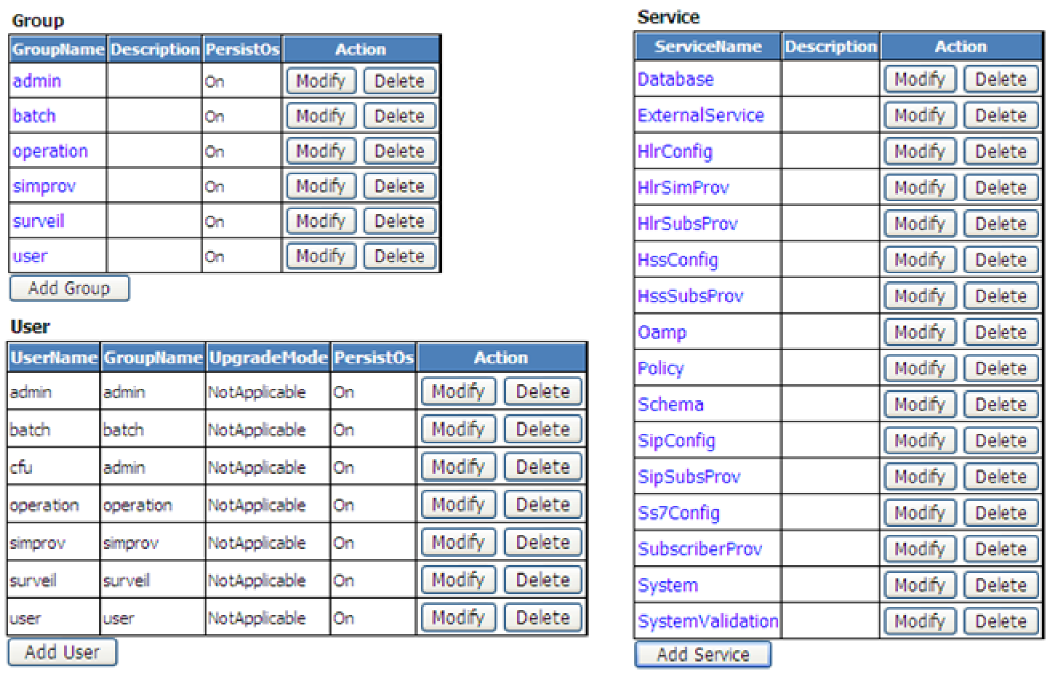
View all User Management Information from the WebCI

This procedure describes how to display the User Manager window.

The User Manager window displays all user management information.

1. From the main menu, go to **Oamp** > **UserManager**.

   The User Manager window displays the tables required for user management provisioning.



Group

GroupName	Description	PersistOs	Action	
admin		On	Modify	Delete
batch		On	Modify	Delete
operation		On	Modify	Delete
simprov		On	Modify	Delete
surveil		On	Modify	Delete
user		On	Modify	Delete

User

UserName	GroupName	UpgradeMode	PersistOs	Action	
admin	admin	NotApplicable	On	Modify	Delete
batch	batch	NotApplicable	On	Modify	Delete
cfu	admin	NotApplicable	On	Modify	Delete
operation	operation	NotApplicable	On	Modify	Delete
simprov	simprov	NotApplicable	On	Modify	Delete
surveil	surveil	NotApplicable	On	Modify	Delete
user	user	NotApplicable	On	Modify	Delete

Service

ServiceName	Description	Action	
Database		Modify	Delete
ExternalService		Modify	Delete
HlrConfig		Modify	Delete
HlrSimProv		Modify	Delete
HlrSubsProv		Modify	Delete
HssConfig		Modify	Delete
HssSubsProv		Modify	Delete
Oamp		Modify	Delete
Policy		Modify	Delete
Schema		Modify	Delete
SipConfig		Modify	Delete
SipSubsProv		Modify	Delete
Ss7Config		Modify	Delete
SubscriberProv		Modify	Delete
System		Modify	Delete
SystemValidation		Modify	Delete

Figure 2: User Manager Window

2. View additional tables by clicking the hyperlinks.
   - To view the AccessPrivileges table for a Group name, click the Group name in the Group table. To return to the User Manager window, click the **Display All Groups** button.
   - To view Services parameters, click the Services name in the Service table. To return to the User Manager window, click the **Display All Services** button.

Provisioning users in the User table

The User table defines users and includes user name, password, and group name. Users must be provisioned first in the User table. Users belonging to the Admin group can add, display, modify, and delete users. The exact operations depend on the user interface.

Table 7: User table operations per user interface

CLI	WebCI
Add user	Add user
Modify user	Modify user
Display user	Delete user

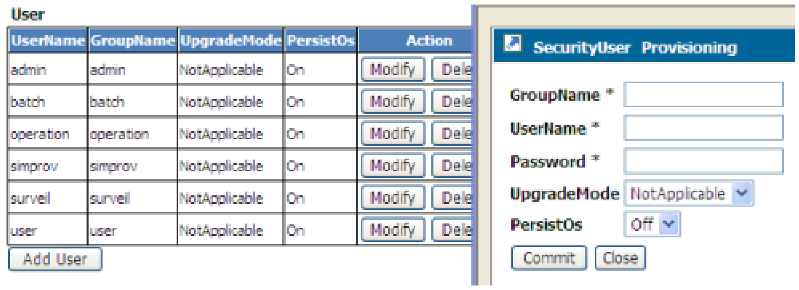
All users can change (modify) their passwords. The exact permissions per user group depend on the settings defined in *Creating and Managing Users for the User Interfaces*.

Creating User from the WebCI

The following WebCI procedure shows how to create a user, give it a username and password, and associate a group to it. This procedure can only be done by the administrator and is

recommended to be one of the first things to do once a WebCI session is started for the first time. For details on the User parameters, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*.

1. From the main menu, navigate to **Oamp** > **User Manager**.
   The User Manager window displays.
2. Click the **Add User** button below the User table.
   The User Provisioning pop-up window displays.



User

UserName	GroupName	UpgradeMode	PersistOs	Action	
admin	admin	NotApplicable	On	Modify	Dele
batch	batch	NotApplicable	On	Modify	Dele
operation	operation	NotApplicable	On	Modify	Dele
simprov	simprov	NotApplicable	On	Modify	Dele
surveil	surveil	NotApplicable	On	Modify	Dele
user	user	NotApplicable	On	Modify	Dele

SecurityUser Provisioning

GroupName *	
UserName *	
Password *	
UpgradeMode	NotApplicable
PersistOs	Off

3. Enter the information to be added (the asterisk (*) identifies a mandatory attribute).
4. Click **Commit**.
   The system returns a confirmation message `User entry was successfully committed`
5. Click **OK**.

Modifying a User from the WebCI

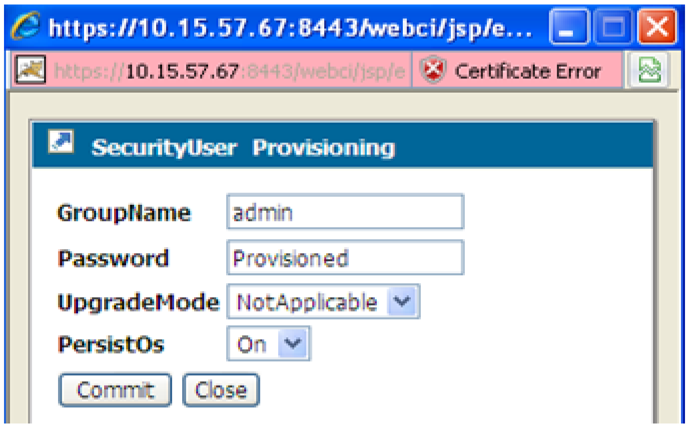
Each user can modify its own password by modifying the User[ ] entity and entering the value of the new password desired. However, only the administrator of the system can modify the following:

- The password of each user.
- The group to which a user is associated to, by modifying the GroupName field.
- The UpgradeMode and the PersistOS (to store or not the user information in the Operating System)

The following WebCI procedure shows how to modify any user information from the User[ ] entity. For details on the User parameters, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual.* To perform the task in the following table, refer to the procedure for the step by step instructions.

1. From the main menu, navigate to **Oamp** > **User Manager**. This will display the User Manager window (as shown in the figure below)
2. Click the **Modify** button beside the User entry of the User you wish to modify.
3. When the User Provisioning window appears (see figure below), enter the new value(s).

   Note: The figure below is for the administrator of the system. The other users can only modify the Password.


https://10.15.57.67:8443/webci/jsp/e...
https://10.15.57.67:8443/webci/jsp/e
Certificate Error
SecurityUser Provisioning

GroupName: admin
Password: Provisioned
UpgradeMode: NotApplicable
PersistOs: On

Figure 3: User Provisioning Window to Modify a User

4. Click **Commit** when all the information has been entered.
5. When the confirmation message "`UserProfile entry was successfully committed`" appears, click **OK**.

Deleting a User from the WebCI

The administrator can delete a user entry from the WebCI. The following WebCI procedure shows how to modify any user information from the User[ ] entity. For details on the User parameters, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual.* To perform the task in the following table, refer to the procedure for the step-by-step instructions.

1. From the main menu, navigate to **Oamp** > **User Manager**. This will display the User Manager window.
2. Click the **Delete** button beside the User entry of the User you wish to delete
3. The following message is returned: "`You are about to delete this entry, Continue?`"
4. Click '**OK**' if you wish to continue or '**Cancel**' otherwise.

Provisioning groups in the Group table

The Group table defines a user group based on system use and common access privileges and permissions. Each group consists of a group name and the access granted for each service.

The Subscriber Data Management system pre-defines six groups with certain access privileges for each service: user, operation, surveil, admin, batch, and simprov. The user group can be displayed by all users, but only the administrator can display, add, modify, or delete access privileges of each service associated to the group. See also *Table 4: Access Privileges*.

Display groups from the WebCI

This procedure describes the steps to view the Group table, all of the groups defined in the system. For details on the Group attributes, refer to the “User Management through CLI” section of the *SDM System Configuration - Reference Manual*.

1. From the main menu, navigate to **Oamp** > **User Manager**.
2. This will display the User Manager window (as shown in User Manager window).

   The Group table displays all of the groups defined in the system.

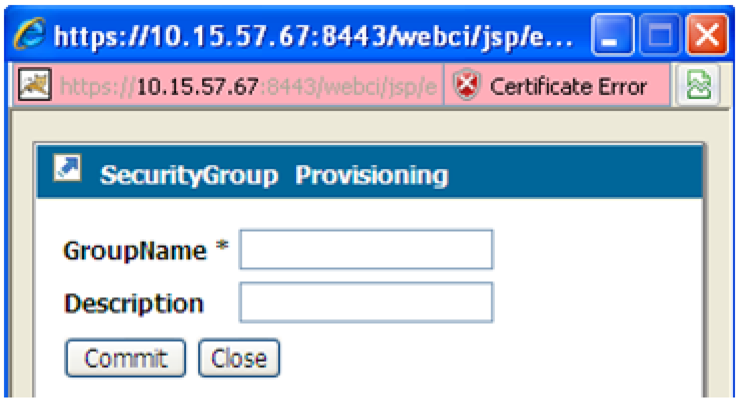
Create a group from the WebCI

This procedure creates a group and defines access privileges by provisioning permissions for each service.

Only the system administrator has permission to perform this procedure.

1. From the main menu, navigate to **Oamp** > **User Manager**.

   The User Manager window displays.
2. Click the **Add Group** button below the Group table.
   The SecurityGroup Provisioning pop-up window displays.


https://10.15.57.67:8443/webci/jsp/e...
https://10.15.57.67:8443/webci/jsp/e
Certificate Error
SecurityGroup Provisioning
GroupName *
Description
Commit
Close

3. Enter the information; the asterisk (*) identifies a mandatory attribute.
4. Click **Commit**.
   The system returns a confirmation message `User entry was successfully committed`
5. Click **OK**.

Modifying a group from the WebCI

The administrator of the system i s the only one that can modify the groups and the description is the only field that can be modified from the Group table.

The following WebCI procedure shows how to modify a group entry from the Group[ ] entity. For details on the User parameters, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*. To perform the task in the following table, refer to the procedure for the step by step instructions.

1. From the main menu, navigate to **Oamp** > **User Manager**. This will display the User Manager window.
2. Click the **Modify** button beside the Group entry of the group you wish to modify.
3. When the Group Provisioning window appears (see figure below), enter the new description you wish to give to the group.

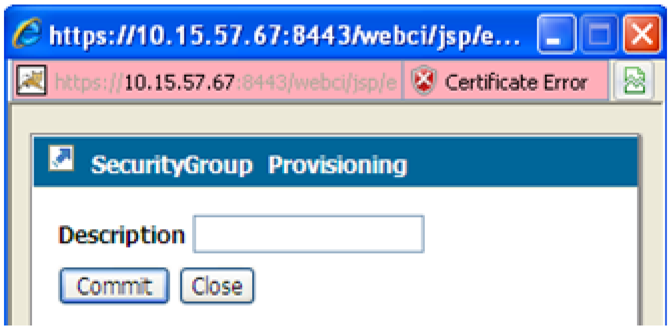

https://10.15.57.67:8443/webci/jsp/e...
https://10.15.57.67:8443/webci/jsp/e
Certificate Error
SecurityGroup Provisioning
Description
Commit
Close

Figure 4: Group Provisioning Window to Modify a Group

4. Click **Commit** when all the information has been entered.
5. When the confirmation message "`Entity entry was successfully committed`" appears, click **OK**.

Delete a group from the WebCI

The administrator of the system can delete a group entry from the WebCI's Group table. The following WebCI procedure shows how to delete a group from the Group[ ] entity. For details on the Group parameters, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*. To perform the task in the following table, refer to the procedure for the step by step instructions.

1. From the main menu, navigate to **Oamp** > **User Manager**. This will display the User Manager window.
2. Click the **Delete** button beside the Group entry of the group you wish to delete.

   The following message is returned: "`You are about to delete this entry, Continue?`"
3. Click '**OK**' if you wish to continue or '**Cancel**' otherwise.

Provisioning access privileges in the Access Privileges table

The Access Privileges table defines access privileges for a user group by making an association between a user group, a service, and access permissions. Each access privilege gives a single group access permission to a single service. Only the administrator can modify the AccessPrivileges table.

Each service has its own associated entities based on functionalities; see Table *Predefined services and associated entities* in the *SDM System Configuration Reference Manual*. All services are predefined in the system with these access permissions: Read, Write, and Execute. Services cannot be created or modified.

The Subscriber Data Management system pre-defines six groups with certain access privileges for each service: user, operation, surveil, admin, batch, and simprov. The user group can be displayed by all users, but only the administrator can display, add, modify, or delete access privileges of each service associated to the group. See also *Creating and Managing Users for the User Interfaces*.

These access privileges operations are available per user interface and associated table:

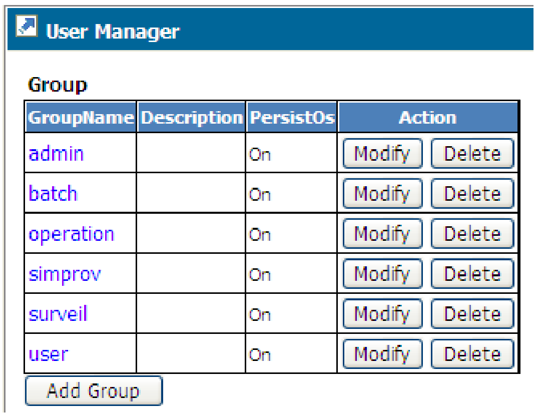
Table 8: Access privileges operations per user interface

CLI SecurityAccessPrivileges table	WebCI UserAccessPrivileges table
Display access privileges	Display access privileges
Modify access privileges	Modify access privileges
	Display All Groups

Display AccessPrivileges table from the WebCI

This procedure describes how to display the AccessPrivileges table associated to a specific Group.

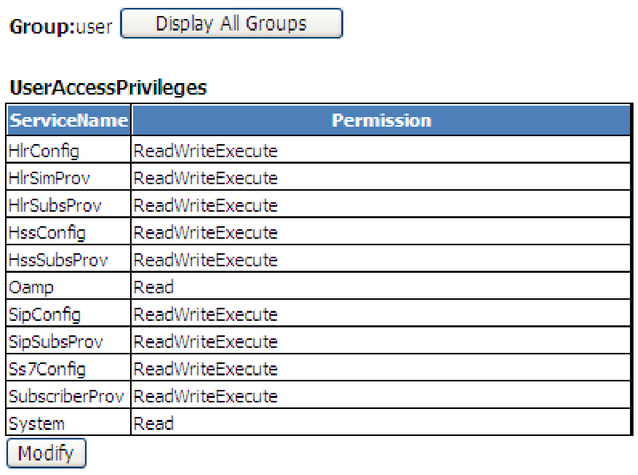
1. From the main menu, navigate to **Oamp** > **User Manager**.
   The User Manager window displays the Group table.



Group

GroupName	Description	PersistOs	Action
admin		On	Modify Delete
batch		On	Modify Delete
operation		On	Modify Delete
simprov		On	Modify Delete
surveil		On	Modify Delete
user		On	Modify Delete

2. In the Group table, click the GroupName hyperlink (blue) to display the associated AccessPrivileges table.
   The UserAccessPrivileges Provisioning window displays service names and permissions.


Group:user

UserAccessPrivileges

ServiceName	Permission
HlrConfig	ReadWriteExecute
HlrSimProv	ReadWriteExecute
HlrSubsProv	ReadWriteExecute
HssConfig	ReadWriteExecute
HssSubsProv	ReadWriteExecute
Oamp	Read
SipConfig	ReadWriteExecute
SipSubsProv	ReadWriteExecute
Ss7Config	ReadWriteExecute
SubscriberProv	ReadWriteExecute
System	Read

3. Click **Display All Groups** to return to the User Manager window.

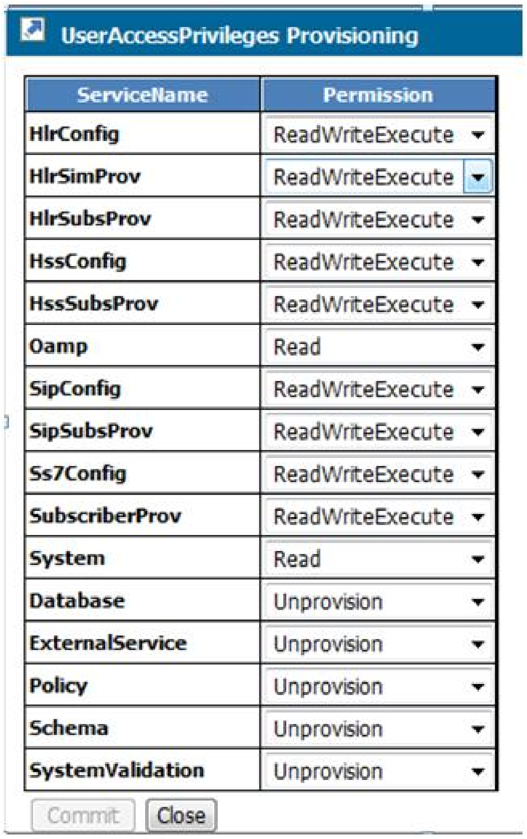
Modify Access Privileges from the WebCI

This procedure describes the steps to modify the AccessPrivileges table associated to a specific Group. Only the administrator can modify the permissions assigned to specific services for each Group. For details on the AccessPrivileges Management attributes, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*.

1. From the main menu, navigate to **Oamp** > **User Manager**.

   This will display the User Manager window (as shown below).

2. In the Group table, click on the **GroupName** (written in blue) of the Group to which you would like to modify the associated AccessPrivileges table.
3. When the UserAccessPrivileges Provisioning window appears, select the new permission you want to give to that Group for each service to which you wish to modify the permission.


UserAccessPrivileges Provisioning

ServiceName	Permission
HlrConfig	ReadWriteExecute
HlrSimProv	ReadWriteExecute
HlrSubsProv	ReadWriteExecute
HssConfig	ReadWriteExecute
HssSubsProv	ReadWriteExecute
Oamp	Read
SipConfig	ReadWriteExecute
SipSubsProv	ReadWriteExecute
Ss7Config	ReadWriteExecute
SubscriberProv	ReadWriteExecute
System	Read
Database	Unprovision
ExternalService	Unprovision
Policy	Unprovision
Schema	Unprovision
SystemValidation	Unprovision

Figure 5: UserAccessPrivileges provisioning window

4. Click **Commit** when all the changes have been entered.
5. When the confirmation message

```
UserAccessPrivileges entry was successfully committed
```

appears, Click **OK**.

6. Click **Display All Groups** to return to the User Manager window.

Display all groups from the AccessPrivileges table in the WebCI

This procedure describes the steps to go back to the Groups/Users main window from the AccessPrivileges table and display all Groups in the Group table. For details on the Group Management attributes, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*.

From the User Manager window displaying the AccessPrivileges table, click on the **Display All Groups** button beside the GroupName.

(see above).

Provisioning services in the Service table

The Service table adds, displays, modifies, and deletes external services.

The Subscriber Data Management system pre-defines internal services and their associated entities; see Table *Predefined services and associated entities* in the *SDM System Configuration Reference Manual*. Any user can display these services within the Service table.

WARNING

Warning: Pre-defined services cannot be deleted by any user (including the system administrator) because deleting these internal services could impact the system.

The system administrator can define other services for external entities by adding them manually to the Global Schema. To assign external entities to a newly defined service, the system administrator must define the association when creating a new entity in the Global Schema (contact the Tekelec *My Oracle Support (MOS)* for assistance).

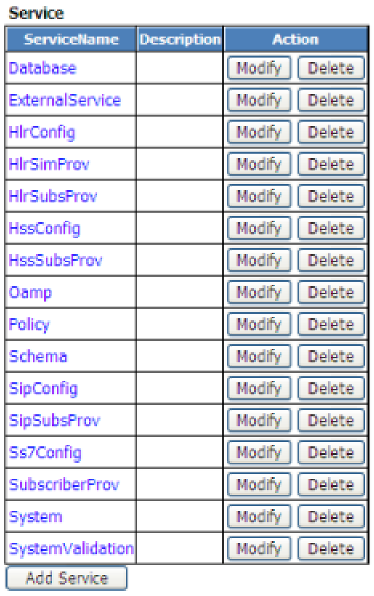
The system administrator can then

- modify the description given to these services
- delete the newly added services

Display services from the WebCI

This procedure describes how to view the Service table with all of the services defined in the system. For details on the Services attributes, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*.

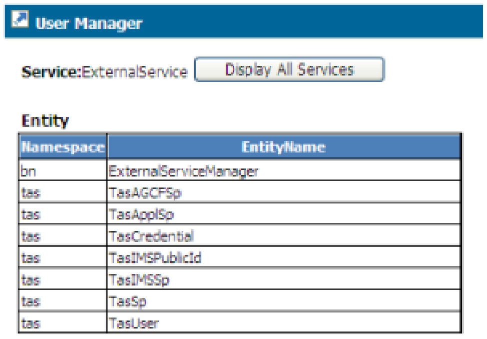
1. From the main menu, navigate to **Oamp** > **User Manager**
   The User Manager window displays including the Services table. The Service table displays all the services defined in the system.



Service

ServiceName	Description	Action
Database		Modify Delete
ExternalService		Modify Delete
HlrConfig		Modify Delete
HlrSimProv		Modify Delete
HlrSubsProv		Modify Delete
HssConfig		Modify Delete
HssSubsProv		Modify Delete
Oamp		Modify Delete
Policy		Modify Delete
Schema		Modify Delete
SipConfig		Modify Delete
SipSubsProv		Modify Delete
Ss7Config		Modify Delete
SubscriberProv		Modify Delete
System		Modify Delete
SystemValidation		Modify Delete

2. Click the hyperlink of the service name to display the entities for this service.



Service:ExternalService

Entity

Namespace	EntityName
bn	ExternalServiceManager
tas	TasAGCFSp
tas	TasApplSp
tas	TasCredential
tas	TasIMSPublicId
tas	TasIMSSp
tas	TasSp
tas	TasUser

Creating a service from the WebCI

The administrator of the system can define services (service name and description) for external entities (entities manually added in the Global Schema by the system administrator) through the WebCI's Service table (e.g., ServiceName: 'ExternalService', which regroups the ExternalServiceManager entity).

In order to assign external entities to a newly defined service, the system administrator must define the association when creating new entities in the Global Schema (contact the Customer Care Center for assistance).

The following WebCI procedure shows how to create a new service. For details on the Service entity's parameters, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*.

1. From the main menu, navigate to **Oamp** > **User Manager**. This will display the User Manager window.
2. Click the **Add Service** button below the Service table.

3. When the Service Provisioning window appears (see figure below), enter the information to be added (the '*' identifies a mandatory attribute).

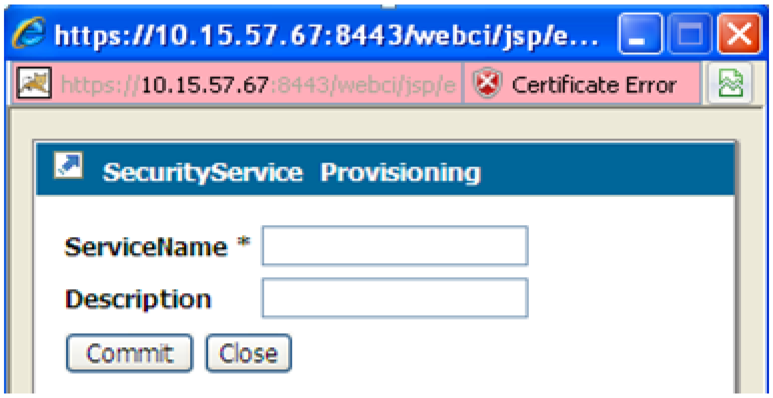

https://10.15.57.67:8443/webci/jsp/e...
https://10.15.57.67:8443/webci/jsp/e
Certificate Error
SecurityService Provisioning
ServiceName *
Description
Commit
Close

Figure 6: Service Security Provisioning Window

4. Click **Commit** when all the information has been entered.
5. When the confirmation message "`Entity entry was successfully committed`" appears, click **OK**.

Modifying a service from the WebCI

The administrator of the system can modify the description given to each service.

The following WebCI procedure shows how to modify a service entry from the Service[ ] entity. For details on the Service parameters, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*.

To perform the task in the following table, refer to the procedure for the step-by-step instructions.

1. From the main menu, navigate to **Oamp** > **User Manager**. This will display the User Manager window.
2. Click the **Modify** button beside the Service entry of the service you wish to modify.
3. When the Service Provisioning window appears, enter the new description you wish to give to the service.
4. Click **Commit** when all the information has been entered.
5. When the confirmation message "`Entity entry was successfully committed`" appears, click **OK**.

Deleting a service from the WebCI

The pre-defined services cannot be deleted by any user (including the administrator of the system) since these are internal services and a deletion could impact the system. However, the administrator of the system can delete services that he has added himself that regroup external entities (not pre-defined ones).

The following WebCI procedure shows how to delete a service from the Service[ ] entity. For details on the Service parameters, refer to the "User Management through CLI" section of the *SDM System Configuration - Reference Manual*. To perform the task in the following table, refer to the procedure for the step by step instructions.

1. From the main menu, navigate to **Oamp** > **User manager**. This will display the User Manager window.
2. Click the **Delete** button beside the Service entry of the service you wish to delete.
3. The following message is returned: "`You are about to delete this entry, Continue?`"
4. Click **OK** if you wish to continue or **Cancel**, otherwise.

Creating and Managing Users for Notifications

The Notification Management functionality allows the management of users with notification subscription permissions for specific applications, entities and parameters.

The CLI and WebCI support multiple types of user accounts, which can only be managed by the administrator. For an external application to be able to subscribe to notifications for specific applications/entities/attributes, it must have a user account defined by the SDM system administrator with a username, password and application name. The SDM system administrator must associate applications to user accounts, by provisioning the UserApplicationMap[ ] entity. The users must first be defined in the User[ ] entity and the applications must also be defined with notification subscription permissions for each entity/attribute as well as with notification properties. This can be provisioned in the ApplicationIdentity[ ], NotificationSubscribe[ ] and ApplicationProperty[ ] entities. Take note that one user can have multiple applications associated to it, which allows the user to subscribe notifications for various applications.

For more details on the system's behavior, refer to the "Security Management" section of the *SDM Product Description*. For more details on the entities to provision for Notification Management, refer to the "Notification Security Management" section of the *SDM System Configuration - Reference Manual*.

Take note that changes made to the system configuration or subscriber provisioning data take effect immediately. There is no rollback mechanism.

Notification Management Using CLI

Notification management requires the provisioning of users, their applications, and notification subscriptions/properties. Use these procedures:

- Provision applications
- Provision notification subscription permissions
- Provision notification properties
- Provision users

Requirements: All CLI provisioning procedures require the operator to log into a CLI session with a valid username and password.

Define/Provision applications from the CLI

The following CLI procedure shows how the system's administrator can provision applications in the ApplicationIdentity[ ] entity.

1. Go to the Oamp subsystem by typing,

```
:> Oamp[]
```

2. Go to Notification Management by typing,

```
Oamp[]> NotificationManager[]
```

3. Provision the ApplicationIdentity[ ] entity by performing one of the following commands, as needed:

   - Display the applications provisioned in the ApplicationIdentity[ ] entity by performing the following:

```
Oamp[]> NotificationManager[]> display ApplicationIdentity[]
```

   - Add an application by performing the following:

```
Oamp[]:NotificationManager[]> add ApplicationIdentity[ApplName=interface2]
```

   The following message will be displayed:

```
Added: 1
```

   - Modify an application by performing the following and specifying the new value:

```
Oamp[]:NotificationManager[]> ApplicationIdentity[ApplName=interface2]> modify
 . Description=RESTful interface
```

   The following message will be returned:

```
Warning, you are about to modify this instance(s) permanently, Proceed with
modify? (y/[n]):
```

   Type **y** if you wish to continue or **n** to cancel.

   - Delete an application by performing the following:

```
Oamp[]:NotificationManager[]> delete ApplicationIdentity[ApplName=interface2
```

   The following message will be returned:

```
Warning, you are about to delete this instance(s) permanently, Proceed with
delete? (y/[n]):
```

   Type **y** if you wish to continue or **n** to cancel.

Provision notification subscription permissions from the CLI

1. Go to the Oamp subsystem by typing **`:> Oamp[]`**

2. Go to User Management by typing **NotificationManager[]**

```
Oamp[]> NotificationManager[]
```

3. Provision the NotificationSubscribe[ ] entity by performing one of the following commands:
   - Display entities and attributes to which notifications can be subscribed for a specific application by typing **Oamp[]> NotificationManager[]> ApplicationIdentity[ApplName=BlueCli] >display NotificationSubscribe[]**
   - Add an entity or attribute to which notifications can be subscribed for a specific application by typing **Oamp[]> NotificationManager[]> ApplicationIdentity[ApplName=BlueCli] > add NotificationSubscribe[Namespace=bn; Entity=MSISDN; Attribute=DefaultBsg]**

     The following message displays:

```
Added: 1
```

   - Delete notification properties for an application by typing **Oamp[]> NotificationManager[]> ApplicationIdentity[ApplName=BlueCli] > delete NotificationSubscribe[Namespace=bn; Entity=MSISDN; Attribute=DefaultBsg]**

     The following message displays:

```
Warning, you are about to delete this instance(s) permanently, Proceed with
delete? (y/[n]):
```

     Type **y** to continue or **n** to cancel.

Provisioning notification properties from the CLI

Notification properties define per application whether the previous value (before update) must be included in the notifications in addition to the current value (after update). The administrator defines these properties in the ApplicationProperty[ ] entity.

1. Go to the Oamp subsystem by typing,

```
:> Oamp[]
```

2. Go to Notification Management by typing

```
Oamp[]> NotificationManager[]
```

3. Provision the ApplicationProperty[ ] entity by performing one of the following commands, as needed:
   - Display an application's notification properties, by performing the following:

```
Oamp[]> NotificationManager[]> ApplicationIdentity[ApplName=BlueCli] > display
 ApplicationProperty[]
```

- Add notification properties to an application, by performing the following:

```
Oamp[]> NotificationManager[]> ApplicationIdentity[ApplName=BlueCli] > add
ApplicationProperty[Namespace=bn; Entity=MSISDN; isValueBefore=1]
```

The following message will be displayed:

```
Added: 1
```

- Delete notification properties for an application, by performing the following:

```
Oamp[]> NotificationManager[]> ApplicationIdentity[ApplName=BlueCli] > delete
ApplicationProperty[Namespace=bn; Entity=MSISDN; isValueBefore=1]
```

The following message will be returned:

```
Warning, you are about to delete this instance(s) permanently, Proceed with
delete? (y/[n]):
```

4. Type **y** if you wish to continue or **n** to cancel.

Provisioning user/application combinations

By default, the system pre-defines users in the User table: user, operation, surveil, admin, batch, and simprov. Each of these pre-defined users is associated with an application in the UserApplicationMap table.

Table 9: UserApplicationMap table

User	Applications associated to each user at installation (pre-defined)
User	BlueCli, WebCI, SOAP, CmdFileLoader, LdapDataServer
Operation	BlueCli, WebCI, LdapDataServer
Surveil (surveillance)	BlueCli, WebCI, LdapDataServer
Admin	BlueCli, WebCI, SOAP, CmdFileLoader, SNMP, LdapDataServer
Batch	BlueCli, WebCI, SOAP, CmdFileLoader, LdapDataServer
Simprov	BlueCli, WebCI, SOAP, CmdFileLoader, LdapDataServer

The administrator can display, add, or delete user/application combinations and modify their login options.

For an external application to subscribe to notifications for specific applications, entities, or attributes, the application must have a user account defined by the SDM system administrator. The user account defined in the User[ ] entity contains the username, password, and application name (from the user/application association defined in the UserApplicationMap[ ] entity.

Note: The application must have been pre-defined in the ApplicationIdentity[ ] entity.

Depending on the user interface used, the following procedures define users and associate them to applications by provisioning the UserApplicationMap table.

Users can only create and display the User table, modify user passwords, and assign a group to each user.

WebCI	**CLI**
Creating user	Provisioning users
Modifying user	
Deleting user	

Provisioning user/application combinations from the CLI

This CLI procedure shows how to create user/application combinations.

The user must have been defined in the User[ ] entity, and the applications must have been defined in the ApplicationIdentity[ ] entity.

1. Go to the Oamp subsystem by typing **`:> Oamp[]`**
2. Go to User Management by typing

```
Oamp[]> NotificationManager[]
```

3. Provision the UserApplicationMap[ ] entity by performing one of the following commands:
   - Display user accounts by typing:

```
Oamp[]> NotificationManager[]> display UserApplicationMap[]
```

   - Add new users by typing:

```
Oamp[]> NotificationManager[]> add
UserApplicationMap[UserName=user2;ApplName=BlueCli]
```

   The following message displays:

```
Added: 1
```

   - Delete notification properties for an application by typing:

```
Oamp[]> NotificationManager[]> delete UserApplicationMap[UserNames=user2;
ApplName=WebCI]
```

   The following message displays:

```
Warning, you are about to delete this instance(s) permanently, Proceed with
delete? (y/[n]):
```

   Type **y** to continue or **n** to cancel.

- Modify the logging option of user accounts by typing:

```
Oamp[]>NotificationManager[]>UserApplicationMap[UserName=user2;ApplName=BlueCli]>
 modify . LogOption=1
```

The following message displays:

```
Warning, you are about to modify this instance(s) permanently, Proceed with
modify? (y/[n]):
```

Type **y** to continue or **n** to cancel.

If you typed **y**, the system returns

```
Modified: 1
```

Notification Management Using WebCI

This section outlines the WebCI procedures to provision the users, their applications and notification subscriptions/properties. To perform the tasks in the following table, refer to the procedures for the step by step instructions.

Task
View all User Management information
Provision applications
Provision notification subscription permissions
Provision notification properties
Provision users

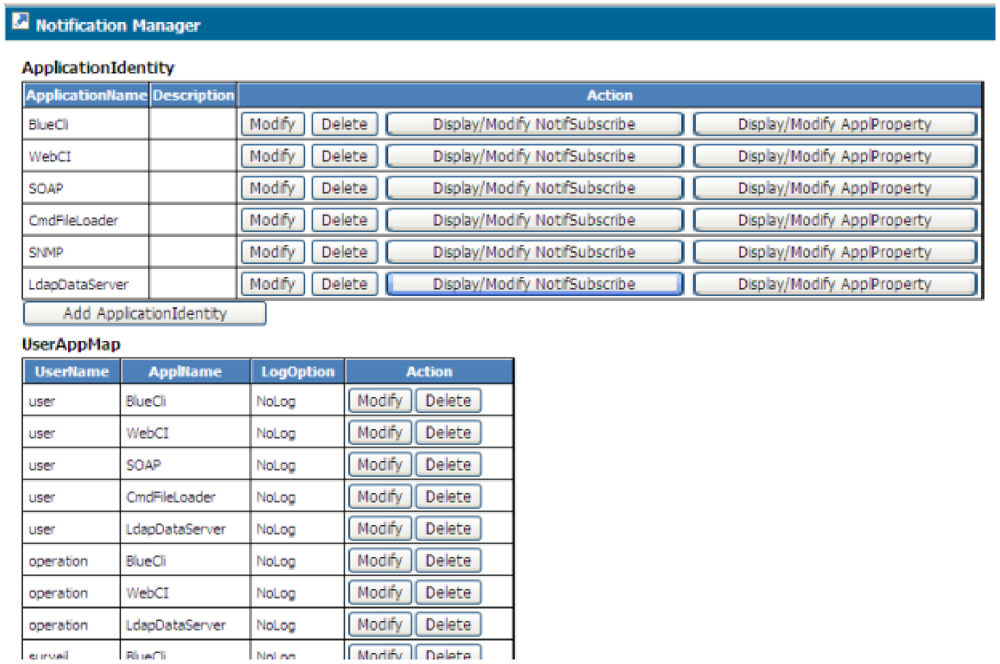
Requirements: All WebCI provisioning procedures require the operator to log into a WebCI session with a valid username and password.

View all notification management information from the WebCI

This procedure displays the Notification Management information.

From the main menu, navigate to **Oamp** > **Notification Manager**.

The Notification Manager window opens.



Notification Manager

ApplicationIdentity

ApplicationName	Description	Action			
BlueCli		Modify	Delete	Display/Modify NotifSubscribe	Display/Modify ApplProperty
WebCI		Modify	Delete	Display/Modify NotifSubscribe	Display/Modify ApplProperty
SOAP		Modify	Delete	Display/Modify NotifSubscribe	Display/Modify ApplProperty
CmdFileLoader		Modify	Delete	Display/Modify NotifSubscribe	Display/Modify ApplProperty
SNMP		Modify	Delete	Display/Modify NotifSubscribe	Display/Modify ApplProperty
LdapDataServer		Modify	Delete	Display/Modify NotifSubscribe	Display/Modify ApplProperty

Add ApplicationIdentity

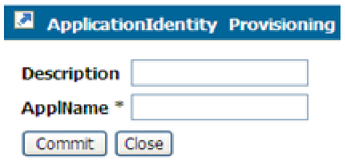
UserAppMap

UserName	ApplName	LogOption	Action	
user	BlueCli	NoLog	Modify	Delete
user	WebCI	NoLog	Modify	Delete
user	SOAP	NoLog	Modify	Delete
user	CmdFileLoader	NoLog	Modify	Delete
user	LdapDataServer	NoLog	Modify	Delete
operation	BlueCli	NoLog	Modify	Delete
operation	WebCI	NoLog	Modify	Delete
operation	LdapDataServer	NoLog	Modify	Delete
surveil	BlueCli	NoLog	Modify	Delete

Define/Provision Applications from the WebCI

The following WebCI procedure shows how the system's administrator can provision applications in the ApplicationIdentity table.

1. From the main menu, navigate to **Oamp** > **NotificationManager**. This will display the Notification Manager window.
2. Click the **Add ApplicationIdentity** button below the ApplicationIdentity table.
3. When the ApplicationIdentity Provisioning window appears (see figure below), enter the information (ApplName) to be added.



ApplicationIdentity Provisioning

Description	
ApplName *	

Commit Close

Figure 7: ApplicationIdentity Provisioning Window to Create Applications

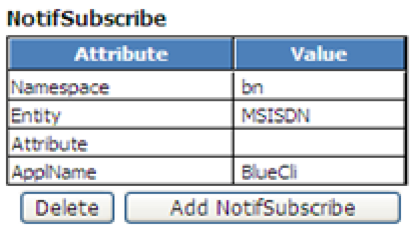
4. Click **Commit** when all the information has been entered.
5. When the confirmation message “`Entity entry was successfully committed`” appears, click **OK**.

--- end ---

Provisioning notification subscription permissions from the WebCI

Notification subscription permissions define per application the entities and attributes to which the user (external application) can subscribe notifications. The administrator defines these permissions in the NotificationSubscribe[ ] entity. This procedure describes how to provision the NotifSubscribe[ ] entity using the add or delete operation.

1. From the main menu, navigate to **Oamp** > **NotificationManager**.
   The Notification Manager window displays.
2. Click the **Display/Modify NotifSubscribe** button of an application in the ApplicationIdentity table.
   The NotifSubscribe window with the NotifSubscribe entity opens.



NotifSubscribe

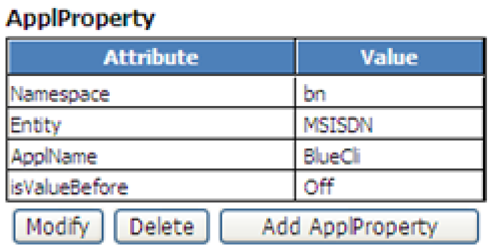
Attribute	Value
Namespace	bn
Entity	MSISDN
Attribute	
ApplName	BlueCli

3. Perform one of the following operations:
   - Click **Add NotifSubscribe** to add notification subscription permissions (Namespace, Entity, Attribute) to the existing application.
   - Click **Delete** to delete an entry one at a time.
4. Click **Commit** in the pop-up window when all the information has been entered.
   The system returns a confirmation message similar to `Entity entry was successfully committed`.
5. Click **OK**.

Provisioning notification properties from the WebCI

Notification properties define per application whether the previous value (before update) must be included in the notifications in addition to the current value (after update). The administrator defines these properties in the ApplicationProperty[ ] entity. This procedure describes how to provision the ApplProperty[ ] entity using the add, modify, or delete operation.

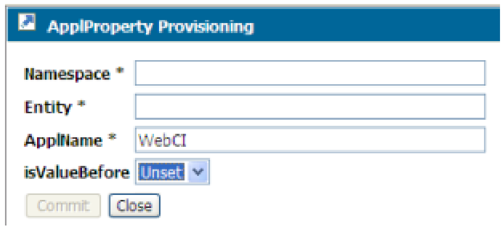
1. From the main menu, navigate to **Oamp** > **NotificationManager**.
   The Notification Manager window displays.
2. Click the **Display/Modify ApplProperty** button of an application in the ApplicationIdentity table.
   The ApplProperty window with the ApplProperty entity opens.



ApplProperty

Attribute	Value
Namespace	bn
Entity	MSISDN
ApplName	BlueCli
isValueBefore	Off

3. Perform one of the following operations:
   - Click **Add ApplProperty** to add new properties to the existing application in the ApplProperty Provisioning pop-up window.



Namespace *	
Entity *	
ApplName *	WebCI
isValueBefore	Unset

- Click **Modify** to modify properties.
- Click **Delete** to delete an entry one by one.

4. Click **Commit** in the pop-up window when all the information has been entered. The system returns a confirmation message similar to `Entity entry was successfully committed`.
5. Click **OK**.

Provisioning Users

By default, the system pre-defines these users in the User table: user, operation, surveil, admin, batch and simprov. Each of these pre-defined users are associated with an application in the UserApplicationMap table.

Table 10: UserApplicationMap table

User	Applications associated to each user at installation (pre-defined)
User	BlueCli, WebCI, SOAP, CmdFileLoader, LdapDataServer
Operation	BlueCli, WebCI, LdapDataServer
Surveil (surveillance)	BlueCli, WebCI, LdapDataServer
Admin	BlueCli, WebCI, SOAP, CmdFileLoader, SNMP, LdapDataServer
Batch	BlueCli, WebCI, SOAP, CmdFileLoader, LdapDataServer
Simprov	BlueCli, WebCI, SOAP, CmdFileLoader, LdapDataServer

The administrator can display, add, or delete user-application combinations and modify their logging options. For an external application to be able to subscribe to notifications for specific applications, entities, or attributes, it must have a user account defined by the SDM system administrator with a username, password (in the User[ ] entity) and application name (user-application association defined in the UserApplicationMap[ ] entity). For the defined user accounts (in User[ ] entity), the SDM system administrator must associate them with applications by provisioning the UserApplicationMap[ ] entity.

Note: The application must already be defined in the ApplicationIdentity[ ] entity (see previous sub-sections).

These procedures define users and associate applications to them by provisioning the UserApplicationMap table. Users can create and display the User table, modify user passwords, and assign a group to each user.

For details on the User table, refer to the "Notification Security Management through CLI" section of the *SDM System Configuration - Reference Manual*.

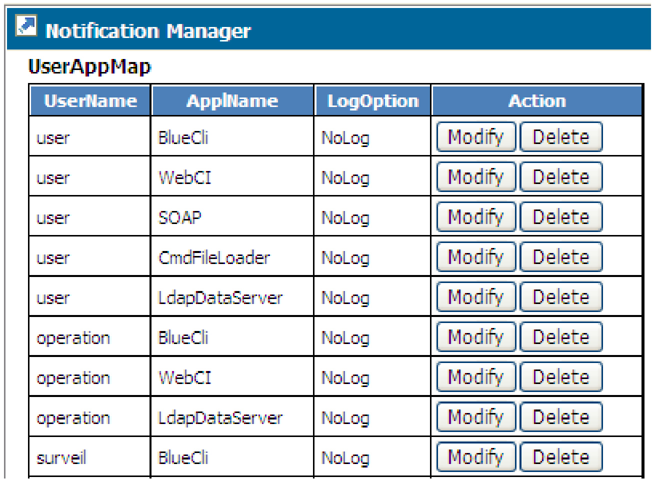
WebCI	**CLI**
Creating user	Provisioning users
Modifying user	
Deleting user	

Provisioning user/application combinations from the WebCI

This procedure creates user/application combinations in the UserAppMap entity.

The user must have been defined in the User[ ] entity, and the applications must have been defined in the ApplicationIdentity[ ] entity.

1. From the main menu, navigate to **Oamp** > **NotificationManager**.
   The Notification Manager window displays with the UserAppMap table.



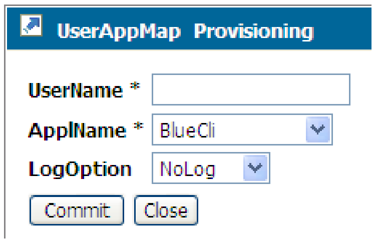
Notification Manager

UserAppMap

UserName	ApplName	LogOption	Action
user	BlueCli	NoLog	Modify Delete
user	WebCI	NoLog	Modify Delete
user	SOAP	NoLog	Modify Delete
user	CmdFileLoader	NoLog	Modify Delete
user	LdapDataServer	NoLog	Modify Delete
operation	BlueCli	NoLog	Modify Delete
operation	WebCI	NoLog	Modify Delete
operation	LdapDataServer	NoLog	Modify Delete
surveil	BlueCli	NoLog	Modify Delete

2. Perform one of the following operations:
   - Click **Add UserAppMap** to add a new user/application combination.

     The UserAppMap Provisioning pop-up window opens.


UserAppMap Provisioning

UserName *

ApplName * BlueCli

LogOption NoLog

- Click Modify to change the Log Option.
- Click **Delete** to delete the UserAppMap entry.

3. Click **Commit** in the pop-up window when all information has been entered.
   The system returns a confirmation message similar to `Entity entry was successfully committed`.
4. Click **OK**.

Chapter 4

Provisioning subscribers for Mobile Number Portability (WebCI)

Topics:

- *Provisioning MNP DB Mismatch Error Handling feature.....63*
- *Add MNP subscriber.....63*
- *Display MNP MSISDN profile.....64*

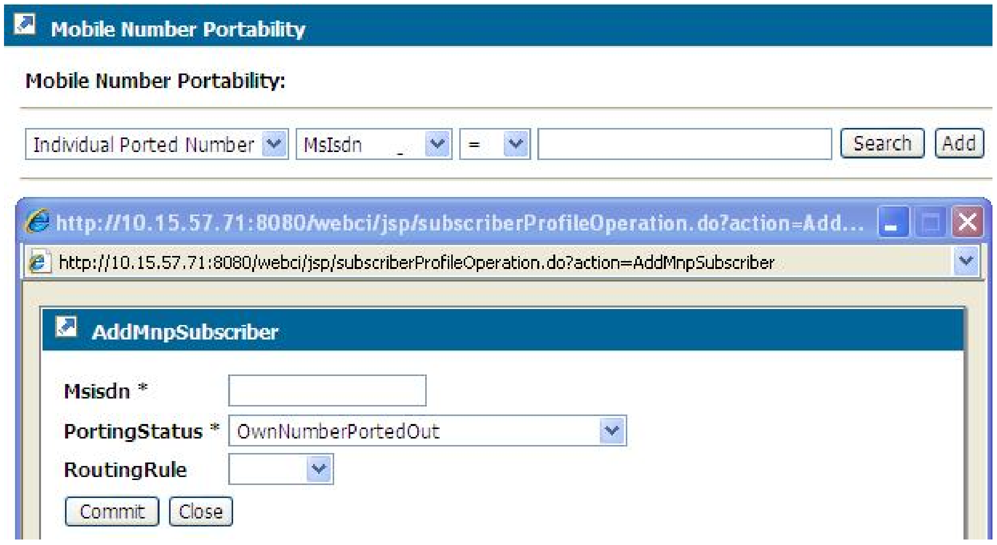
This chapter contains procedures for provisioning subscribers for Mobile Number Portability.

Provisioning MNP DB Mismatch Error Handling feature

Add MNP subscriber

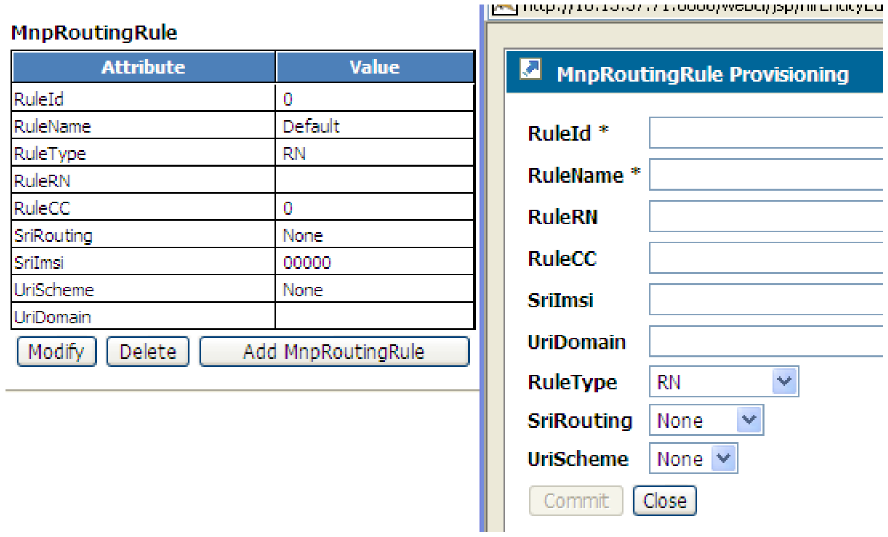
This procedure adds a subscriber to the number portability database and updates the MSISDN porting status by provisioning the MnpPortedOut table.

1. Go to **HLR> Mobile Number Portability**.
   The Mobile Number Portability window displays.
2. Below the Mobile Number Portability status field, click the drop-down arrow of the first field and select *Individual Ported Number*.


Mobile Number Portability
Mobile Number Portability:
Individual Ported Number
MsIsdn
=
Search
Add
http://10.15.57.71:8080/webci/jsp/subscriberProfileOperation.do?action=Add...
http://10.15.57.71:8080/webci/jsp/subscriberProfileOperation.do?action=AddMnpSubscriber
AddMnpSubscriber
Msisdn *
PortingStatus *
OwnNumberPortedOut
RoutingRule
Commit
Close

3. Click the drop-down arrow of the second field and select *Msisdn*.
4. Click **Add**.
5. Type the MSISDN number, select the routing rule to be used, and update the subscriber porting status if applicable; then click **Commit** to add the subscriber to the database.

   Note: Changing the porting status in this table will update the porting status in the MSISDN table. Then click **Commit** to add the subscriber to the database.
6. Define the rules used to build routing numbers for outgoing messages in the MnpRoutingRule table; perform one of these operations:
   - Click **Add MnpRoutingRule** to define the rule properties and add the rule to the database.



MnpRoutingRule

Attribute	Value
RuleId	0
RuleName	Default
RuleType	RN
RuleRN	
RuleCC	0
SriRouting	None
SriImsi	00000
UriScheme	None
UriDomain	

MnpRoutingRule Provisioning

RuleId *	
RuleName *	
RuleRN	
RuleCC	
SriImsi	
UriDomain	
RuleType	RN
SriRouting	None
UriScheme	None

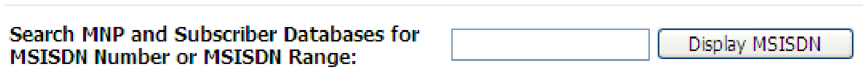
- Click **Modify** to change attributes for the routing rule.
- Click **Delete** to delete the routing rule entry.

7. Provision the list of ported-out numbers associated with a subscription ID. These numbers may or may not be a part of the "Own Number Range".

Display MNP MSISDN profile

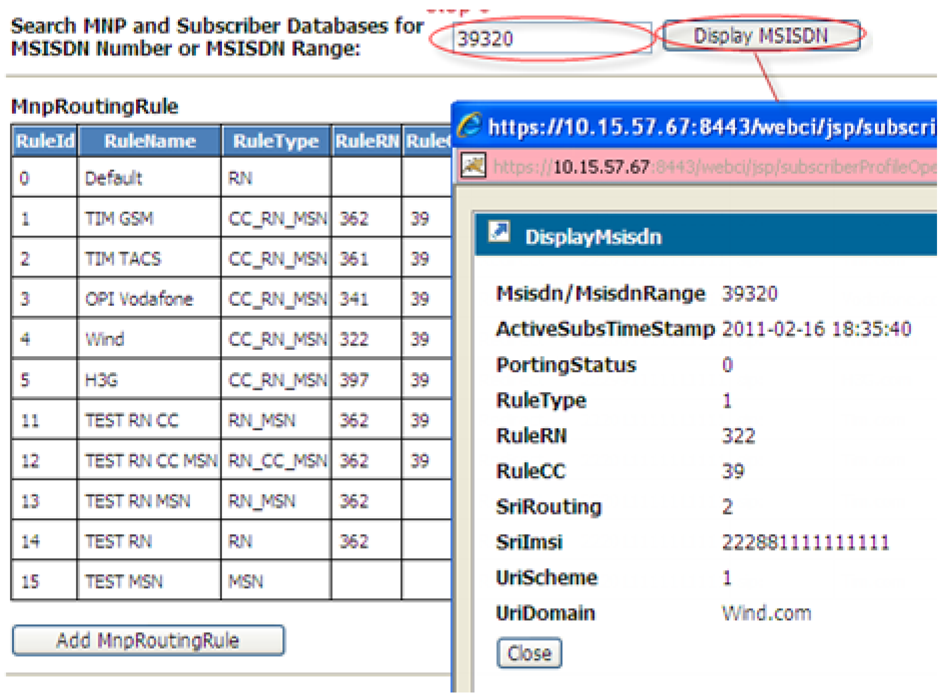
This procedure displays the profile associated with a given MSISDN and whether it has been ported out. The search accesses all Number Portability databases and subscriber tables.

1. Go to **HLR> Mobile Number Portability**.
   The Mobile Number Portability window displays.
2. Enter the subscriber MSISDN number or range.



Search MNP and Subscriber Databases for MSISDN Number or MSISDN Range:

3. Click **Display MSISDN**.


Search MNP and Subscriber Databases for MSISDN Number or MSISDN Range:
39320
Display MSISDN
https://10.15.57.67:8443/webci/jsp/subscri
https://10.15.57.67:8443/webci/jsp/subscriberProfileOpe

MnpRoutingRule

RuleId	RuleName	RuleType	RuleRN	Rule
0	Default	RN		
1	TIM GSM	CC_RN_MSN	362	39
2	TIM TACS	CC_RN_MSN	361	39
3	OPI Vodafone	CC_RN_MSN	341	39
4	Wind	CC_RN_MSN	322	39
5	H3G	CC_RN_MSN	397	39
11	TEST RN CC	RN_MSN	362	39
12	TEST RN CC MSN	RN_CC_MSN	362	39
13	TEST RN MSN	RN_MSN	362	
14	TEST RN	RN	362	
15	TEST MSN	MSN		

DisplayMsisdn

Msisdn/MsisdnRange	39320
ActiveSubsTimeStamp	2011-02-16 18:35:40
PortingStatus	0
RuleType	1
RuleRN	322
RuleCC	39
SriRouting	2
SriImsi	222881111111111
UriScheme	1
UriDomain	Wind.com

Chapter 5

Examples of XML Templates for Subscriber Provisioning

Topics:

- *Introduction.....67*
- *Subscriber provisioning sequence.....67*
- *Subscription provisioning template.....68*
- *SIM card provisioning template.....69*
- *HLR subscriber profile provisioning template....70*
- *SIP Subscriber Provisioning.....71*
- *HSS subscriber provisioning file.....72*
- *SLF subscriber provisioning file.....80*
- *DNS Enum subscriber provisioning file.....80*
- *AAA subscriber provisioning file.....81*
- *LTE-HSS subscriber provisioning templates.....83*
- *Update request template.....85*
- *Delete XML request template.....86*
- *Select XML request file.....86*
- *Request-Reply XML file.....87*

Introduction

This chapter presents the following examples of XML or XML-REST files:

- XML files with insert requests for the provisioning of subscribers with templates:
- Subscription provisioning
- SIM card Provisioning
- HLR Subscriber Provisioning
- SIP Subscriber Provisioning
- HSS Subscriber Provisioning
- SLF Subscriber Provisioning
- DNS ENUM Subscriber Provisioning
- AAA Subscriber Provisioning
- SPR Subscriber Provisioning (XML, XML-REST)
- XML file with Update request

  Note: It is easier to simply delete and insert a new provisioning entry than to modify it with an update request.

- XML file with Delete requests
- XML Request followed by the XML Reply

  For more explanations on the format of the different types of requests, for a description on the templates and for detailed information of each parameter, refer to the SDM Subscriber Provisioning - Reference Manual.

  For a description of the error notifications the system may generate in the XML reply, refer to the Error Notifications section of the *SDM Monitoring, Maintaining, Troubleshooting – Reference Manual*.

Subscriber provisioning sequence

This section presents the general provisioning order that must be followed to define the subscriber in the SDM by creating an entry in the Subscription entity and giving it a unique SubscriptionID value to identify the subscriber.

1. **HLR subscriber:**
   a. Provision SIM cards and assign a SubscriptionID to each, by provisioning the Sim and SimImsiMap entities or performing the AddSIM() operation.
   b. Define MSISDN(s) for the subscriber by provisioning the MSISDN entity.
   c. Provision the HLR Service Profile (Subscriber Profile, Call Forward, Call Barring Services, Supplementary Services, etc…) and create a HlrServiceProfileID that will identify this service profile. In the current release, only one HLR service profile per subscriber (SubscriptionID) is supported.
   d. Once all of the above is defined, you must create MSISDN-IMSI couples (Primary and if needed Alternate) by associating the subscriber's MSISDNs with its IMSIs and assign them to an HLR service profile (HlrServiceProfileID) through the MsIsdnImsiProfileAssociation entity.
2. **SIP subscriber profile:**

Provision the subscriber with Address Of Records by provisioning the AddressOfRecord entity. Multiple AoRs can be provisioned for one subscriber (SubscriptionID).

3. **HSS subscriber profile:**

   a. Provision HSS Subscriptions by defining ChargingIDs in the HssSubscription entity. Multiple HSS Subscriptions can be provisioned for one subscriber (SubscriptionID).
   b. Define Private Identities (IMPI) for an HSS Subscription by provisioning the PrivateIdentity entity.
   c. Define Public Identities (IMPU) for an HSS Subscription by provisioning the PublicIdentity entity.
   d. Link Public Identities to Private Identities for an HSS Subscription by performing the LinkPublicIdentities() operation.
   e. Define Service Profiles for the HSS Subscription by provisioning the ServiceProfile entity. Define each Service Profile with:

      a. Initial Filter Criterias
      b. Service Point Triggers
      c. Link HSS Shared Initial Filter Criterias if needed.

4. **HSS host of HSS subscriber:** Provision the HssSlfPublic2HssName entity.
5. **ENUM users:**

   Provision DNS ENUM users in the DNSEnumUser entity. Multiple ENUM users can be provisioned for one subscriber (SubscriptionID).

6. **AAA subscriber:**

   a. Provision AAAUserIds in the AAAUserId entity. Multiple AAA users can be provisioned for one subscriber (SubscriptionID).
   b. Define AAA User Vendor Attributes by provisioning the AAAUserVendorAttributes entity.
   c. Define authorized IP address pools for a user configured as a “special user” by provisioning the AAAUserIPAddressPools entity.

Refer to the *SDM Subscriber Provisioning - Reference Manual* for details on each entity that is used to provision subscriber profiles.

Subscription provisioning template

This template provisions the subscription ID of a subscriber. The SubscriptionID is a unique identity that represents the subscriber and is common for all the profiles it has provisioned in the SDM (HLR profile, HSS profile, SLF, AAA, DNS Enum).

The template file is followed by an Invoking file, which allows the creation of the SubscriptionID.

Template file

```
<file>
<!-- Add basic subscription. -->
<!-- Template requests for adding basic subscription. -->
<req name="insert" id="11100">
```

```
    <ent name="Subscription" ns="global" />
    <set>
        <expr><attr name="SubscriptionID"/><value val=""/></expr>
    </set>
</req>
<!—Command Template for adding/modifying basic subscriber. -->
<template id="111" otherAttributesModifiable="N">
    <attr name="SubscriptionID" mandatory="Y"/>
    <tr id="11100"/>
    </template>
</file>
```

Invoke file

```
<file>
<!-- Request templates to add basic Subscription. -->
    <tp id="111">
        <tpi nm="SubscriptionID" val="7"/>
    </tp>
</file>
```

SIM card provisioning template

This template and the invoke file provision the SIM card in the SDM database.

Template file

```
<file>
<!-- Create SIM for a subscriber with existing Subscription. -->
<!-- Template request for creating SIM. -->
<req name="insert" id="10001">
    <ent name="Sim" ns="bn" />
    <set>
        <expr><attr name="SimId"/><value val=""/> </expr>
            <expr><attr name="SubscriptionID"/><value val=""/></expr>
            <expr><attr name="AlgorithmName"/><value val="XOR"/></expr>
            <expr><attr name="Ki32HexChar"/><value val=""/></expr>
            <expr><attr name="PUK"/><value val=""/></expr>
            <expr><attr name="ManufacturerId"/><value val=""/></expr>
            <expr><attr name="SimType"/><value val="SIM"/></expr>
    </set>
</req>
<req name="insert" id="10002">
    <ent name="SimImsiMap" ns="bn" />
    <set>
        <expr><attr name="SimId"/><value val=""/> </expr>
            <expr><attr name="Imsi"/><value val=""/></expr>
            <expr><attr name="PrimaryImsi"/><value val="1"/></expr>
    </set>
</req>
<!—Command Template for creating SIM. -->
<template id="1000" otherAttributesModifiable="N">
    <attr name="SimId" mandatory="Y"/>
    <attr name="SubscriptionID" mandatory="Y"/>
    <attr name="Imsi" mandatory="Y"/>
    <attr name="Ki32HexChar" mandatory="Y"/>
    <attr name="PUK" mandatory="Y"/>
    <tr id="10001"/>
    <tr id="10002"/>
</template>
</file>
```

Invoke File

```
<file>
        <!-- Request template to create SIM for a subscriber. -->
        <tp id="1000">
            <tpi nm="SimId" val="234445666006"/>
            <tpi nm="Imsi" val="310910421000106"/>
            <tpi nm="SubscriptionID" val="7"/>
            <tpi nm="Ki32HexChar" val="A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5A5"/>
            <tpi nm="PUK" val="2345522333"/>
        </tp>
</file>
```

HLR subscriber profile provisioning template

This template and the invoke file provision a basic HLR subscriber profile.

Refer to the previous chapter for instructions on the sequence that needs to be followed when provisioning HLR Subscriber Profiles.

Requirement: This Template file doesn't include the creation of the SubscriptionID for the subscriber, therefore prior to executing it, the subscriber's SubscriptionID must be created.

Template file

```
<file>
<!-- Add basic subscriber. -->
<!-- Subscription must exists -->
<!-- Template requests for adding basic subscriber. -->
<req name="insert" id="101">
    <ent name="SubscriberProfile" ns="bn" />
    <set>
            <expr><attr name="SubscriptionID"/><value val=""/></expr>
          <expr><attr name="HlrServiceProfileID"/><value val="1"/></expr>

        <expr><attr name="TeleServiceList"/><value
val="TS11,TS21,TS22"/></expr>
        <expr><attr name="OdbMask"/><value
val="PremiumRateInfo,PremiumRateEntertainment,RoamingOutsideHplmnCountry"/></expr>

        <expr><attr name="MsIsdnAlertInd"/><value val=""/></expr>
        <expr><attr name="PreferredRoutingNetworkDomain"/><value
val="1"/></expr>
        <expr><attr name="OCPlmnTemplateId"/><value val="0"/></expr>
        <expr><attr name="Nam"/><value val="0"/></expr>
    </set>
```

```
</req>
<req name="insert" id="102">
    <ent name="MSISDN" ns="bn" />
    <set>
        <expr><attr name="SubscriptionID"/><value val=""/></expr>
        <expr><attr name="MsIsdn"/><value val=""/></expr>
    </set>
</req>
<req name="insert" id="103">
    <ent name="MsIsdnImsiProfileAssociation" ns="bn" />
    <set>
        <expr><attr name="SubscriptionID"/><value val=""/></expr>
        <expr><attr name="Imsi"/><value val=""/></expr>
        <expr><attr name="MsIsdn"/><value val=""/></expr>
        <expr><attr name="Displayed"/><value val="1"/></expr>
        <expr><attr name="HlrServiceProfileID"/><value val="1"/></expr>
    </set>
</req>
<!--Command Template for adding basic subscriber. -->
<template id="1" otherAttributesModifiable="N">
    <attr name="SubscriptionID" mandatory="Y"/>
    <attr name="Imsi" mandatory="Y"/>
    <attr name="MsIsdn" mandatory="Y"/>
    <attr name="Nam" mandatory="N"/>
    <tr id="101"/>
    <tr id="102"/>
    <tr id="103"/>
    </template>
</file>
```

Invoke file

```
<file>
<!-- Request template to add basic HLR subscriber. -->
    <tp id="1">
            <tpi nm="SubscriptionID" val="7"/>
            <tpi nm="Imsi" val="310910421000106"/>
            <tpi nm="MsIsdn" val="15634210106"/>
            <tpi nm="Nam" val="1"/>
    </tp>
</file>
```

SIP Subscriber Provisioning

This template and the invoke file provision the basic SIP Subscriber profile.

Requirement: This Template file doesn't include the creation of the SubscriptionID for the subscriber, therefore prior to executing it, the subscriber's SubscriptionID must be created.

Template file

```
<file>
<!-- Add SIP profile with GSM authentication. -->
<!-- Template requests for adding SIP profile with GSM authentication.
-->
<req name="insert" odk="yes" id="2002">
    <ent name="AddressOfRecord" ns="bn"/>
    <set>
        <expr><attr name="SubscriptionID"/><value val=""/></expr>
        <expr><attr name="Scheme"/><value val="1"/></expr>
        <expr><attr name="User"/><value val=""/></expr>
        <expr><attr name="Host"/><value val="192.168.10.27"/></expr>
        <expr><attr name="ServiceAllowed"/><value val="1"/></expr>
        <expr><attr name="AuthUserName"/><value val=""/></expr>
        <expr><attr name="AuthPasswd"/><value val=""/></expr>
        <expr><attr name="IsAorAuthenticationEnabled"/><value
val="1"/></expr>
       <expr><attr name="IsSendRegisterAllowed"/><value val="1"/></expr>

        <expr><attr name="IsReceiveRegisterAllowed"/><value
val="1"/></expr>
       <expr><attr name="IsReceiveInviteAllowed"/><value val="1"/></expr>

    </set>
</req>
<!--Command Template for adding SIP profile with GSM authentication. -->
<template id="20" otherAttributesModifiable="N">
    <attr name="SubscriptionID" mandatory="Y"/>
    <attr name="User" mandatory="Y"/>
    <attr name="AuthUserName" mandatory="Y"/>
    <attr name="AuthPasswd" mandatory="Y"/>
    <tr id="2002"/>
</template>
</file>
```

Invoke file

```
<file>
<!-- Request template to add SIP profile with GSM authentication. -->
        <tp id="20">
            <tpi nm="SubscriptionID" val="7"/>
            <tpi nm="User" val="Auth1"/>
            <tpi nm="AuthUserName" val="ep4"/>
            <tpi nm="AuthPasswd" val="ep4ep4"/>
        </tp>
</file>
```

HSS subscriber provisioning file

This XML request file provisions the HSS subscriber profiles in the SDM database.

Note: This XML file includes the creation of the *Subscription provisioning template* for each subscriber.

Prerequisite: The Hss service must be running on the blade and the HSS Server must be enabled (Hss Config) prior to provisioning HSS subscribers.

```
<file>
<!-- Default HSS Subscription profile -->
<!-- to be loaded in the database at start-up  -->
<!-- HSS-Charging-Information  -->
<ent name="HssChargingInfo" ns="bn">
<ChargingID>ChargingID-1</ChargingID>
<PrimEventChargFunction>aaa://host.example1.com;transport=tcp</PrimEventChargFunction>
<PrimChargCollectionFunction>aaa://host.example1.com;transport=tcp</PrimChargCollectionFunction>
<SecEventChargFunction>aaa://host.example2.com;transport=tcp</SecEventChargFunction>
<SecChargCollectionFunction>aaa://host.example2.com;transport=tcp</SecChargCollectionFunction>
</ent>
<ent name="HssChargingInfo" ns="bn">
<ChargingID>ChargingID-2</ChargingID>
<PrimEventChargFunction>aaa://host.example1.com:6666;transport=tcp</PrimEventChargFunction>
<PrimChargCollectionFunction>aaa://host.example1.com:6666;transport=tcp</PrimChargCollectionFunction>
<SecEventChargFunction>aaa://host.example2.com:6666;transport=tcp</SecEventChargFunction>
<SecChargCollectionFunction>aaa://host.example2.com:6666;transport=tcp</SecChargCollectionFunction>
</ent>
<ent name="HssChargingInfo" ns="bn">
<ChargingID>ChargingID-3</ChargingID>
<PrimEventChargFunction>aaa://host.example1.com;protocol=diameter</PrimEventChargFunction>
<PrimChargCollectionFunction>aaa://host.example1.com;protocol=diameter</PrimChargCollectionFunction>
<SecEventChargFunction>aaa://host.example2.com;protocol=diameter</SecEventChargFunction>
<SecChargCollectionFunction>aaa://host.example2.com;protocol=diameter</SecChargCollectionFunction>
</ent>
<ent name="HssChargingInfo" ns="bn">
<ChargingID>ChargingID-4</ChargingID>
<PrimEventChargFunction>aaa://host.example1.com:6666;protocol=diameter</PrimEventChargFunction>
<PrimChargCollectionFunction>aaa://host.example1.com:6666;protocol=diameter</PrimChargCollectionFunction>
<SecEventChargFunction>aaa://host.example2.com:6666;protocol=diameter</SecEventChargFunction>
<SecChargCollectionFunction>aaa://host.example2.com:6666;protocol=diameter</SecChargCollectionFunction>
</ent>
<ent name="HssChargingInfo" ns="bn">
<ChargingID>ChargingID-5</ChargingID>
<PrimEventChargFunction>aaa://host.example1.com:1813;transport=udp;protocol=radius</PrimEventChargFunction>
<PrimChargCollectionFunction>aaa://host.example1.com:1813;transport=udp;protocol=radius</PrimChargCollectionFunction>
<SecEventChargFunction>aaa://host.example2.com:1813;transport=udp;protocol=radius</SecEventChargFunction>
<SecChargCollectionFunction>aaa://host.example2.com:1813;transport=udp;protocol=radius</SecChargCollectionFunction>
</ent>
<!-- HSS-Scscf-Server  -->
<ent name="HssScscfServer" ns="bn">
<ServerCapabilitiesID>ServerCapabilitiesID-1</ServerCapabilitiesID>
<ServerName>ScscfMontreal</ServerName>
<MandatoryCapability>1</MandatoryCapability>
<OptionalCapability>2</OptionalCapability>
</ent>
<ent name="HssScscfServer" ns="bn">
<ServerCapabilitiesID>ServerCapabilitiesID-2</ServerCapabilitiesID>
<ServerName>ScscfToronto</ServerName>
<MandatoryCapability>3</MandatoryCapability>
<OptionalCapability>4</OptionalCapability>
</ent>
<ent name="HssScscfServer" ns="bn">
<ServerCapabilitiesID>ServerCapabilitiesID-3</ServerCapabilitiesID>
<ServerName>ScscfDouala</ServerName>
<MandatoryCapability>5</MandatoryCapability>
<OptionalCapability>6</OptionalCapability>
</ent>
<ent name="HssScscfServer" ns="bn">
<ServerCapabilitiesID>ServerCapabilitiesID-4</ServerCapabilitiesID>
```

```
<ServerName>ScscfParis</ServerName>
<MandatoryCapability>7</MandatoryCapability>
<OptionalCapability>9</OptionalCapability>
</ent>
<ent name="HssScscfServer" ns="bn">
<ServerCapabilitiesID>ServerCapabilitiesID-5</ServerCapabilitiesID>
<ServerName>ScscfQuebec</ServerName>
<MandatoryCapability>10</MandatoryCapability>
<OptionalCapability>11</OptionalCapability>
</ent>
<ent name="HssScscfServer" ns="bn">
<ServerCapabilitiesID>ServerCapabilitiesID-6</ServerCapabilitiesID>
<ServerName>ScscfTokyo</ServerName>
<MandatoryCapability>12</MandatoryCapability>
<OptionalCapability>13</OptionalCapability>
</ent>
<ent name="HssScscfServer" ns="bn">
<ServerCapabilitiesID>ServerCapabilitiesID-7</ServerCapabilitiesID>
<ServerName>Vancouver</ServerName>
<MandatoryCapability>11</MandatoryCapability>
<OptionalCapability>12</OptionalCapability>
</ent>
<!-- HSS-Authorized-Visited-Networks  -->
<ent name="HssAuthorizedVisitedNetworks" ns="bn">
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<NetworkIdentifier>192.168.30.30</NetworkIdentifier>
</ent>
<ent name="HssAuthorizedVisitedNetworks" ns="bn">
<RoamingProfileID>RoamingProfileID-2</RoamingProfileID>
<NetworkIdentifier>192.168.30.31</NetworkIdentifier>
</ent>
<ent name="HssAuthorizedVisitedNetworks" ns="bn">
<RoamingProfileID>RoamingProfileID-3</RoamingProfileID>
<NetworkIdentifier>192.168.30.32</NetworkIdentifier>
</ent>
<ent name="HssAuthorizedVisitedNetworks" ns="bn">
<RoamingProfileID>RoamingProfileID-4</RoamingProfileID>
<NetworkIdentifier>192.168.30.33</NetworkIdentifier>
</ent>
<ent name="HssAuthorizedVisitedNetworks" ns="bn">
<RoamingProfileID>RoamingProfileID-5</RoamingProfileID>
<NetworkIdentifier>192.168.30.34</NetworkIdentifier>
</ent>
<!-- HSS-Subscription-Profile  -->
<!-- Create as many subscription profiles as needed -->
<!-- HSS-privateSubscription-1 -->
<!-- HSS-Subscription -->
<ent name="Subscription" ns="global">
<SubscriptionID>sub-1</SubscriptionID>
</ent>
<ent name="HssSubscription" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<ChargingID>ChargingID-1</ChargingID>
</ent>
<!-- HSS-PrivateIdentity -->
<ent name="HssPrivateIdentity" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>

<PrivateIdentity>310910421010100@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>

<AlgoName>BasicDigest</AlgoName>
</ent>
<ent name="HssPrivateIdentity" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
```

```
<PrivateIdentity>310910421010101@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>

<AlgoName>GsmMilenage</AlgoName>

</ent>

<ent name="HssPrivateIdentity" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PrivateIdentity>310910421010102@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<AlgoName>HttpDigest</AlgoName>
</ent>
<!-- HSS-Service-Profile -->
<ent name="HssServiceProfile" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<ServiceProfileID>servProf-1-1-1</ServiceProfileID>
<ServerCapabilitiesID>ServerCapabilitiesID-1</ServerCapabilitiesID>
<SubscriptionMediaProfID>1</SubscriptionMediaProfID>
</ent>
<!-- HSS-Public-Identity -->
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PublicIdentity>sip:alice1010100@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-1-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>1</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PublicIdentity>sip:alice1010101@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-1-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>1</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PublicIdentity>sip:alice1010102@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-1-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>2</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PublicIdentity>sip:alice1010103@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-1-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>2</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PublicIdentity>sip:alice1010104@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-1-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>3</ImplicitRegistrationSet>
```

```
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PublicIdentity>sip:alice1010105@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-1-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>3</ImplicitRegistrationSet>
</ent>
<!-- PrivatePublicLink -->
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PrivateIdentity>310910421010100@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010100@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PrivateIdentity>310910421010100@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010101@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PrivateIdentity>310910421010101@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010102@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PrivateIdentity>310910421010101@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010103@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PrivateIdentity>310910421010102@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010104@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<PrivateIdentity>310910421010102@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010105@blueslice.com</PublicIdentity>
</ent>
<!-- HSS-Initial-filter-Criteria -->
<ent name="HssInitialFilteringCriteria" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<ServiceProfileID>servProf-1-1-1</ServiceProfileID>
<InitialFiltCritID>ifc-1-1-1-1</InitialFiltCritID>
<iFCPriority>0</iFCPriority>
<ProfilePartIndicator>1</ProfilePartIndicator>
<ConditionTypeCNF>1</ConditionTypeCNF>
<ASName>sip:AS-98-1@homedomain.com</ASName>
<ASDefaultHandling>0</ASDefaultHandling>
<!-- ASServiceInfo>'NULL'</ASServiceInfo -->
</ent>
<!-- HSS-Service-Point-Trigger -->
<ent name="HssServicePointTrigger" ns="bn">
<SubscriptionID>sub-1</SubscriptionID>
<ServiceProfileID>servProf-1-1-1</ServiceProfileID>
<InitialFiltCritID>ifc-1-1-1-1</InitialFiltCritID>
<ServPointTriggerID>stp-1-1-1-1</ServPointTriggerID>
<ServPointTriggerType>0</ServPointTriggerType>
<GroupList>0</GroupList>
<RegistrationType>0</RegistrationType>
<ConditionNegated>0</ConditionNegated>
<!-- RequestUriInfo>1</RequestUriInfo -->
```

```
<SipMethodInfo>SUBSCRIBE</SipMethodInfo>
<!-- SipHeaderHeader>'NULL'</SipHeaderHeader -->
<!-- SipHeaderContent>'NULL'</SipHeaderContent -->
<!-- SessionCaseInfo>'NULL'</SessionCaseInfo -->
<!-- SessionDescriptionContent>'NULL'</SessionDescriptionContent -->
<!-- SessionDescriptionLine>'NULL'</SessionDescriptionLine -->
</ent>
<!-- HSS-Subscription-2 -->
<ent name="Subscription" ns="global">
<SubscriptionID>sub-2</SubscriptionID>
</ent>
<ent name="HssSubscription" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<ChargingID>ChargingID-1</ChargingID>
</ent>
<!-- HSS-PrivateIdentity -->

<ent name="HssPrivateIdentity" ns="bn">

<SubscriptionID>sub-2</SubscriptionID>

<PrivateIdentity>310910421010103@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>

<AlgoName>BasicDigest</AlgoName>

</ent>

<ent name="HssPrivateIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>

<PrivateIdentity>310910421010104@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>

<AlgoName>BasicDigest</AlgoName>

</ent>

<ent name="HssPrivateIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PrivateIdentity>310910421010105@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<AlgoName>BasicDigest</AlgoName>
</ent>
<!-- HSS-Service-Profile -->
<ent name="HssServiceProfile" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<ServerCapabilitiesID>ServerCapabilitiesID-1</ServerCapabilitiesID>
<SubscriptionMediaProfID>1</SubscriptionMediaProfID>
</ent>
<!-- HSS-Public-Identity -->
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PublicIdentity>sip:alice1010106@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>1</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PublicIdentity>sip:alice1010107@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<IdentityType>0</IdentityType>
```

```
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>1</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PublicIdentity>sip:alice1010108@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>2</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PublicIdentity>sip:alice1010109@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>2</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PublicIdentity>sip:alice1010110@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>3</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PublicIdentity>sip:alice1010111@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>3</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PublicIdentity>sip:alice1010112@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>4</ImplicitRegistrationSet>
</ent>
<ent name="HssPublicIdentity" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PublicIdentity>sip:alice1010113@blueslice.com</PublicIdentity>
<RoamingProfileID>RoamingProfileID-1</RoamingProfileID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<IdentityType>0</IdentityType>
<BarringIndication>0</BarringIndication>
<ImplicitRegistrationSet>4</ImplicitRegistrationSet>
</ent>
<!-- PrivatePublicLink -->
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PrivateIdentity>310910421010103@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010106@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
```

```
<SubscriptionID>sub-2</SubscriptionID>
<PrivateIdentity>310910421010103@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010107@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PrivateIdentity>310910421010103@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010108@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PrivateIdentity>310910421010103@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010109@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PrivateIdentity>310910421010104@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010110@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PrivateIdentity>310910421010104@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010111@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PrivateIdentity>310910421010105@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010112@blueslice.com</PublicIdentity>
</ent>
<ent name="HssPrivatePublicLink" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<PrivateIdentity>310910421010105@ims.mnc910.mcc310.blueslice.com</PrivateIdentity>
<PublicIdentity>sip:alice1010113@blueslice.com</PublicIdentity>
</ent>
<!-- HSS-Initial-filter-Criteria -->
<ent name="HssInitialFilteringCriteria" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<InitialFiltCritID>ifc-2-1-1-1</InitialFiltCritID>
<iFCPriority>0</iFCPriority>
<ProfilePartIndicator>1</ProfilePartIndicator>
<ConditionTypeCNF>1</ConditionTypeCNF>
<ASName>sip:AS-98-1@homedomain.com</ASName>
<ASDefaultHandling>0</ASDefaultHandling>
<!-- ASServiceInfo>'NULL'</ASServiceInfo -->
</ent>
<!-- HSS-Service-Point-Trigger -->
<ent name="HssServicePointTrigger" ns="bn">
<SubscriptionID>sub-2</SubscriptionID>
<ServiceProfileID>servProf-1-2-1</ServiceProfileID>
<InitialFiltCritID>ifc-2-1-1-1</InitialFiltCritID>
<ServPointTriggerID>stp-2-1-1-1</ServPointTriggerID>
<ServPointTriggerType>0</ServPointTriggerType>
<GroupList>0</GroupList>
<RegistrationType>0</RegistrationType>
<ConditionNegated>0</ConditionNegated>
<!-- RequestUriInfo>1</RequestUriInfo -->
<SipMethodInfo>REGISTER</SipMethodInfo>
<!-- SipHeaderHeader>'NULL'</SipHeaderHeader -->
<!-- SipHeaderContent>'NULL'</SipHeaderContent -->
<!-- SessionCaseInfo>'NULL'</SessionCaseInfo -->
<!-- SessionDescriptionContent>'NULL'</SessionDescriptionContent -->
<!-- SessionDescriptionLine>'NULL'</SessionDescriptionLine -->
```

```
</ent>
</file>
```

SLF subscriber provisioning file

This XML Request file in long format provisions SLF subscribers.

Prerequisite: The Hss service must be running on the blade and the SLF Server must be enabled (SLF Config) prior to being able to provision SLF subscribers.

Note: This XML file includes the creation of the SubscriptionIDs for each subscriber.

```
<file>
<tx nbreq="1"><req name="insert" ver="6.2.0" state="undefined"><ent
name="Subscription" ns="global" /><set><expr><attr name="SubscriptionID" /><op
value="="/><value val="sub1"/></expr></set></req></tx>
<tx nbreq="1"><req name="insert" ver="6.2.0" state="undefined"><ent
name="HssSlfPublic2HssName" ns="bn" /><set><expr><attr name="SubscriptionID"
/><op value="="/><value val="sub1"/></expr><expr><attr name="HssName" /><op
value="="/><value val="aaa://foreignhss.blueslice.com"/></expr><expr><attr
name="PublicIdentity" /><op value="="/><value
val="sip:slfpublic1@blueslice.qc.ca"/></expr></set></req></tx>
</file>
```

DNS Enum subscriber provisioning file

This template in XML long format provisions DNS Enum subscribers.

Prerequisite: The Hss service must be running on the blade and the ENUM Server must be enabled (ENUM Server Config) prior to being able to provision DNS ENUM subscribers.

Note: This XML file includes the creation of the SubscriptionIDs for each subscriber.

```
<file>
<!-- add subscription -->
<tx nbreq="1"><req name="insert" ver="6.2.0" state="undefined"><ent
name="Subscription" ns="global" /><set><expr><attr name="SubscriptionID" /><op
value="="/><value val="sub-1"/></expr></set></req></tx>
<!--Configure Domain Name List -->
<tx nbreq="2"><req name="insert" ver="6.2.0" state="undefined"><ent
name="DNSDomainNameList" ns="bn" /><set><expr><attr name="EnumDomainName" /><op
 value="="/><value val="e164.arpa"/></expr><expr><attr name="EnumDomainNameId"
/><op value="="/><value val="1"/></expr><expr><attr name="DefaultEnumDomainName"
 /><op value="="/><value val="0"/></expr></set></req></tx>
<!-- add ENUM user -->
<tx nbreq="3"><req name="insert" ver="6.2.0" state="undefined">
<ent name="DNSEnumUser" ns="bn" /><set><expr><attr name="SubscriptionID" /><op
value="="/><value val="sub-1"/></expr><expr><attr name="EnumUserId" /><op
value="="/><value val="1"/></expr><expr><attr name="NAPTRRegExp" /><op
value="="/><value val="!^.*$!sip:1@blueslice.com"/></expr><expr><attr
```

```
name="EnumDomainNameId" /><op value="="/><value val="1"/></expr></set></req></tx>
</file>
```

AAA subscriber provisioning file

This XML Request file in short format provisions AAA subscriber profiles.

Prerequesite: The Hss service must be running on the blade and the AAA Server must be enabled (AAA Config) prior to being able to provision AAA subscribers.

Note: This XML file includes the creation of the SubscriptionIDs for each subscriber.

```

<file>
<!—Creation of SubscriptionIDs -->
<!-- Create as many subscription profiles as needed -->
<ent name="Subscription" ns="global">
        <SubscriptionID>sub0</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub1</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub2</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub3</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub4</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub5</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub6</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub7</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub8</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub9</SubscriptionID>
    </ent>
    <ent name="Subscription" ns="global">
        <SubscriptionID>sub10</SubscriptionID>
    </ent>
<!—AAA-Subscription-Profile-->
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-0</AAAUserName>
        <SubscriptionId>sub0</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
```

```
        <AAAUserName>aaauser-1</AAAUserName>
        <SubscriptionId>sub1</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-2</AAAUserName>
        <SubscriptionId>sub2</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-3</AAAUserName>
        <SubscriptionId>sub3</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-4</AAAUserName>
        <SubscriptionId>sub4</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-5</AAAUserName>
        <SubscriptionId>sub5</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-6</AAAUserName>
        <SubscriptionId>sub6</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-7</AAAUserName>
        <SubscriptionId>sub7</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-8</AAAUserName>
        <SubscriptionId>sub8</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-9</AAAUserName>
        <SubscriptionId>sub9</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
    <ent name="AAAUserId" ns="bn">
        <AAAUserName>aaauser-10</AAAUserName>
        <SubscriptionId>sub10</SubscriptionId>
        <AuthMethod>0</AuthMethod>
        <ServiceType>2</ServiceType>
        <FramedProtocol>7</FramedProtocol>
    </ent>
</file>
```

LTE-HSS subscriber provisioning templates

Prerequisites: HLR-proxy=On

- The LteHss service must be running on the blade (LTE-HSS configuration must already have been completed).
- The LTE-HSS must be enabled (in LTE-HSS Config).
- At least one PDN Context template must already have been configured in the system.
- A subscription must already be created for the subscriber with the SIM/MSISDN/SimImsiMap/MsIsdnImsiProfileAssociation and HLR subscriber profile (with service profile) already defined.

  Note: When defining a SIM with a SimType=Offboard, an AlgorithName must still be provided. The AlgorithName can be any of the ones already configured in the system AuC.
- The HLR-Proxy functionality must have been configured. To achieve this, the LteHssImsiRangeConfig entity must be configured per IMSI Ranges. For the SDM ngHLR's HLRProxy to be able to proxy messages to the external HLR hosting the subscriber, the LteHssImsiRangeConfig must be configured for the subscriber's IMSI Range. You can only configure this entity through the Tekelec CLI, refer to the HLR Proxy functionality section of the SDM System Configuration – Reference Manual for details on the LteHssImsiRangeConfig entity and the CLI Navigation and syntax. For general instructions on how to configure/provision through the CLI, refer to section of this document.

Example: HLR-proxy=On

```
<?xml version="1.0" encoding="utf-8"?>
<file>

<tx nbreq="1"><req name="insert" ver="6.2.1" state="undefined"><ent
name="SubscriberProfile" ns="bn" /><set><expr><attr name="SubscriptionID"
/><op value="="/><value val="ltesub4"/></expr><expr><attr
name="HlrServiceProfileID" /><op value="="/><value
val="1"/></expr><expr><attr name="PreferredRoutingNetworkDomain" /><op
value="="/><value val="1"/></expr><expr><attr name="DefaultPdnContextId"
/><op value="="/><value val="1"/></expr><expr><attr
name="SpPdnChargingCharacteristics" /><op value="="/><value
val="HotBilling"/></expr><expr><attr name="AMBRUL" /><op value="="/><value
val="2000"/></expr><expr><attr name="AMBRDL" /><op value="="/><value
val="1000"/></expr><expr><attr name="HlrProxyMode" /><op value="="/><value
val="1"/></expr></set></req></tx>

<tx nbreq="1"><req name="insert" ver="6.2.1" state="undefined"><ent
name="ServiceProfilePDNContext" ns="bn" /><set><expr><attr
name="HlrServiceProfileID" /><op value="="/><value
```

```
val="1"/></expr><expr><attr name="SubscriptionID" /><op value="="/><value
val="ltesub4"/></expr><expr><attr name="PdnContextId" /><op
value="="/><value
val="5"/></expr><expr><attr name="PdnType" /><op value="="/><value
val="1"/></expr><expr><attr name="PdnAddress1" /><op value="="/><value
val="30.30.30.10"/></expr><expr><attr name="PdnTemplateId" /><op
value="="/><value val="1"/></expr></set></req></tx>

</file>
```

Prerequisites: HLR-proxy=Off

- The LteHss service must be running on the blade (LTE-HSS configuration must already have been completed).
- The LTE-HSS must be enabled (in LTE-HSS Config).
- At least one PDN Context template must already have been configured in the system.
- A subscription must already be created for the subscriber with the SIM/MSISDN/SimImsiMap/MsIsdnImsiProfileAssociation and HLR subscriber profile (with service profile) already defined.

  Note: When defining a SIM with a SimType=Offboard, an AlgorithName must still be provided. The AlgorithName can be any of the ones already configured in the system AuC.

Example: HLR-proxy=Off

```
<?xml version="1.0" encoding="utf-8"?>
<file>
<tx nbreq="1"><req name="insert" ver="6.2.1" state="undefined"><ent
name="SubscriberProfile" ns="bn" /><set><expr><attr name="SubscriptionID"
/><op value="="/><value val="ltesub4"/></expr><expr><attr
name="HlrServiceProfileID" /><op value="="/><value
val="1"/></expr><expr><attr name="PreferredRoutingNetworkDomain" /><op
value="="/><value val="1"/></expr><expr><attr name="DefaultPdnContextId"
/><op value="="/><value val="1"/></expr><expr><attr
name="SpPdnChargingCharacteristics" /><op value="="/><value
val="HotBilling"/></expr><expr><attr name="AMBRUL" /><op value="="/><value
val="2000"/></expr><expr><attr name="AMBRDL" /><op value="="/><value
val="1000"/></expr><expr><attr name="HlrProxyMode" /><op value="="/><value
val="0"/></expr></set></req></tx>

<tx nbreq="1"><req name="insert" ver="6.2.1" state="undefined"><ent
name="ServiceProfilePDNContext" ns="bn" /><set><expr><attr
name="HlrServiceProfileID" /><op value="="/><value
val="1"/></expr><expr><attr name="SubscriptionID" /><op value="="/><value
val="ltesub4"/></expr><expr><attr name="PdnContextId" /><op
value="="/><value
val="5"/></expr><expr><attr name="PdnType" /><op value="="/><value
val="1"/></expr><expr><attr name="PdnAddress1" /><op value="="/><value
val="30.30.30.10"/></expr><expr><attr name="PdnTemplateId" /><op
value="="/><value val="1"/></expr></set></req></tx>

</file>
```

Update request template

This template and the invoking file modify a basic HLR subscriber.

The logic in this example apply to the entities of all the other applications (HLR, SIP, HSS, SLF, AAA, DNS ENUM).

Template file

```
<file>
<!--Template request to update a basic HLR subscriber-->
<req name="update" id="100">
<ent name="SubscriberProfile" ns="bn"/>
<set>
    <expr>
        <attr name="AtiSubsInfoLevel"/><op value="="/><value val=""/>
    </expr>
    <expr>
        <attr name="SubsRoamingMsgOn"/><op value="="/><value val=""/>
    </expr>
</set>
<where>
    <expr>
        <attr name="SubscriptionID"/><op value="="/><value val=""/>
    </expr>
    <op value="and"/>
    <expr>
        <attr name="HlrServiceProfileID"/>
    <op value="="/><value val="1"/>
    </expr>
</where>
</req>
<!--Command Template to update a basic HLR subscriber-->
<template id="1" otherAttributesModifiable="N">
        <attr name="AtiSubsInfoLevel" mandatory="Y"/>
        <attr name="SubsRoamingMsgOn" mandatory="Y"/>
        <attr name="SubscriptionID" mandatory="Y"/>
        <tr id="100"/>
</template>
</file>
```

Invoking file

```
<file>
<!--Invoking file to update a basic HLR subscriber-->
    <tp id="1">
        <tpi nm="SubsRoamingMsgOn" val="1"/>
        <tpi nm="AtiSubsInfoLevel" val="2"/>
        <tpi nm="SubscriptionID" val="0000001"/>
    </tp>
</file>
```

Delete XML request template

This template file deletes an MSISDN entry.

The logic in this example applies to the entities of all the other applications (HLR, SIP, HSS, SLF, AAA, DNS ENUM).

Template file

```
<file>
<!--Template request to delete a MSISDN--req name="delete" id="200">>
<
<ent name="MSISDN" ns="bn" />
<where>
    <expr><attr name="SubscriptionID"/>
    <op value="="/><value val=""/></expr>
    <op value="and"/>
    <expr><attr name="MsIsdn"/>
    <op value="="/><value val=""/></expr>
</where>
</req>
<!--Command Template to delete a MSISDN->
<template id="2" otherAttributesModifiable="N">
    <attr name="SubscriptionID" mandatory="Y" />
    <attr name="MsIsdn" mandatory="Y" />
    <tr id="200" />
</template>
</file>
```

Invoking file

```
<file>
<!-- Invoking file to delete a basic HLR subscriber-->
<tp id="2">
    <tpi nm="MsIsdn" val="15643212201" />
    <tpi nm="SubscriptionID" val="0000001" />
</tp>
</file>
```

Select XML request file

This XML file selects and entity (HssServiceProfile). The Select operation cannot be loaded with a template.

The logic in this example applies to the entities of all the other applications (HLR, SIP, SLF, AAA, DNS ENUM).

```
<tx>
```

```
    <req name="select">
        <ent name=" HssServiceProfile " ns="bn" />
            <select>
                <expr><attr name="ServerCapabilitiesID" ent="HssServiceProfile"
 ns="bn" />
                </expr>
                <expr><attr name="IMSSubscriptionID" ent="HssServiceProfile"
ns="bn" />
                </expr>
                <expr><attr name="ServiceProfileID" ent="HssServiceProfile"
ns="bn" />
                </expr>
              <expr><attr name="SubscriptionMediaProfID" ent="HssServiceProfile"
 ns="bn"/>
                </expr>
            </select>
            <where>
            <expr>
                <expr><attr name="IMSSubscriptionID" ent="HssServiceProfile"
ns="bn" />
                <op value="="/><value val="sub-4"/>
                </expr><op value="and"/>
                <expr><attr name="ServiceProfileID" ent="HssServiceProfile"
ns="bn" />
                <op value="="/><value val="servProf-1-14"/>
                </expr>
                </expr>
            </where>
    </req>
</tx>
```

Request-Reply XML file

Here is an example of an operation type of XML Request followed by the system XML reply.

The logic in this example applies to all the other operations of each supported application (HLR, SIP, SLF, AAA, DNS ENUM).

XML request: Operation

```
<file>
<req name="operation">
<oper name="HssStatistics" ent="HssStatistics" ns="bn"/>
<oper name="GetNumActivePeers" ent="HssStatistics" ns="bn"/>
</req>
</file>
```

XML reply: Operation

```
<tx nbreq="1"><req name="operation" ver="6.2.0" state="undefined"><oper
name="HssStatistics" ent="HssStatistics" ns="bn"/><oper
name="GetNumActivePeers" ent="HssStatistics" ns="bn"/><res error="0"
affected="1"></res><rset><row><rv>0</rv></row></rset></req></tx>
```

Chapter 6

Converting an original XML request file to an XML template file

Topics:

- *About XML request files and XML template files.....89*
- *About XML long and short formats.....89*
- *Comparison of XML request files and XML template files.....89*
- *Create XML template request.....91*
- *Create command template.....94*
- *Create invoking file.....96*

Subscriber provisioning through templates is highly efficient for bulk provisioning by minimizing operator efforts and increasing provisioning performance.

Original XML Request files of request types insert, update, delete can be easily converted to an XML template file by adding a unique identification to the template request and creating both, a command file and an invoking file.

These sections demonstrate how to convert an original XML Request file to an XML template file.

- Differences of XML Request file and XML Template file
- Create a template request
- Create a command template
- Create an invoking file

About XML request files and XML template files

An XML request file, which is also known as the original XML request, can contain update, insert, delete, select, and operation requests, and the file is stored in the SDM database. The XML Request file supports all XML requests in long format; and it supports the Insert request in short format.

The XML template file is made up of one or more template requests and a command template.

- Each template request can contain only one update, insert, or delete request. Template definitions are stored in the database in the original XML request format.
- The command template contains the constraints for template attributes that can be overwritten as well as the template requests it uses.

The template file requires an invoking file, which calls the command template and contains the values for the attributes defined in the Command template. The invoking file is the file used for the actual provisioning.

About XML long and short formats

In XML short format, the operator specifies the entity name, the entity namespace, and the list of attributes and values required.

```
<ent name=”EntityName” ns=”Namespace”>
   <attr>attribute value</ attr >
</ent>
```

In XML long format, the operator specifies request identification, entity identification, and a set of attributes and values.

```
<req name="delete">
   <ent name = " EntityName " ns=" Namespace "/>
   <where>
      <expr>
         <attr name=" AttributeName"/>
         <op value=" Operator "/>
         <value val=" value "/>
      </expr>
   </where>
</req>
```

Comparison of XML request files and XML template files

This table provides the major differences between the original XML Request file and an XML template file.

Table 11: Conversion summary

Delta/Type of file	Provisioning with XML Request file	Provisioning with template files
Number of files needed	1file	2 files: • Template file • Invoking file
Content	Multiple requests	Template file: • Multiple Template requests • One single CommandTemplate Invoking file (includes provisioning commands and values)
Types of XML requests supported	• Update • Insert • Delete • Select * • Operation * *Requests of type select and operation must be written in XML files without using templates.	• Insert • Delete • Update
Content Comparison	Contains a series of requests.	Template file: • Contains a series of Template requests, which are based on the original XML requests, but simply written in a slightly different format. • Contains a Command Template that refers to the Template requests and that defines which attributes are mandatory or not. For the mandatory attributes, values must absolutely be provided in the Invoking file. Invoking file • Contains the values for the attributes defined in the Command Template.
File Format	`<file> <req name="RequestType"> <ent name = "EntityName" ns="Namespace"/> <set>`	< `file> <!- - Template requests - -> <req`

Delta/Type of file	Provisioning with XML Request file	Provisioning with template files
(legend: in red is the delta in green are the comments)	`<expr><attr name="AttributeName"/> <value val="value"/> </expr> <! - - etc…- -> </set> </req> <req name="RequestType"> <ent name = "EntityName" ns="Namespace"/> <set> <expr><attr name="AttributeName"/> <value val="value"/> </expr> </set> </req> <! - - etc…- -> </file>` **Note:** Supports XML files in long and short formats.	`name="RequestType" id= “trid#”> <ent name = "EntityName" ns="Namespace"/> <set> <expr><attr name="AttributeName"/> <value val=""/> </expr> <expr><attr name="AttributeName"/> <value val="value"/> </expr> <! - - etc…- -> </set> </req> <req name="RequestType" id= ““trid#””> <ent name = "EntityName" ns="Namespace"/> <set> <expr><attr name="AttributeName"/> <value val="value"/> </expr> </set> </req> <! - - etc…- -> <!- - Command Template - -> <template id= “tpid#” otherAttributesModifiable= “Y/N”> <attr name= “AttributeName" mandatory= “Y/N”/> <attr name= “AttributeName" mandatory= “Y/N”/> <! - - etc…- -> <tr id=“trid#”/> <tr id=“trid#”/> <! - - etc…- -> </template> </file> <!- - Invoking file - -> <file> <tp id= “tpid#”> <tpi nm= “AttributeName" val= “value”/> <tpi nm= “AttributeName" val= “value”/> </tp> </file>` **Note:** XML templates only support long format requests.

Create XML template request

Adding the template request identification to an existing XML request.

- SubscriptionID and SIM have already been provisioned
- Original XML Request file with multiple XML requests.

Original XML Request file with multiple XML requests.

```
<file>
```

```
<!--Add basic subscriber-->
<req name="insert">
<ent name="SubscriberProfile" ns="bn"/>
<set>
<expr><attr name="SubscriptionID"/><value
val="1"/></expr>
<expr><attr name="HlrServiceProfileID"/><value
val="1"/></expr>
<expr><attr name="TeleServiceList"/><value
val="TS11,TS21,TS22"/></expr>
<expr><attr name="OdbMask"/><value
val="PremiumRateInfo,PremiumRateEntertainment,RoamingOutsideHplmnCountry"/></expr>
<expr><attr name="MsIsdnAlertInd"/><value
val=""/></expr>
<expr><attr
name="PreferredRoutingNetworkDomain"/><value
val="1"/></expr>
<expr><attr name="OCPlmnTemplateId"/><value
val="0"/></expr>
<expr><attr name="Nam"/><value val="0"/></expr>
</set>
</req>
<req name="insert">
<ent name="MSISDN" ns="bn"/>
<set>
<expr><attr name="SubscriptionID"/><value
val="1"/></expr>
<expr><attr name="MsIsdn"/><value
val="15634210100"/></expr>
</set>
</req>
<req name="insert">
<ent name="MsIsdnImsiProfileAssociation" ns="bn"/>
<set>
<expr><attr name="SubscriptionID"/><value val=
"1"/></expr>
<expr><attr name="HlrServiceProfileID"/><value
val="1"/></expr>
<expr><attr name="Imsi"/><value
val="310910421000100"/></expr>
<expr><attr name="MsIsdn"/><value
val="15634210100"/></expr>
<expr><attr name="Displayed"/><value val="1"/>
</expr>
</set>
</req>
</file>
```

This procedure describes how to convert an original XML Request file made up of multiple XML requests into multiple XML template requests, a Command template, and an Invoking file .

1. Locate the first XML Request in the original XML Request file. Look for the **`req`** start and end tags .

```
<req name="insert" >
<ent name="SubscriberProfile" ns="bn"/>
<set>
<expr><attr name="SubscriptionID"/><value
val="1"/></expr>
<expr><attr name="HlrServiceProfileID"/><value
val="1"/></expr>
<expr><attr name="TeleServiceList"/><value
```

```
val="TS11,TS21,TS22"/></expr>
<expr><attr name="OdbMask"/><value
val="PremiumRateInfo,PremiumRateEntertainment,RoamingOutsideHplmnCountry"/></expr>
<expr><attr name="MsIsdnAlertInd"/><value
val=""/></expr>
<expr><attr
name="PreferredRoutingNetworkDomain"/><value
val="1"/></expr>
<expr><attr name="OCPlmnTemplateId"/><value
val="0"/></expr>
<expr><attr name="Nam"/><value val="0"/></expr>
</set>
</req>
```

2. Modify the request by adding a request ID next to the request name value; for example, add **`id="101".`**

```
</req>
<req name="insert"> id="101"
<set>
<expr><attr name="SubscriptionID"/><value
val="1"/></expr>
```

3. Modify the attributes as needed.
   a) Identify the mandatory attributes of the entity defined in the template request.
   b) Review the values of each attribute to determine which ones should be left 'blank'.

   In general, if the mandatory attribute needs to be provided in the Command template or in the Invoking file, the value of this attribute can be left blank. This value will be overwritten by the value in the Command template or the Invoking file.

   In this example, mandatory attributes are:

   - **`SubscriptionID`** - The value can be left blank because it will be the only one required in the Invoking file.

```
<expr><attr name="SubscriptionID"/><value
val=""/></expr>
```

   - **`HlrServiceProfileID`** - this attribute must have a value in the template request because this attribute currently supports only one value '1', therefore the value is static and must not be provided in the Command Template.
   - **`PreferredRoutingNetworkDomain`** - This value can be provided in the template request if it is always the same.

4. Locate the second XML Request in the original XML Request file.
5. Enter a new request id, for example, enter **`id="102"`**

```
<req name="insert"> id="102"
<ent name="MSISDN" ns="bn"/>
<set>
<expr><attr name="SubscriptionID"/><value
val="1"/></expr>
<expr><attr name="MsIsdn"/><value
val="15634210100"/></expr>
</set>
</req>
```

6. Modify the attributes as needed.

   The mandatory attributes are:

   - **`SubscriptionID`** - The value can be left blank because it will be the only one required in the Invoking file.
   - **`MsIsdn`** - The value must be left blank because the Command template set the modification of this attribute to mandatory; therefore, the value must be specified in the Invoking vile.

7. Locate the third XML Request in the original XML Request file.
8. Enter a new request id, for example, enter **`id="103"`**

```
<req name="insert" id="103">
<ent name="MsIsdnImsiProfileAssociation" ns="bn" />
<set>
<expr><attr name="SubscriptionID"/><value val=
"1"/></expr>
<expr><attr name="HlrServiceProfileID"/><value
val="1"/></expr>
<expr><attr name="Imsi"/><value
val="310910421000100"/></expr>
<expr><attr name="MsIsdn"/><value
val="15634210100"/></expr>
<expr><attr name="Displayed"/><value val="1"/>
</expr>
</set>
</req>
```

9. Modify the attributes as needed.

   The mandatory attributes are:

   - **`SubscriptionID`** - The value can be left blank because it will be the only one required in the Invoking file.
   - **`HlrServiceProfileID`** - this attribute must have a value in the template request because this attribute currently supports only one value '1', therefore the value is static and must not be provided in the Command Template.
   - **`Imsi`** - The value must be left blank because the Command template set the modification of this attribute to mandatory; therefore, the attribute and its value must absolutely be defined in the Invoking file.
   - **`MsIsdn`** - The value must be left blank because the Command template set the modification of this attribute to mandatory; therefore, the attribute and its value must absolutely be defined in the Invoking file.

Create command template

1. Create the command template structure.

```
<template id=”Command template  ID number” otherAttributesModifiable=”Y or N”>

<-!List all attributes allowed to be modified by the invoking file. -->
```

```
<attr name=”Name of the modifiable attribute 1” mandatory=”Y or N”/>
<attr name=”Name of the modifiable attribute 2” mandatory=”Y or N”/>
<attr name=”Name of the modifiable attribute 3” mandatory=”Y or N”/>

<-!List all ID numbers of the template requests to be included. -->

<tr id=”Template request ID number 1”>
<tr id=”Template request ID number 2”>
<tr id=”Template request ID number 3”>

</template>
```

2. Type the command template ID number.

```
<template id=”1” otherAttributesModifiable=”Y or N”>
```

3. For otherAttributesModifiable, declare whether any attributes in the referenced template request files can or cannot be modified by the commands in the invoking file.
   - Type **Y** if the values of the attributes that are defined in the referenced template requests and that are not listed in the Attribute Name section of the command template, can be modified by commands in the Invoking file. In this case, any of the attributes defined in the referenced template requests can be provided in the Invoking file commands along with new values and these new values will overwrite the ones defined in the template requests.
   - Type **N** if the values of the attributes that are defined in the referenced template requests and that are not listed in the Attribute Name section of the command template, cannot be modified by commands in the Invoking file. Only attributes listed in the command template can be modified by commands in the Invoking file.

```
<template id=”1” otherAttributesModifiable=”N”>
```

4. Type each attribute name to be modified by the invoking file.
5. For each attribute name, declare whether the modification by the invoking file is mandatory.
   - Type **Y** if the attribute and its value must absolutely be defined in the Invoking file. All the attributes that have been implemented as Mandatory in the system database must have the XML field Mandatory set to **Y**.
   - Type **N** if the attribute and its value may or may not be defined in the Invoking file.
     - The attribute is not required to be included in the Invoking file.

       The default value of the XML field mandatory is **N**. This means that when the mandatory field is not specified, as shown in the example, the Nam attribute and its value are not required in the Invoking file.
     - An attribute that has been implemented as Mandatory in the system database cannot have the XML field Mandatory set to **N**.
     - All attributes that have been implemented as Optional in the system database can either have the XML field Mandatory set to **Y** or **N**.

```
<!—Command Template for adding basic subscriber. -->
<template id="1"otherAttributesModifiable="N">
<attr name="SubscriptionID" mandatory="Y"/>
<attr name="Imsi" mandatory="Y"/>
```

```
<attr name="MsIsdn" mandatory="Y"/>
<attr name="Nam" mandatory="N"/>
<tr id="101"/>
<tr id="102"/>
<tr id="103"/>
</template>
```

Create invoking file

1. Create an XML request using this format:

```
<tp id=”Template ID number”>
<tpi nm=”Name of the modifiable attribute to over ride” val=”value”/>
<tpi nm=”Name of the modifiable attribute to override” val=”value”/>
<tpi rid=”Template request ID number” nm=”Name of the modifiable attribute to
override” val=”value”/>
```

2. Enter the identification number of the command template.
3. Enter the list of attributes that must be modified and their values.

```
<file>
<!--Invoking file to add a basic HLR subscriber-->
<tp id="1">
<tpi nm="SubscriptionID" val="7"/>
<tpi nm="Imsi" val="310910421000106"/>
<tpi nm="MsIsdn" val="15634210106"/>
<tpi nm="Nam" val="1"/>
</tp>
</file>
```

Note: If multiple attributes have the same name within the template file, and you wish to modify only one specific attribute, then specify the template request id in which the attribute that you wish to modify is defined.

```
<tpi rid="template request id number" nm="Nam" val="1"/>
```

If the ‘rid= “Template request ID number”’ is not specified, the value of all the attributes with that same name will be modified with the new value.

The final output includes the results from these procedures:

- Create a template request
- Create a command template

This output is an example of the original XML Request file that provisions a basic HLR subscriber profile. This example does not include the provisioning of a SubscriptionID or the SIM. In this context, it is assumed that they have already been provisioned in other XML Template files.

```
<file>
<!--Template file to add a basic subscriber. --> <!--Template requests for adding
 basic subscriber-->
```

```
<req name="insert" id="101">
<ent name="SubscriberProfile" ns="bn" />
<set>
<expr><attr name="SubscriptionID"/><value
val=""/></expr>
<expr><attr
name="HlrServiceProfileID"/><value
val="1"/></expr>
<expr><attr name="TeleServiceList"/><value
val="TS11,TS21,TS22"/></expr>
<expr><attr name="OdbMask"/><value
val="PremiumRateInfo,PremiumRateEntertainment,RoamingOutsideHplmnCountry"/></expr>
<expr><attr name="MsIsdnAlertInd"/><value
val=""/></expr>
<expr><attr
name="PreferredRoutingNetworkDomain"/><value
val="1"/></expr>
<expr><attr name="OCPlmnTemplateId"/><value
val="0"/></expr>
<expr><attr name="Nam"/><value
val="0"/></expr>
</set>
</req>
<req name="insert" id="102">
<ent name="MSISDN" ns="bn" />
<set>
<expr><attr name="SubscriptionID"/><value
val=""/></expr>
<expr><attr name="MsIsdn"/><value
val=""/></expr>
</set>
</req>
<req name="insert" id="103">
<ent name="MsIsdnImsiProfileAssociation" ns="bn" />
<set>
<expr><attr name="SubscriptionID"/><value
val=""/></expr>
<expr><attr name="Imsi"/><value
val=""/></expr>
<expr><attr name="MsIsdn"/><value
val=""/></expr>
<expr><attr name="Displayed"/><value
val="1"/></expr>
<expr><attr
name="HlrServiceProfileID"/><value
val="1"/></expr>
</set>
</req>
<!--Command Template for adding basic subscriber-->
<template id="1" otherAttributesModifiable="N">
<attr name="SubscriptionID" mandatory="Y"/>
<attr name="Imsi" mandatory="Y"/>
<attr name="MsIsdn" mandatory="Y"/>
<attr name="Nam" mandatory="N"/>
<tr id="101"/>
<tr id="102"/>
<tr id="103"/>
</template>
</file>
<file>
<!--Invoking file to add a basic HLR subscriber-->
<tp id="1">
<tpi nm="SubscriptionID" val="7"/>
<tpi nm="Imsi" val="310910421000106"/>
```

```
<tpi nm="MsIsdn" val="15634210106"/>
<tpi nm="Nam" val="1"/>
</tp>
</file>
```

Chapter
7

Subscriber Profile Repository (SPR)

Topics:

- *Subscription Profile Repository (SPR) 100*

Subscription Profile Repository (SPR)

SPR subscriber provisioning (XML)

Prerequisite: The Hss service must be running on the blade and the HSS Server must be enabled (Hss Config) prior to being able to provision the SPR Subscriber.

The keys to access a subscriber can be AccountId, IMSI, MSISDN, or NAI if defined These samples use MSISDN.

Display subscriber profile by MSISDN

```
<req name="select">
    <ent name="Subscriber" ns="policy"/>
    <select>
        <expr><attr name="MSISDN"/></expr>
        <expr><attr name="Entitlement"/></expr>
        <expr><attr name="Tier"/></expr>
        <expr><attr name="BillingDay"/></expr>
    </select>
    <where>
        <expr><attr name="MSISDN"/><op value="="/><value
val="33123654862"/></expr>
    </where>
</req>
```

Add SPR subscriber

```
<req name="insert">
    <ent name="Subscriber" ns="policy"/>
    <set>
        <expr><attr name="AccountID"/><value val="10404723525"/></expr>
        <expr><attr name="MSISDN"/><value val="+33123654862"/></expr>
        <expr><attr name="IMSI"/><value val="184569547984229"/></expr>
        <expr><attr name="BillingDay"/><value val="1"/></expr>
       <expr><attr name="Entitlement"/><value val="DayPass,DayPassPlus"/></expr>

    </set>
</req>
```

Delete SPR subscriber by MSISDN

```
<req name="delete">
    <ent name="Subscriber" ns="policy"/>
    <where>
        <expr><attr name="MSISDN"/><op value="="/><value
val="33123654862"/></expr>
    </where>
</req>
```

Add/update billing day by MSISDN

```
<req name="update">
    <ent name="Subscriber" ns="policy"/>
    <set>
    <expr><attr name="BillingDay"/><value val="23"/></expr>
    </set>
    <where>
        <expr><attr name="MSISDN"/><op value="="/><value
val="33123654862"/></expr>
    </where>
</req>
```

Remove billing day by MSISDSN

```
<req name="update">
    <ent name="Subscriber" ns="policy"/>
    <set>
        <expr><attr name="BillingDay"/><op value="="/><value val=""
isnull="y"/></expr>
    </set>
    <where>
        <expr><attr name="MSISDN"/><op value="="/><value
val="33123654862"/></expr>
    </where>
</req>
```

Add entitlement value by MSISDN

```
<req name="update">
    <ent name="Subscriber" ns="policy"/>
    <set>
    <oper name="AddToSet" ns="bn">
        <expr><attr name="Entitlement"/><value val="a"/></expr>
    </oper>
    </set>
    <where>
        <expr><attr name="MSISDN"/><op value="="/><value
val="33123654862"/></expr>
    </where>
</req>
```

Remove entitlement value by MSISDN

```
<req name="update">
    <ent name="Subscriber" ns="policy"/>
    <set>
        <oper name="RemoveFromSet" ns="bn">
        <expr><attr name="Entitlement"/><value val="a"/></expr>
        </oper>
    </set>
    <where>
        <expr><attr name="MSISDN"/><op value="="/><value
val="33123654862"/></expr>
    </where>
</req>
```

Display State value by MSISDN

```
<req name="select">
    <ent name="Subscriber" ns="policy"/>
    <select>
        <expr><attr name="State"/></expr>
    </select>
    <where>
        <expr><attr name="MSISDN"/><op value="="/><value
```

```
val="33123654862"/></expr>
    </where>
</req>
```

Add/Update State value by MSISDN

```
<req name="update">
    <ent name="Subscriber" ns="policy"/>
    <set>
        <expr><attr name="Status"/><op value="="/><cdata>
<![CDATA[<?xml version=\"1.0\" encoding=\"UTF-8\"?>
<state>
    <version>1</version>
    <property>
        <name>mcc</name>
        <value>315</value>
    </property>
    <property>
        <name>expire</name>
        <value>2010-02-09T11:20:32</value>
    </property>
    <property>
        <name>approved</name>
        <value>yes</value>
    </property>
</state>
]]></cdata></expr>
    </set>
    <where>
        <expr><attr name="MSISDN"/><op value="="/><value
val="33123654862"/></expr>
    </where>
</req>
```

Remove State value by MSISDN

```
<req name="update">
    <ent name="Subscriber" ns="policy"/>
    <set>
       <expr><attr name="State"/><op value="="/><value val="" isnull="y"/></expr>

    </set>
    <where>
        <expr><attr name="MSISDN"/><op value="="/><value
val="33123654862"/></expr>
    </where>
</req>
```

Add/Update dynamic quota

```
<req name="update">
  <ent name="Subscriber" ns="policy"/>
  <set>
    <expr><attr name="DynamicQuota"/><op value="="/><cdata>
    <![CDATA[<?xml version="1.0" encoding="UTF-8"?>
    <usage>
    <version>1</version>
    <dynamicquota name="AggregateLimit">
    <Type>Roll-Over</Type>
    <InstanceId>15678</InstanceId>
    <Priority>4</Priority>
    <InitialTime>135</InitialTime >
    <InitialTotalVolume>2000</InitialTotalVolume>
    <InitialInputVolume>1500</InitialInputVolume>
    <InitialOutputVolume>500</InitialOutputVolume>
    <InitialServiceSpecific>4</InitialServiceSpecific>
    <ActivationDateTime>32</ActivationDateTime>
    <ExpirationDateTime>28</ExpirationDateTime>
    <InterimReportingInterval>100</InterimReportingInterval>
</dynamicquota></usage>
    ]]></cdata></expr>
    </set>
  <where>
    <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

  </where>
</req>
```

Delete dynamic quota

```
<req name="update">
<ent name="Subscriber" ns="policy"/>
  <set>
  <expr>
    <attr name="DynamicQuota"/>
    <op value="="/>
    <value val="" isnull="y"/>
  </expr>
  </set>
  <where>
  <expr>
    <attr name="MSISDN"/>
    <op value="="/>
    <value val="33123654862"/>
  </expr>
  </where>
</req>
```

Display dynamic quota

```
<req name="select">
  <ent name="Subscriber" ns="policy"/>
  <select>
    <expr><attr name="DynamicQuota"/></expr>
  </select>
  <where>
    <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

  </where>
</req>
```

SPR subscriber provisioning (XML-REST)

These examples show how to provision the SPR subscriber using the XML-REST provisioning interface.

Prerequisite: The Hss service must be running on the blade and the HSS Server must be enabled (Hss Config) prior to being able to provision the SPR Subscriber.

Note: The keys to access a subscriber can be AccountId, IMSI, MSISDN, or NAI if defined. The provisioning samples use the MSISDN key.

Display SPR subscriber profile by MSISDN

```
{baseURI}/msr/sub/MSISDN/123456
METHOD: GET
BODY: None
```

Add SPR subscriber profile

```
BaseURI}/msr/sub
METHOD: PUT
BODY:
<?xml version="1.0" encoding="UTF-8"?>
<subscriber>
<field name="AccountID">10404723525</field>
<field name="MSISDN">+33123654862</field>
<field name="IMSI">184569547984229</field>
<field name="BillingDay">1</field>
<field name="Tier"></field>
<field name="Entitlement">DayPass</field>
</subscriber>
```

Add/Update State value by MSISDN

```
URL: /rs/msr/sub/MSISDN/123456/data/state
METHOD: PUT
BODY:<?xml version="1.0" encoding="UTF-8"?>
<subscriber>
<data name="state">
<![CDATA[
<state>
<version>1</version>
<property><name>mcc</name><value>315</value></property>
<property><name>expire</name><value>2010-02-09T11:20:32</value></property>
<property><name>approved</name><value>yes</value></property>
</state>
]]>
</data>
</subscriber>
```

Display quota (one row) by MSISDN and quota name

```
URL: /rs/msr/sub/MSISDN/123123/data/quota/Cle%203G-5Go (%20 stands for space
character)
METHOD: GET
BODY: None
```

Add quota by MSISDN and quota name

Note: If the creation request contains only a subset of Quota fields, unfilled fields are created with their default values.

```
URL: /rs/msr/sub/MSISDN/123456/data/Quota
METHOD: PUT
BODY
<subscriber><data name="quota"><![CDATA[
<?xml version="1.0" encoding="UTF-8"?>
<usage>
<version>1</version>
<quota name="AggregateLimit">
<cid>9223372036854775807</cid>
<time>3422</time>
<totalVolume>1000</totalVolume>
<inputVolume>980</inputVolume>
<outputVolume>20</outputVolume>
<serviceSpecific>12</serviceSpecific>
<nextResetTime>2011-04-22T00:00:00-05:00</nextResetTime>
<Type>Quota</Type>
<QuotaState>Expired</QuotaState>
<RefInstanceId>184569547984765</RefInstanceId >
</quota></usage>
]]></data></subscriber>
```

Update quota field by MSISDN and quota name

```
URL: /rs/msr/sub/MSISDN/123456/data/quota/Key%203G-5Go/inputvolume/10000
METHOD: PUT
BODY: None
```

Reset quota by MSISDN and quota name

```
URL: /rs/msr/sub/MSISDN/123456/data/quota/Cle%203G-5Go
METHOD: POST
BODY: None
```

Add/update dynamic quota (opaque)

```
URL: /rs/msr/sub/MSISDN/123456/data/DynamicQuota
METHOD: PUT
BODY:
<subscriber><data name="dynamicquota"><![CDATA[
<?xml version="1.0" encoding="UTF-8"?>
<usage>
<version>1</version>
<dynamicquota name="Key 203G-5Go">
 <Type>Roll-Over</Type>
 <InstanceId>15678</InstanceId>
 <Priority>4</Priority>
 <InitialTime>135</InitialTime >
 <InitialTotalVolume>2000</InitialTotalVolume>
 <InitialInputVolume>1500</InitialInputVolume>
 <InitialOutputVolume>500</InitialOutputVolume>
 <InitialServiceSpecific>4</InitialServiceSpecific>
 <ActivationDateTime>32</ActivationDateTime>
 <ExpirationDateTime>28</ExpirationDateTime>
 <InterimReportingInterval>100</InterimReportingInterval>
</dynamicquota></usage>
]]></data></subscriber>
```

SPR Subscriber Quota provisioning (XML)

These examples provision the SPR subscriber quota within the Quota entity. For quota provisioning in the subscriber profile, refer to SPR subscriber provisioning (XML).

The keys to access a subscriber can be AccountId, IMSI, MSISDN, or NAI if defined.

Prerequisite: The Hss service must be running on the blade and the HSS Server must be enabled (Hss Config) prior to being able to provision the SPR Subscriber.

Add quota by MSISDN and quota name

Both the Insert or Update request operation can add a quota. If the quota name does not exist yet for the subscriber profile, the operation creates it. If the request contains only a subset of Quota fields, the other fields are added and contain their default values.

```
<req name="insert">
 <ent name="QuotaEntity" ns="policy"/>
 <set>
  <expr><attr name="Name"/><value val="3G-5Go_Key"/></expr>
  <expr><attr name="Cid"/><value val="9223372036854999999"/></expr>
  <expr><attr name="Time"/><value val="10:10"/></expr>
  <expr><attr name="totalVolume"/><value val="7600"/></expr>
  <expr><attr name="inputVolume"/><value val="1900"/></expr>
  <expr><attr name="MSISDN"/><value val="380561234567"/></expr>
  <expr><attr name="outputVolume"/><value val="5700"/></expr>
  <expr><attr name="serviceSpecific"/><value val="8348"/></expr>
  <expr><attr name="nextResetTime"/><value val="2011-12-15T09:04:03"/></expr>
  <expr><attr name="Type"/><value val="Pass"/></expr>
  <expr><attr name="GrantedTotalVolume"/><value val="1000"/></expr>
  <expr><attr name="GrantedInputVolume"/><value val="500"/></expr>
  <expr><attr name="GrantedOutputVolume"/><value val="500"/></expr>
  <expr><attr name="GrantedTime"/><value val="200"/></expr>
  <expr><attr name="GrantedServiceSpecific"/><value val="1243"/></expr>
  <expr><attr name="QuotaState"/><value val="Expired"/></expr>
  <expr><attr name="RefInstanceId"/><value val="184569547984765"/></expr>
 </set>
</req>
```

```
<req name="update">
 <ent name="QuotaEntity" ns="policy"/>
 <set>
  <expr><attr name="Cid"/><value val="9223372036854775807"/></expr>
  <expr><attr name="Time"/><value val="3422"/></expr>
  <expr><attr name="totalVolume"/><value val="7600"/></expr>
  <expr><attr name="inputVolume"/><value val="1900"/></expr>
  <expr><attr name="outputVolume"/><value val="5700"/></expr>
  <expr><attr name="serviceSpecific"/><value val="9348"/></expr>
  <expr><attr name="nextResetTime"/><value
val="2011-05-01T00:00:00-05:00"/></expr>
  <expr><attr name="Type"/><value val="Pass"/></expr>
  <expr><attr name="GrantedTotalVolume"/><value val="1000"/></expr>
  <expr><attr name="GrantedInputVolume"/><value val="500"/></expr>
  <expr><attr name="GrantedOutputVolume"/><value val="500"/></expr>
  <expr><attr name="GrantedTime"/><value val="200"/></expr>
  <expr><attr name="GrantedServiceSpecific"/><value val="1243"/></expr>
  <expr><attr name="QuotaState"/><value val="Expired"/></expr>
  <expr><attr name="RefInstanceId"/><value val="184569547984765"/></expr>
 </set>
 <where>
  <expr><attr name="MSISDN"/><op value="="/><value val="380561234567"/></expr>
  <expr><attr name="Name"/><op value="="/><value val="New Quota"/></expr>
 </where>
</req>
```

Delete quota by MSISDN and quota name

```
<req name="delete">
 <ent name="QuotaEntity" ns="policy"/>
 <where>
  <expr><attr name="MSISDN"/><op value="="/><value val="184569547984229"/></expr>

  <expr><attr name="Name"/><op value="="/><value val="Cle 3G-5Go"/></expr>
 </where>
</req>
```

Select quota by MSISDN and quota name

```
<req name="select">
  <ent name="QuotaEntity" ns="policy"/>
  <select>
     <expr><attr name="Cid"/></expr>
     <expr><attr name="Time"/></expr>
     <expr><attr name="totalVolume"/></expr>
     <expr><attr name="inputVolume"/></expr>
     <expr><attr name="outputVolume"/></expr>
     <expr><attr name="serviceSpecific"/></expr>
     <expr><attr name="nextResetTime"/></expr>
  </select>
  <where>
   <expr><attr name="MSISDN"/><op value="="/><value val="380561234567"/></expr>

   <expr><attr name="Name"/><op value="="/><value val="3G-5Go_Key"/></expr>
  </where>
</req>
```

Update quota by MSISDN and quota name

```
<req name="update">
<ent name="QuotaEntity" ns="policy"/>
<set>
  <expr><attr name="Time"/><value val="11:05"/></expr>
  <expr><attr name="totalVolume"/><value val="14000"/></expr>
</set>
<where>
  <expr><attr name="MSISDN"/><op value="="/><value val="380561234567"/></expr>
  <expr><attr name="Name"/><op value="="/><value val="3G-5Go_Key"/></expr>
 </where>
</req>
```

Reset quota by MSISDN and quota name.

```
<req name="operation">
<oper name="ResetQuota" ent="Subscription" ns="global">
  <expr><param name="MSISDN"/><op value="="/><value val="380561234567"/></expr>

  <expr><param name="Name" /><"/><op value="="/><value val="NEW QUOTA"/></expr>
</oper>
</req>
```

Display a particular Quota entity by name

This request will return quota instances that match the specified name and have the field type set to the specified value. In this example the value is quota. Any other quota instances sharing that name but with different type values will not be returned.

If multiple quota instances exist for a subscriber with the same type then each will be included in the response.

```
<req name="select">
 <ent name="QuotaEntity" ns="policy"/>
 <select>
  <expr><attr name="CId"/></expr>
 <expr><attr name="TotalVolume"/></expr>
 <expr><attr name="InputVolume"/></expr>
<select>
 </where>
  <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>
  <expr><attr name="Name"/><op value="="/><value val="Q1"/></expr>
  <expr><attr name="Type"/><op value="="/><value val="quota"/></expr>
 </where>
</req>
```

Display a particular Quota instance for passes and top-ups

The CId is the unique identifier that can be used to select a specific instance.

Note: In the case of passes and top-ups, the CId is set to be the same as the InstanceID of the associated dynamic quota.

```
<req name="select">
 <ent name="QuotaEntity" ns="policy"/>
 <select>
  <expr><attr name="CId"/></expr>
  <expr><attr name="TotalVolume"/></expr>
  <expr><attr name="InputVolume"/></expr>
 <select>
  </where>
   <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>
   <expr><attr name="Name"/><op value="="/><value val="Q1"/></expr>
   <expr><attr name="CId"/><op value="="/><value val="12345"/></expr>
  </where>
</req>
```

Update a Quota instance

```
<req name="update">
 <ent name="QuotaEntity" ns="policy"/>
 <set>
  <expr><attr name="Time"/><value val="3422"/></expr>
  <expr><attr name="TotalVolume"/><value val="7600"/></expr>
  <expr><attr name="InputVolume"/><value val="1900"/></expr>
  <expr><attr name="OutputVolume"/><value val="5700"/></expr>
  <expr><attr name="ServiceSpecific"/><value val="9348"/></expr>
  <expr><attr name="NextResetTime"/><value val="2013-02-09T00:05:00"/></expr>
 <set>
  </where>
   <expr><attr name="MSISDN"/><value val="+33123654862"/></expr>
   <expr><attr name="Name"/><value val="Q1"/></expr>
  </where>
</req>
```

This update request will only update quota instances that match the specified name and have quota type set to "quota". Any other instances with the same name but *different* quota type will remain unchanged.

Insert a new Quota instance

This occurs when the provisioning system creates a quota instance associated with a plan.

Note: In the case of passes and top-ups, the CId is set to be the same as the InstanceID of the associated dynamic quota.

```
<req name="insert">
 <ent name="QuotaEntity" ns="policy"/>
 <set>
  <expr><attr name="MSISDN"/><value val="+33123654862"/></expr>
  <expr><attr name="Name"/><value val="Q1"/></expr>
  <expr><attr name="Time"/><value val="10:10"/></expr>
  <expr><attr name="Type"/><value val="quota"/></expr>
  <expr><attr name="TotalVolume"/><value val="55000"/></expr>
  <expr><attr name="InputVolume"/><value val="50000"/></expr>
  <expr><attr name="OutputVolume"/><value val="5000"/></expr>
  <expr><attr name="ServiceSpecific"/><value val="ServiceSpecific"/></expr>
  <expr><attr name="NextResetTime"/><value val="2013-02-09T00:05:00"/></expr>
 <set>
</req>
```

Delete a specific Quota instance by name

This shall return only the quota instance associated with the plan, which is identified by the field “type=quota”.

```
<req name="delete">
   <ent name="QuotaEntity" ns="policy"/>
   </where>
    <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

    <expr><attr name="Name"/><op value="="/><value val="Q1"/></expr>
    <expr><attr name="Type"/><op value="="/><value val="quota"/></expr>
   </where>
 </req>
```

Delete a specific Quota instance for passes and top-ups

This shall return only the quota instance associated with the plan, which is identified by the field “type=quota”.

Note: In the case of passes and top-ups, the CId is set to be the same as the InstanceID of the associated dynamic quota.

```
<req name="delete">
   <ent name="QuotaEntity" ns="policy"/>
   </where>
    <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

    <expr><attr name="Name"/><op value="="/><value val="Q1"/></expr>
    <expr><attr name="CId"/><op value="="/><value val="12345"/></expr>
   </where>
 </req>
```

Reset Quota to reset the base quota associated with the subscriber’s plan

This request will only reset instances that match the specified name and have the field “type” set to the specified value, quota in this example. Any other quota instances sharing that name but with different “type” value willl remain unchanged. If multiple instances exist with the specified name and “type” value, then an error will be returned.

If the subscriber profile does not contain any instances with the specified name “type” value, then an error will be returned.

```
<req name="operation">
 <oper name="ResetQuota" ent=”Subscription” ns="global"/>
   <expr><param name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

   <expr>< param name="Name"/><op value="="/><value val="Q1"/></expr>
   <expr><param name="Type"/><op value="="/><value val="quota"/></expr>
  </oper>
</req>
```

Reset Quota to reset the quota usage associated with passes and top-ups

The CId is the unique identifier that can be used to select a specific instance.

Note: In the case of passes and top-ups, the CId is set to be the same as the InstanceID of the associated dynamic quota.

```
<req name="operation">
 <oper name="ResetQuota" ent=”Subscription” ns="global"/>
   <expr><param name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

   <expr>< param name="Name"/><op value="="/><value val="Q1"/></expr>
   <expr><param name="Cid"/><op value="="/><value val="123456"/></expr>
 </oper>
</req>
```

In the event that multiple quota instances exist with the specified name, then there should only be one instance with the specified CId value. Only the data in this instance will be updated. Any other instances will be skipped.

In the event that multiple quota instances exist with the specified name and the specified CId value, then an error will be generated.

SPR Subscriber Quota Provisioning (XML-REST)

These examples provision the subscriber quota using the XML-REST interface.

Prerequisite: The Hss service must be running on the blade and the HSS Server must be enabled (Hss Config) prior to being able to provision the SPR Subscriber.

Add a subscriber quota

This request adds a quota to the subscriber profile using the subscriber MSISDN.

Note: An Add Quota request may contain only a subset of quota fields. In that case, the unfilled fields are created in the Quota with their default values.

```
{baseURI}/msr/sub/MSISDN/123456/data/quota
METHOD: PUT
BODY:
<?xml version="1.0" encoding="UTF-8"?>
<usage>
<version>1</version>
<quota name="AggregateLimit">
 <cid>9223372036854775807</cid>
 <time>3422</time>
 <totalVolume>1000</totalVolume>
 <inputVolume>980</inputVolume>
 <outputVolume>20</outputVolume>
 <serviceSpecific>12</serviceSpecific>
 <nextResetTime>2011-04-22T00:00:00-05:00</nextResetTime>
 <Type>Pass</Type>
 <State>Expired</State>
 <RefInstanceId>184569547984765</RefInstanceId >
</quota></usage>
```

Display a subscriber quota

This request displays a quota using the subscriber MSISDN and the quota name.

```
{baseURI}/msr/sub/MSISDN/123456/data/quota/Key%203G-5Go
METHOD: GET
BODY: None
```

Modify a subscriber quota

This request modifies a quota using the subscriber MSISDN and the quota name.

```
{baseURI}/msr/sub/MSISDN/123456/data/quota/Key%203G-5Go
METHOD: PUT
BODY:
<?xml version="1.0" encoding="UTF-8"?>
<usage>
<version>1</version>
<quota name="Key 203G-5Go">
 <cid>9223372036854775807</cid>
 <time>3422</time>
 <totalVolume>1000</totalVolume>
 <inputVolume>980</inputVolume>
 <outputVolume>20</outputVolume>
 <serviceSpecific>12</serviceSpecific>
 <nextResetTime>2011-04-22T00:00:00-05:00</nextResetTime>
 <Type>Pass</Type>
 <State>Expired</State>
 <RefInstanceId>184569547984765</RefInstanceId >
</quota></usage>
```

Modify the inputVolume field of a subscriber quota

This request modifies the InputVolume field of the quota using the subscriber MSISDN and the quota name.

```
{baseURI}/msr/sub/MSISDN/123456/data/quota/Key%203G-5Go/inputvolume/10000
METHOD: PUT
BODY: None
```

Reset a subscriber quota

This request resets a quota to the default value using the subscriber MSISDN and the quota name.

```
{baseURI}/msr/sub/MSISDN/123456/data/quota/Key%203G-5Go
METHOD: POST
BODY: None
```

Delete a subscriber quota

This request deletes a quota from the subscriber profile using the subscriber MSISDN and the quota name.

```
{baseURI}/msr/sub/MSISDN/123456/data/quota/Key%203G-5Go
METHOD: DELETE
BODY: None
```

Display a particular Quota entity

This request will return all attributes of every instance that matches the Quota name outlined in the request.

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/quota/{quota name}/row/type/{quota
type}/
METHOD: GET
BODY: None
```

Display a field from a specific Quota instance by name

If multiple DynamicQuota instances exist for a subscriber with the same InstanceId then each one will be included in the response.

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/quota/{quota name}/{field name}
METHOD: GET
BODY: None
```

Delete a specific Quota instance by name

If multiple Quota instances exist for the subscriber with the specified name and type=quota, then each of them will be deleted.

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/quota/{quota name}
METHOD: DELETE
BODY: None
```

Delete a specific Quota instance by name of a specific type

If multiple Quota instances exist for the subscriber with the same name and type then all of them will be deleted.

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/quota/{quotaname}/row/type/{quota type}
METHOD: DELETE
BODY: None
```

Update a field within a quota instance by name for a specific type

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/quota/{quotaname}/row/type/{quota
type}/{field value}
METHOD: PUT
BODY: None
```

Update a field within a quota instance for passes and top-ups

The CId is the unique identifier that can be used to select a specific instance.

Note: In the case of passes and top-ups, the CId is set to be the same as the InstanceID of the associated dynamic quota.

```
{baseURI}/msr/sub/MSISDN/{key value}/data/quota/{quota name}/row/cid/{cid
value}/[field name]/{FieldValue}
METHOD: PUT
BODY: None
```

Reset the base quota associated with a subscriber's plan

```
{baseURI}/msr/sub/MSISDN/{key value}/data/quota/{quota name}
METHOD: POST
BODY: None
```

Reset the quota usage based on name and type

```
{baseURI}/msr/sub/MSISDN/{key value}/data/quota/{quota name}/row/type/{quota
type}
METHOD: POST
BODY: None
```

Reset the quota usage associated with passes and top-ups

```
{baseURI}/msr/sub/MSISDN/{key value}/data/quota/{quota name}/row/cid/{cid value}
METHOD: POST
BODY: None
```

SPR subscriber DynamicQuota provisioning (XML)

These examples provision the SPR subscriber DynamicQuota.

Prerequisite: The HSS service must be running on the blade and the HSS Server must be enabled (HssConfig) prior to provisioning the SPR subscriber.

The following keys can be used to access a subscriber: AccountId, IMSI, MSISDN, or NAI. The examples use MSISDN.

Any field (attribute) specified as a part of the Global Schema for DynamicQuota can be used in a request. These attributes include: Name, InstanceId, Type, Priority, InitialTime, InitialTotalVolume, InitialInput Volume, InitialOutputVolume, InitialServiceSpecific, ActivationDateTime, ExpirationDateTime, PurchaseDateTime , Duration, and InterimReportingInterval. Only provisioned attributes will be returned in the response.

Display all DynamicQuota instances by MSISDN and DynamicQuota name

```
<req name="select">
  <ent name="DynamicQuotaEntity" ns="policy"/>
  <select>
    <expr><attr name="InstanceId"/></expr>
    <expr><attr name="InitialTotalVolume"/></expr>
    <expr><attr name="InitialInputVolume"/></expr>
  </select>
  <where>
   <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>
   <expr><attr name="Name"/><op value="="/><value val="DQ1"/></expr>
  </where>
</req>
```

This request returns all attributes of every instance that matches the DynamicQuota name outlined in the request.

Note: InstanceId should always be included in the request so that each instance returned in the result can be distinguished from the others.

Display DynamicQuota instance by MSISDN and InstanceId

```
<req name="select">
  <ent name="DynamicQuotaEntity" ns="policy"/>
  <select>
    <expr><attr name="InstanceId"/></expr>
    <expr><attr name="InitialTotalVolume"/></expr>
    <expr><attr name="InitialInputVolume"/></expr>
   </select>
   <where>
    <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

    <expr><attr name="Name"/><op value="="/><value val="DQ1"/></expr>
    <expr><attr name="InstanceId"/><op value="="/><value val="12345"/></expr>
   </where>
</req>
```

This request returns all DynamicQuota instances with the same InstanceId.

Delete DynamicQuota instance by MSISDN and InstanceId

```
<req name="delete">
  <ent name="DynamicQuotaEntity" ns="policy"/>
  <where>
    <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

    <expr><attr name="Name"/><op value="="/><value val="DQ1"/></expr>
    <expr><attr name="InstanceId"/><op value="="/><value val="12345"/></expr>
  </where>
</req>
```

This request deletes all DynamicQuota instances with the same InstanceId. If the InstanceId does not exist, an error will be returned.

Add a new DynamicQuota instance

```
<req name="insert">
   <ent name="DynamicQuotaEntity" ns="policy"/>
   <set>
    <expr><attr name="MSISDN"/><value val="33123654862"/></expr>
    <expr><attr name="Name"/><value val="DQ1"/></expr>
    <expr><attr name="Type"/><value val="Roll-Over"/></expr>
    <expr><attr name="InstanceId"/><value val="15678"/></expr>
    <expr><attr name="Priority"/><value val="4"/></expr>
    <expr><attr name="InitialTime"/><value val="135"/></expr>
    <expr><attr name="InitialTotalVolume"/><value val="2000"/></expr>
    <expr><attr name="InitialInputVolume"/><value val="1500"/></expr>
    <expr><attr name="InitialOutputVolume"/><value val="500"/></expr>
    <expr><attr name="InitialServiceSpecific"/><value val="4"/></expr>
    <expr><attr name="ActivationDateTime"/><value
                     val="2013-02-09T00:00:00"/></expr>
    <expr><attr name="ExpirationDateTime"/><value
```

```
                     val="2013-02-15T00:00:00"/></expr>
    <expr><attr name="InterimReportingInterval"/><value val="100"/></expr>
    <expr><attr name="Duration"/><value val="10"/></expr>
   </set>
</req>
```

This request adds a new DynamicQuota instance using one of the subscriber keys and any DynamicQuota attributes.

Note: If the request includes attributes that have not been defined in the global schema, then the request will be rejected and an error message sent.

SPR subscriber DynamicQuota provisioning (XML-REST)

These examples provision the SPR subscriber DynamicQuota.

Prerequisite: The HSS service must be running on the blade and the HSS Server must be enabled (HssConfig) prior to provisioning the SPR subscriber.

The following keys can be used to access a subscriber: AccountId, IMSI, MSISDN, or NAI. The examples use MSISDN.

Any field (attribute) specified as a part of the Global Schema for DynamicQuota can be used in a request. These attributes include: Name, InstanceId, Type, Priority, InitialTime, InitialTotalVolume, InitialInput Volume, InitialOutputVolume, InitialServiceSpecific, ActivationDateTime, ExpirationDateTime, PurchaseDateTime , Duration, and InterimReportingInterval. Only provisioned attributes will be returned in the response.

Display all DynamicQuota instances by MSISDN and DynamicQuota name

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/dynamicquota/{dynamicquota name}
METHOD: GET
BODY: None
```

This request returns all attributes of every instance that matches the DynamicQuota name outlined in the request.

Display DynamicQuota by MSISDN and InstanceId

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/dynamicquota/{dynamicquota
name}/InstanceId/{InstanceId value}
METHOD: GET
BODY: None
```

This request returns all DynamicQuota instances with the same InstanceId. If multiple DynamicQuota instances exist for a subscriber with the same InstanceId then each one will be included in the response.

Delete DynamicQuota instance by MSISDN and InstanceId

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/dynamicquota/{dynamicquota
name}/InstanceId/{InstanceId value}
METHOD: DELETE
BODY: None
```

This request deletes all DynamicQuota instances with the same InstanceId. If the InstanceId does not exist, an error will be returned.

Add (insert) a new DynamicQuota instance

```
{baseURI}/msr/sub/MSISDN/{key value}/data/dynamicquota/{DynamicQuota name}
METHOD: PUT
BODY:
<?xml version="1.0" encoding="UTF-8"?>
<subscriber>
<data name="dynamicquota">
<![CDATA[
<?xml version="1.0" encoding="UTF-8"?>
<definition>
<version>1</version>
<dynamicquota name="AggregateLimit">
<instanceid>54321</instanceid>
<type>Roll-Over</type>
<initialtotalVolume>1000</initialtotalVolume>
<initialinputVolume>980</initialinputVolume>
<outputVolume>20</outputVolume>
<InitialserviceSpecific>12</InitialserviceSpecific>
<ActivationDateTime>2011-04-22T00:00:00-05:00</ActivationDateTime>
</dynamicquota>
</definition>
]]>
</data>
</subscriber>
```

This request adds a new DynamicQuota instance using one of the subscriber keys and any DynamicQuota attributes.

Note: No check is made to ensure the uniqueness of the InstanceId in this request against other instances that may already exist. This request will be inserted even if other instances already exist with the same InstanceId. It is the responsibility of the provisioning system to ensure uniqueness of the InstanceId in each DynamicQuota instance.

SPR subscriber State provisioning (XML)

This section outlines the provisioning requests when updating properties within a subscriber's State entity. The state entity is comprised of one or more properties and each property is comprised of a name/value pair.

Prerequisite: The HSS service must be running on the blade and the HSS Server must be enabled (HssConfig) prior to provisioning the SPR subscriber.

The following keys can be used to access a subscriber: AccountId, IMSI, MSISDN, or NAI.

Display a State property

```
<req name="select">
   <ent name="State" ns="policy"/>
   <select>
      <expr><attr name="S1"/></expr>
   </select>
   <where>
    <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

   </where>
</req>
```

This request returns the value associated with the property name (S1). Multiple state properties can be included in a single Select request. Include an additional <expr> clause within the <select> clause for each additional property.

Delete a State property

```
<req name="update">
   <ent name="StateEntity" ns="policy"/>
   <set>
     <expr><attr name="S1"/>"/><op value="="/><value val="" isnull="y"/></expr>

   </set>
   <where>
    <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

   </where>
</req>
```

This request deletes a State property. If the property does not exist, an error will be returned.

Only one instance of a state property with a specific name is valid in the subscriber record. If multiple state properties exist in the subscriber record with the same name, then one of them will be updated with the value in this request. The remaining properties will be deleted as a part of processing this request.

Insert a State property value

If the value of the State property is S1, then change it to 315 using the following request.

```
<req name="insert">
     <ent name="StateEntity" ns="policy"/>
     <set>
      <expr><attr name="MSISDN"/><value val="33123654862"/></expr>
      <expr><attr name="S1"/><value val="315"/></expr>
     </set>
</req>
```

This request adds a value to a State property. If the property does not exist, an error will be returned.

Only one instance of a state property with a specific name is valid in the subscriber record. If multiple state properties exist in the subscriber record with the same name, then one of them will be updated with the value in this request. The remaining properties will be deleted as part of processing this request.

Update a State property value

```
<req name="update">
   <ent name="StateEntity" ns="policy"/>
   <set>
     <expr><attr name="S1"/><value val="315"/></expr>
   </set>
   <where>
    <expr><attr name="MSISDN"/><op value="="/><value val="33123654862"/></expr>

   </where>
</req>
```

This request adds a value (315) to a State property (S1). If the property does not exist, an error will be returned.

If the specified State entity does not exist, then an error is returned.

Only one instance of a state property with a specific name is valid in the subscriber record. If multiple state properties exist in the subscriber record with the same name, then one of them will be updated with the value in this request and the remaining ones will be deleted as part of processing this request.

SPR subscriber State provisioning (XML-REST)

These examples provision the SPR subscriber State.

Prerequisite: The HSS service must be running on the blade and the HSS Server must be enabled (HssConfig) prior to provisioning the SPR subscriber.

The following keys can be used to access a subscriber: AccountId, IMSI, MSISDN, or NAI. The examples use MSISDN.

Display a specific state property

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/state/{property name}
METHOD: GET
BODY: None
```

If multiple state properties exist with the same name then each will be displayed in a separate <property> row within the <state> clause in the response.

Delete a specific state property

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/state/{property name}
METHOD: DELETE
BODY: None
```

Only one instance of a state property with a specific name is valid in the subscriber record. If multiple state properties exist in the subscriber record with the same name, then they will all be deleted as part of processing this request.

Update the value associated with a specific state property

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/state/{property name}/{property value}
METHOD: PUT
BODY: None
```

If the specified state does not exist then an error will be returned.

Add (insert) a new state property

```
{baseURI}/msr/sub/MSISDN/{keyvalue}/data/state/{property name}/{property value}
METHOD: PUT
BODY: None
```

If the name already exists in the SPR database then an error is returned.

SPR Pool provisioning (XML)

These examples provision the SPR Pool using the XML interface.

Select a Pool by MSISDN

```
<req name="select">
 <ent name="Pool" ns="policy"/>
 <where>
  <expr><attr name="MSISDN"/><op value="="/><value val="380561234567"/></expr>
 </where>
</req>
```

Select a pool data by PoolID

```
<req name="select">
  <ent name="Pool" ns="policy"/>
  <select>
     <expr><attr name="PoolID"/></expr>
     <expr><attr name="Entitlement"/></expr>
     <expr><attr name="BillingDay"/></expr>
  </select>
  <where>
    <expr><attr name="PoolID"/><op value="="/><value val="1000"/></expr>
  </where>
</req>
```

Select a pool quota or pool state by PoolID

```
<req name="select">
  <ent name="Pool" ns="policy"/>
  <select>
    <expr><attr name="PoolQuota"/></expr>
  </select>
  <where>
    <expr><attr name="PoolID"/><op value="="/><value val="1000"/></expr>
  </where>
</req>
```

For a pool state request, replace name="PoolQuota" with name="PoolState".

Add a subscriber to a pool by PoolID and MSISDN

```
<req name="operation">
  <oper name="AddPoolMember" ent="Subscription" ns="global">
   <expr><param name="PoolID" /><op value="="/><value val="1000"/></expr>
   <expr><param name="MSISDN"/><op value="="/><value val="380561234567"/></expr>

  </oper>
</req>
```

Add a pool data by PoolID

```
<req name="insert">
 <ent name="Pool" ns="policy"/>
 <set>
  <expr><attr name="PoolID"/><op value="="/><value val="1000"/></expr>
  <expr><attr name="BillingDay"/><value val="8"/></expr>
  <expr><attr name="Entitlement"/><value val="DayPass,DayPassPlus"/></expr>
 </set>
</req>
```

Add/update a pool quota by PoolID

```
<req name="update">
  <ent name="Pool" ns="policy"/>
  <set>
    <expr><attr name="PoolQuota"/><op value="="/><cdata>
      <![CDATA[<?xml version="1.0" encoding="UTF-8"?>
      <usage>
        <version>1</version>
        <name>3G-5Go_Key</name>
        <PoolQuota name="AggregateLimit">
        <cid>9223372036854999999</cid>
        <time>3422</time>
        <totalvolume>514</totalvolume>
        <inputvolume>998</inputvolume>
        <outputvolume>2722</outputvolume>
        <servicespecific>8348</serviceSpecific>
        <nextresettime>2011-12-15T09:04:03</nextresettime>
        <type>pass</type>
        <grantedtime>200</grantedtime>
        <grantedservicespecific>1234</grantedservicespecific>
        <quotastate>Expired</quotastate>
        <refinstanceid>184569547984765</refinstanceid>
        </PoolQuota>
        </usage>
      ]]></cdata></expr>
    </set>
  <where>
    <expr><attr name="PoolID"/><op value="="/><value val="1000"/></expr>
  </where>
</req>
```

Add/update a pool dynamic quota by PoolID

```
<req name="update">
  <ent name="Pool" ns="policy"/>
  <set>
    <expr><attr name="PoolDynamicQuota"/><op value="="/><cdata>
      <![CDATA[<?xml version="1.0" encoding="UTF-8"?>
      <usage>
        <version>1</version>
        <name>3G-5Go_Key</name>
        <PoolDynamicQuota name="AggregateLimit">
        <Type>Roll-over</Type>
        <InstanceId>15678</InstanceId>
        <Priority>4</Priority>
        <InitialTime>135</InitialTime >
        <InitialTotalVolume>2000</InitialTotalVolume>
        <InitialInputVolume>1500</InitialInputVolume>

<InitialOutputVolume>500</InitialOutputVolume><InitialTime>3422</InitialTime>
        <InitialServiceSpecific>4</InitialServiceSpecific>
        <ActivationDateTime>32</ActivationDateTime>
        <ExpirationDateTime>28</ExpirationDateTime>
        <PurchaseDateTime>28</PurchaseDateTime>
        <Duration>28</Duration>
        <InterimReportingInterval>100</InterimReportingInterval>
        </PoolDynamicQuota>
        </usage>
      ]]></cdata></expr>
    </set>
  <where>
    <expr><attr name="PoolID"/><op value="="/><value val="1000"/></expr>
  </where>
</req>
```

Delete a pool data by PoolID

```
<<req name="delete">
<ent name="Pool" ns="policy"/>
<where>
  <expr><attr name="PoolID"/><op value="="/><value val="1000"/></expr>
</where>
</req>
```

Delete a pool quota, pool dynamic quota, or pool state by PoolID

```
<req name="update">
  <ent name="Pool" ns="policy"/>
  <set>
    <expr>
      <attr name="PoolQuota"/>
      <op value="="/>
      <value val="" isnull="y"/>
    </expr>
  </set>
  <where>
    <expr>
      <attr name="PoolID"/>
      <op value="="/>
      <value val="1000 "/>
    </expr>
  </where>
</req>
```

For a pool dynamic quota request, replace name="PoolQuota" with name="PoolDynamicQuota".

For a pool state request, replace name="PoolQuota" with name="PoolState".

Delete a subscriber from a pool by PoolID and MSISDN

```
<req name="operation">
  <oper name="DelPoolMember" ent="Subscription" ns="global">
   <expr><param name="PoolID" /><op value="="/><value val="1000"/></expr>
   <expr><param name="MSISDN"/><op value="="/><value val="380561234567"/></expr>

  </oper>
</req>
```

Get list of pool subscriber members by PoolID

```
<req name="operation">
  <oper name="GetPoolMembers" ent="Subscription" ns="global">
   <expr><param name="PoolID" /><op value="="/><value val="1000"/></expr>
  </oper>
</req>
```

The result of this request will be one row per subscriber in the pool. Each row will contains the subscriber's public identity followed by the value of its four keys in this order: MSISDN, IMSI, NAI, AccountId.

Get PoolID of subscriber by MSISDN

```
<req name="operation">
  <oper name="GetPoolID" ent="Subscription" ns="global">
   <expr><param name="<MSISDN"/><op value="="/><value val="380561234567"/></expr>

  </oper>
</req>
```

SPR Pool provisioning (XML-REST)

These examples show how to provision the Pool profile using the XML-REST interface.

Select a pool data by PoolID

```
{baseURI}/msr/pool/PoolID/1000
METHOD: GET
BODY: None
```

Add a pool data by PoolID

```

{baseURI}/msr/pool
METHOD: POST
BODY:
<?xml version="1.0" encoding="UTF-8"?>
<usage>
<version>8</version>
<pool>
 <field name="PoolID">1000</field>
 <field name="BillingDay">5</field>
 <field name="Tier">12</field>
 <field name="Entitlement">Weekpass</field>
 <field name="Entitlement">Daypass</field>
 <field name="Custom15">allo</field>
</pool></usage>
```

Delete a pool data by PoolID

```
{baseURI}/msr/pool/PoolID/1000
METHOD: DELETE
BODY: None
```

Add pool (opaque)

```
URL: /rs/msr/pool/1000
METHOD: POST
BODY: None
```

Add/update a pool quota (opaque)

```
URL: /rs/msr/pool/1000/data/poolquota/
METHOD: PUT
BODY:
<?xml version="1.0" encoding="UTF-8"?>
<pool>
<data name="poolquota">
<![CDATA[
<?xml version="1.0" encoding="UTF-8"?>
<usage>
<version>1</version>
<quota name="Key 203G-5Go">
<cid>9223372036854775807</cid>
<time>3422</time>
<totalVolume>1000</totalVolume>
<inputVolume>980</inputVolume>
<outputVolume>20</outputVolume>
<serviceSpecific>12</serviceSpecific>
<nextResetTime>2011-04-22T00:00:00-05:00</nextResetTime>
<Type>Quota</Type>
<QuotaState>Expired</QuotaState>
<RefInstanceId>184569547984765</RefInstanceId>
</quota>
</usage>
]]
</data>
</pool>
```

Add/update a pool dynamic quota (opaque)

```
URL: /rs/msr/pool/1000/data/PoolDynamicQuota
METHOD: PUT
BODY:
<pool><data name="PoolDynamicQuota">
<![CDATA[
<?xml version="1.0" encoding="UTF-8"?>
<definition>
<version>1</version>
<dynamicquota name="AggregateLimit">
<Type>Roll-Over</Type>
<InstanceId>15678</InstanceId>
<Priority>4</Priority>
<InitialTime>135</InitialTime >
<InitialTotalVolume>2000</InitialTotalVolume>
<InitialInputVolume>1500</InitialInputVolume>
<InitialOutputVolume>500</InitialOutputVolume>
<InitialServiceSpecific>4</InitialServiceSpecific>
<ActivationDateTime>32</ActivationDateTime>
<ExpirationDateTime>28</ExpirationDateTime>
<PurchaseDateTime>28</PurchaseDateTime>
<Duration>28</Duration>
<InterimReportingInterval>100</InterimReportingInterval>
</dynamicquota>
</definition>
]]>
</data>
</pool>
```

Update pool state (opaque)

```
URL: /rs/msr/pool/1000/data/poolstate/
METHOD: PUT
BODY:
<?xml version="1.0" encoding="UTF-8"?>
<pool>
<data name="poolstate">
<![CDATA[
<state>
<version>1</version>
<property><name>mcc</name><value>315</value></property>
<property><name>expire</name><value>2010-02-09T11:20:32</value></property>
<property><name>approved</name><value>yes</value></property>
</state>
]]>
</data>
</pool>
```

Get list of subscribers members of a Pool by PoolID

```
{baseURI}/msr/pool/PoolID/1000/member
METHOD: GET
BODY: None
```

Add a Subscriber to a Pool by PoolID and PublicIdentity

```
{baseURI}/msr/pool/PoolID/1000/member/PublicIdentity/tel:+380561234567
METHOD: POST
BODY: None
```

Delete Subscriber from a Pool by PoolID and PublicIdentity

```
{baseURI}/msr/pool/PoolID/1000/member/PublicIdentity/tel:+380561234567
METHOD: DELETE
BODY: None
```

Glossary

A	
AAA	Authentication, Authorization, and Accounting (Rx Diameter command)
AuC	Authentication Center
C	
CLI	Command-line interface
E	
ENUM	T**E**lephone **NU**mber **M**apping - A technology for unifying various communications and telephone addresses for private and business numbers, facsimile and mobile phone numbers, SMS services, Instant Messaging and email. ENUM integrates legacy phone numbers with the Domain Name System (DNS). Users can access and maintain a directory that supports all forms of wired communication, mobile communications networks, and the Internet. ENUM allows for an end user to be reached on multiple devices via one phone number and allows the end user to determine which device to contact first or multiple devices simultaneously. E.164 Number Mapping
ESD	Electro-Static Discharge
H	
HLR	Home Location Register

H

	A component within the Switching Subsystem of a GSM network. The HLR database is the central database within the GSM architecture. This is where information about the mobile communications subscribers who are assigned to a specific location area is stored. The subscriber data is used to establish connections and control services. Depending on the network size, the number of subscribers and the network organization, a number of HLRs can exist within a GSM network.
HSS	Home Subscriber Server A central database for subscriber information.

I

IMPI	IP Multimedia Private Identity
IMPU	IP Multimedia Public Identity

M

MSISDN	Mobile Station International Subscriber Directory Number The MSISDN is the network specific subscriber number of a mobile communications subscriber. This is normally the phone number that is used to reach the subscriber. Mobile Subscriber Integrated Services Digital Network [Number] Mobile Station International Subscriber Directory Number. The unique, network-specific subscriber number of a mobile communications subscriber. MSISDN follows the E.164 numbering plan; that is, normally

M

	the MSISDN is the phone number that is used to reach the subscriber.

O

OAM&P	Operations, Administration, Maintenance, and Provisioning. These functions are generally managed by individual applications and not managed by a platform management application, such as PM&C Operations – Monitoring the environment, detecting and determining faults, and alerting administrators. Administration – Typically involves collecting performance statistics, accounting data for the purpose of billing, capacity planning, using usage data, and maintaining system reliability. Maintenance – Provides such functions as upgrades, fixes, new feature enablement, backup and restore tasks, and monitoring media health (for example, diagnostics). Provisioning – Setting up user accounts, devices, and services.

P

PDN	Packet Data Network A digital network technology that divides a message into packets for transmission.

S

SC	System Controller
SDM	Subscriber Data Management

S

SIM	Subscriber Identity Module An ID card the size of a credit card for GSM network subscribers, and is typically referred to as a chip card or smartcard.
SIP	Session Initiation Protocol A peer-to-peer protocol used for voice and video communications.
SLF	Subscription Locator Function
SOAP	Simple Object Access Protocol
SPR	Subscriber Profile Repository A logical entity that may be a standalone database or integrated into an existing subscriber database such as a Home Subscriber Server (HSS). It includes information such as entitlements, rate plans, etc. The PCRF and SPR functionality is provided through an ecosystem of partnerships.
SS7	Signaling System #7 A communications protocol that allows signaling points in a network to send messages to each other so that voice and data connections can be set up between these signaling points. These messages are sent over its own network and not over the revenue producing voice and data paths. The EAGLE is an STP, which is a device that routes these messages through the network.

S	
SSH	Secure Shell A protocol for secure remote login and other network services over an insecure network. SSH encrypts and authenticates all EAGLE IPUI and MCP traffic, incoming and outgoing (including passwords) to effectively eliminate eavesdropping, connection hijacking, and other network-level attacks.
T	
TCP	Transfer-Cluster-Prohibited Transfer Control Protocol Transmission Control Protocol A connection-oriented protocol used by applications on networked hosts to connect to one another and to exchange streams of data in a reliable and in-order manner.
U	
USM	User Security Management
W	
WebCI	Web Craft Interface
X	
XML	eXtensible Markup Language A version of the Standard Generalized Markup Language (SGML) that allows Web developers to create customized tags for additional functionality.