Avaya Aura® System Manager

User management web services programmer guide

Release
10.1.x
Issue 1.0
Last Modified Date: 14th December 2021

Notice

While reasonable efforts were made to ensure that the information in this document was complete and accurate at the time of printing, Avaya Inc. can assume no liability for any errors. Changes and corrections to the information in this document might be incorporated in future releases.

Documentation disclaimer

Avaya Inc. is not responsible for any modifications, additions, or deletions to the original published version of this documentation unless such modifications, additions, or deletions were performed by Avaya. Customer and/or End User agree to indemnify and hold harmless Avaya, Avaya's agents, servants and employees against all claims, lawsuits, demands and judgments arising out of, or in connection with, subsequent modifications, additions or deletions to this documentation to the extent made by the Customer or End User.

Link disclaimer

Avaya Inc. is not responsible for the contents or reliability of any linked Web sites referenced elsewhere within this documentation, and Avaya does not necessarily endorse the products, services, or information described or offered within them. We cannot guarantee that these links will work all the time and we have no control over the availability of the linked pages.

Warranty

Avaya Inc. provides a limited warranty on this product. Refer to your sales agreement to establish the terms of the limited warranty. In addition, Avaya's standard warranty language, as well as information regarding support for this product, while under warranty, is available through the Avaya Support Web site: http://support.avaya.com

License

USE OR INSTALLATION OF THE PRODUCT INDICATES THE END USER'S ACCEPTANCE OF THE TERMS SET FORTH HEREIN AND THE GENERAL LICENSE TERMS AVAILABLE ON THE AVAYA WEB SITE

http://support.avaya.com/LicenseInfo/ ("GENERAL LICENSE TERMS"). IF YOU DO NOT WISH TO BE BOUND BY THESE TERMS, YOU MUST RETURN THE PRODUCT(S) TO THE POINT OF PURCHASE WITHIN TEN (10) DAYS OF DELIVERY FOR A REFUND OR CREDIT.

Avaya grants End User a license within the scope of the license types described below. The applicable number of licenses and units of capacity for which the license is granted will be one (1), unless a different number of licenses or units of capacity is specified in the Documentation or other materials available to End User. "Designated Processor" means a single stand-alone computing device. "Server" means a Designated Processor that hosts a software application to be accessed by multiple users. "Software" means the computer programs in object code, originally licensed by Avaya and ultimately utilized by End User, whether as stand-alone Products or pre-installed on Hardware. "Hardware" means the standard hardware Products, originally sold by Avaya and ultimately utilized by End User.

License type(s)

Concurrent User License (CU). End User may install and use the Software on multiple Designated Processors or one or more Servers, so long as only the licensed number of Units are accessing and using the Software at any given time. A "Unit" means the unit on which Avaya, at its sole discretion, bases the pricing of its licenses and can be, without limitation, an agent, port or user, an e-mail or voice mail account in the name of a person or corporate function (e.g., webmaster or helpdesk), or a directory entry in the administrative database utilized by the Product that permits one user to interface with the Software. Units may be linked to a specific, identified Server.

Copyright

Except where expressly stated otherwise, the Product is protected by copyright and other laws respecting proprietary rights.

Unauthorized reproduction, transfer, and or use can be a criminal, as well as a civil, offense under the applicable law.

Third-party components

Certain software programs or portions thereof included in the Product may contain software distributed under third party agreements ("Third Party Components"), which may contain terms that expand or limit rights to use certain portions of the Product ("Third Party Terms"). Information identifying Third Party Components and the Third Party Terms that apply to them is available on the Avaya Support Web site:

http://support.avaya.com/ThirdPartyLicense/

Preventing toll fraud

"Toll fraud" is the unauthorized use of your telecommunications system by an unauthorized party (for example, a person who is not a corporate employee, agent, subcontractor, or is not working on your company's behalf). Be aware that there can be a risk of toll fraud associated with your system and that, if toll fraud occurs, it can result in substantial additional charges for your telecommunications services.

Avaya fraud intervention

If you suspect that you are being victimized by toll fraud and you need technical assistance or support, call Technical Service Center Toll Fraud Intervention Hotline at +1-800-643-2353 for the United States and Canada. For additional support telephone numbers, see the Avaya Support Web site: http://support.avaya.com

Trademarks

Avaya and the Avaya logo are either registered trademarks or trademarks of Avaya Inc. in the United States of America and/or other jurisdictions. All other trademarks are the property of their respective owners.

Downloading documents

For the most current versions of documentation, see the Avaya Support Web site: http://support.avaya.com

Avaya support

Avaya provides a telephone number for you to use to report problems or to ask questions about your product. The support telephone number is 1-800-242-2121 in the United States. For additional support telephone numbers, see the Avaya Support Web site: http://support.avaya.com

Contents

Chapter 1: About this document 7
Intended audience 7
Related documents 8
What's New in System Manager 6.3.4 User Web Services 8
What's New in System Manager 6.3.8 User Web Services 9
What's New in System Manager 7.0 User Web Services 9
What's New in System Manager 7.0.1 User Web Services 10
What's New in System Manager 7.1 User Web Services 11
What's New in System Manager 7.1.2 User Web Services 11
What's New in System Manager 8.1.3 User Web Services 12
What's New in System Manager 8.1.11 User Web Services 12
What's New in System Manager 8.1.12 User Web Services 13
What's New in System Manager 10.1 User Web Services 13
Chapter 2: Getting started 14
Setting up the development environment 14
Setting up and running a sample web services Java client 16
Configuring a sample web services Java client 16
Chapter 3: Writing a client application 19
Understanding RESTful web services 19
User Management API (web services interface) details 20
System Manager web service requests (Version 1.0) 20
Create user 21
Update user 23
Retrieve users with minimal user data based on search criteria 26
Get user profile 30
Delete user 32
System Manager web service requests (Version 2.0) 34
Create user 34
Update user 40
Retrieve users with minimal user data based on search criteria 46

Get user profile 53
Delete user 82
Self provisioning password feature (web services interface) details 87
SIP Password Change 88
Endpoint Password Change 90
Agent Password Change 91
Messaging Password Change 92
Reset Password 94
Activate Password 96
Changing Multiple Communication Profile Passwords in single API 97
Exceptions 99
Multi-tenancy considerations 101
Security considerations 102
Authentication 102
Generating certificates 103
System Manager Internal CA certificate are in use. 103
Third-party CA Issued certificate are in use. 103
Sample code 104
Sample code to establish SSL connection 104
Sample code snippet to create, update and retrieve users 104
Chapter 4: Debugging 108
Getting support 108
Appendix A – Sample code 108
Appendix B – XSD as attachments 109
Appendix C – HTTP verbs 110
Appendix D – Error Code 111
Appendix E – Must Read 120
Appendix F – XML Schema Definitions 122
XML Schema Definition for retrieving and creating User(s): Basic Attributes 122
XML Schema Definition for updating User(s): Basic Attributes 139
XML Schema Definition for retrieving, creating and updating and User(s): Session Manager 142
XML Schema Definition for retrieving, creating and updating and User(s): Breeze 146
XML Schema Definition for retrieving, creating and updating and User(s): Endpoint 147
XML Schema Definition for retrieving, creating and updating and User(s): Messaging 200
XML Schema Definition for retrieving, creating and updating and User(s): Agent 229
XML Schema Definition for retrieving, creating and updating and User(s): CS1000 and Callpilot 241

XML Schema Definition for retrieving, creating and updating and User(s): Avaya Branch Gateway 242
XML Schema Definition for retrieving, creating and updating and User(s): Presence 262
XML Schema Definition for deleting User(s) 262
XML Schema Definition for retrieving, creating and updating and User(s): Work Assignment 264
XML Schema Definition for retrieving, creating and updating and User(s): Officelinx 265
XML Schema Definition for retrieving, creating and updating and User(s): Equinox / Scopia 266
XML Schema Definition for retrieving, creating and updating and User(s): Messaging 267
Appendix G – XSD attributes description 296
XSD attribute Definition for retrieving, creating and updating and User(s): Basic Attributes 296
XSD attribute definition for deleting User(s) 307
XSD attribute definition for retrieving, creating and updating and User(s): Messaging 307
XSD attribute definition for retrieving, creating and updating and User(s): Session Manager 323
XSD attribute definition for retrieving, creating and updating and User(s): Endpoint 326
XSD attribute definition for retrieving, creating and updating and User(s): Work Assignment 351
XSD attribute definition for retrieving, creating and updating and User(s): Officelinx 351
XSD attribute definition for retrieving, creating and updating and User(s): Equinox / Scopia 352

C About this document

The Avaya Aura® System Manager user management web service interface will support provisioning of users in the System Manager.

The purpose of this document is to describe:

- The Avaya Aura® System Manager user management web services (UM WS) interfaces available for managing user profiles.
- The sample Java client configurations and specifications to manage user profiles.

Note: A.NET application can also interface to the UM WS. This guide describes the sample Java client.

This document provides the information needed to develop, debug, and deploy a web services client for managing user profiles using UM WS implementation. It discusses the interface definition of the UM WS and explains how to write a sample client that uses UM WS.

- Chapter 1: About this document gives an overview of this document, identifies certain prerequisites needed to effectively utilise its contents and lists supporting reference documents.
- Chapter 2: Getting started provides information needed to configure and execute the sample client application.
- Chapter 3: Writing a client application provides details of each UM WS API, System Manager UM WS URL, input and output parameters, and illustrates the API with a code snippet, as well as a walkthrough of the sample code.
- Chapter 4: Debugging provides details of the logs and exceptions.
- Appendix A provides Java sample code files and XML files.
- Appendix B lists the XML schema definition.
- Appendix C lists the HTTP verbs.
- Appendix D provides the error code.
- The Glossary defines the terminology and acronyms used in this document.

Intended audience

This document is written for applications developers who are familiar with:

- Basic RESTful web services programming
- XML Schema Definition (XSD)

To learn more about Avaya Aura® System Manager and its features see http://www.avaya.com/support.

Related documents

While planning, developing, deploying, or troubleshooting your application, you might need to reference the following Avaya Aura® System Manager document available on the Avaya Support Center website at http://www.avaya.com/support.

[1] Administering Avaya Aura® System Manager Release (Issued for System Manager 6.2): Provides additional information about administering Avaya Aura® System Manager.

What's New in System Manager 6.3.4 User Web Services

System Manager Release 6.3.4 introduces following new features

- Multi-tenancy feature - See Section Multi-tenancy Considerations for more details.
- User provisioning Rule – You can create a user using user provisioning rule via user web services. When the administrator creates the user using the user provisioning rule, the system populates the default values, the communication addresses, and the communication profiles for the user based on the rules defined in the user provisioning rule.

  Note: An administrator can create rules called user provisioning rules using System Manager Web console. In the user provisioning rule, the administrator specifies the following information to provision the user:
  - Basic information that includes communication profile password, time zone, language preference.
  - The communication system that the user must use. For example, Communication Manager and Session Manager.
  - The method to assign or create a communication profile for the user. For example, by assigning the next available extension for Communication Manager.

- Latin translation of first name and surname of the user and contact - When you enter the first name and the last name of the user associated with an endpoint, the Latin translation of the first name and the last name is auto populated.

To see all new attributes introduced as part of System Manager 6.3.4, see NEW ATTRIBUTES INTRODUCED IN XSD IN SYSTEM MANAGER 6.3.4.

The XSD has been highlighted to display new attributes introduced in System Manager 6.3.4. See XML Schema Definition for retrieving and creating User(s): Basic Attributes.

For Update user, see XML Schema Definition for updating User(s): Basic Attributes.

Also, the updated Create User (userimport.xsd) and Update User (userdeltaimport.xsd) has been attached in Appendix B.

What's New in System Manager 6.3.8 User Web Services

System Manager Release 6.3.8 introduces following new UM WS feature

- Change Login Name through UM Web Service: As an administrator, I want to be able to change a User's login name through the Web Service interface. Currently I can only change a User's login name through the UI.

To see all new attributes introduced as part of System Manager 6.3.8, see NEW ATTRIBUTES INTRODUCED IN XSD IN SYSTEM MANAGER 6.3.8.

The XSD has been highlighted to display new attributes introduced in System Manager 6.3.8. See XML Schema Definition for retrieving and creating User(s): Basic Attributes.

For Update user, see XML Schema Definition for updating User(s): Basic Attributes.

Also, the updated Create User (userimport.xsd) and Update User (userdeltaimport.xsd) has been attached in Appendix B.

What's New in System Manager 7.0 User Web Services

System Manager Release 7.0 has introduced new attributes in Endpoint Communication Profile XSD.

Dual Registration: This field will provide the ability to automatically generate an OPS entry for an H.323 set type. The default value is 'false'.
Calculate Route Pattern: This field will allow SMGR to automatically select the Route Pattern for a SIP endpoint based on the Primary and Secondary Session Manager. The default value is 'false'.
Enable Reachability for Station Domain Control: This is the new field supported in station object. This field can accept either system/yes/no. The default value is 'system'.

To see these new attributes introduced as part of System Manager 7.0, see Appendix F: XSD attribute definition for retrieving, creating and updating and User(s): Endpoint.
The Endpoint XSD is updated at XML Schema Definition for retrieving, creating and updating and User(s): Endpoint.

System Manager 7.0 introduces Updated Messaging XSD. Now it is possible to do CRUD operation using UM WS of all attributes which are available on Edit Subscriber console page. To see these new attributes introduced as part of System Manager 7.0, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): Messaging

System Manager Release 7.0 has introduced new Work Assignment Communication Profile XSD. To see this new XSD introduced as part of System Manager 7.0, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): Work Assignment.

System Manager 7.0 introduces Updated Presence XSD. To see these new attributes introduced as part of System Manager 7.0, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): Presence

What's New in System Manager 7.0.1 User Web Services

System Manager 7.0.1 has updated Messaging XSD with new field 'Time Zone'. This is time zone for Avaya Aura Messaging time subscribers. This attribute must be in 'Standardized Name' format. Example: 'America/Phoenix' and the administrator must be sure that entering time zone exists on AAM Server; otherwise the System Manager Server time zone will be set for AAM subscriber.
To see these new attributes introduced as part of System Manager 7.0.1, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): Messaging

System Manager 7.0.1 has updated Endpoint XSD, added following fields for supporting CallrId and K2500 Set types: displayCallerId; callerIdMsgWaitingIndication and recallRotaryDigit
To see these new attributes introduced as part of System Manager 7.0.1, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): Endpoint

System Manager 7.0.1 has updated Agent XSD, added following fields: auxAgentConsideredIdleMIA
To see these new attributes introduced as part of System Manager 7.0.1, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): Agent

What's New in System Manager 7.1 User Web Services

System Manager 7.1 has added Offlicelinx XSD to support Offlicelinx Communication Profile.
To see new attributes of OffliceLinx XSD as part of System Manager 7.1, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): OffliceLinx

System Manager 7.1 has added Equinox XSD to support Equinox / Scopia Communication Profile.
To see new attributes of OffliceLinx XSD as part of System Manager 7.1, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): Equinox

System Manager 7.1 support Password feature related APIs through self-provisioning e.g. SIP password change, CM Endpoint password change, Agent password change, Messaging password Change, Reset Communication Profile password, Activate Communication Profile password, Password Manager to change passwords.
F: Self provisioning password reset feature (web services interface) details

What's New in System Manager 7.1.2 User Web Services

System Manager 7.1.2 has added new attributes in Messaging Communication Profile XSD Attributes added are **Tui Message Order Unread**, **Play Unread Urgent First**, **Arrange By Sender Unread**, **Tui Message Order Read**, **Play Read Urgent First**, **Arrange By Sender Read**, **Tui Message Order Saved**, **Play Saved Urgent First** and **Arrange By Sender Save**
To see new attributes of Messaging XSD as part of System Manager 7.1.2, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): Messaging

What's New in System Manager 8.1.3 User Web Services

1. System Manager 8.1.3, default passwords are not set for any system manager users that are added/updated. The password field shall become mandatory in following scenarios:
   - When using the create user web service API if one or more roles (anyone except end-user role) are added to the user, then the password must be provided using the 'userPassword' element.
   - When using the update user web service API if one or more roles (anyone except end-user role) are added to the user, then the password must be provided using the 'userPassword' element. This scenario is applicable for merge/replace "updateMode" element in the XML and for a user that currently only has the end-user role assigned or no roles assigned.

2. Starting SMGR 8.1.3, a new version of the REST API has been implemented which improves response format and adhering to standard REST API guidelines with regards to error handling in REST API.

   URL for this new REST API is - **https://<System Manager Server FQDN>/web/v2/mgmtwebservice/<URL parameter(s) >**.

   This new API behaves the same as previous one in terms of the input parameters and their handling. The only difference between two API versions is the response format and the HTTP status codes.

What's New in System Manager 8.1.11 User Web Services

System Manager 8.1.11 has added new attributes in Session Manager Communication Profile XSD Attributes added are **policyType, thirdSM, fourthSM, primaryFixedRegion, secondaryFixedRegion, thirdFixedRegion** and **fourthFixedRegion.**
To see new attributes of Session Manager XSD as part of System Manager 8.1.11, see Appendix F: XML Schema Definition for retrieving, creating and updating and User(s): Session Manager and XSD attribute definition for retrieving, creating and updating and User(s): Session Manager

What’s New in System Manager 8.1.12 User Web Services

System Manager 8.1.12 has introduced an optional query parameter in the Get user method(V2), to fetch only selective portions of User data. If the optional query parameter is not passed, then it will retrieve the complete data of the user as before. See Section Get User Profile for more details.

What’s New in System Manager 10.1 User Web Services

Starting System Manager 10.1, Conferencing communication profile is no longer supported.
Some fields in the Messaging communication profile are no longer supported starting SMGR 10.1. Refer to sections "XML Schema Definition for retrieving, creating and updating and User(s): Messaging" and "XSD attribute definition for retrieving, creating and updating and User(s): Messaging" for more details.

C Getting started

The System Manager UM WS interface supports provisioning of users, specifically User profile management provisioning fields that are available in the System Manager User Interface. Using System Manager UM WS, you can:

- Create and update user profiles and associate users with groups, roles, and communication addresses, for example, handle and domain, and communication profiles. Communication profiles include the Avaya Aura® Session Manager (SM) profile, Avaya Aura® Communication Manager endpoint profile, CS 1000 endpoint profile, Avaya Aura® Messaging profile, CallPilot messaging profile, and B5800 Branch Gateway endpoint profile.
- Create and update the user profile with a contact list, addresses, and private contacts.
- Retrieve a list of users with minimal user data based on search criteria.
- Obtain complete user profiles based on the login name.
- Delete user profiles.

Note:
Performing different tasks requires different permissions. A **role** is a collection of permissions that you can assign to users. You can **group** resources such as users in any way that works best for the business, such as, organizing by location, organization, and function. A **communication profile** represents a user subscription to Avaya Communication Services and contains the specific configuration needs of the user. A **contact list** is a collection of internal or external contacts of the user.
To find out more about groups, roles, communication profiles, and contact list, see [1]

This section describes what you must do incase you are developing a JAVA solution and what you must know before you begin programming to this API, including:

- Setting up the development environment
- Setting up and running sample Java client
- Configuring sample Java client

Note: If you are using .NET to develop web services client, above is not a must.

Setting up the development environment

Before running an application, you must set up a development machine for the web services client and install System Manager.

To support the web services sample Java client, install Sun Microsystems™ Java 2 Platform, Standard Edition (J2SE™) 1.6 or later.

Downloading Java SDK

Your development machine must have a 7 Java SE Java Development Kit (JDK) or open JDK 7 installed, 1.7.0_01 or later. If you have not installed Java SDK, download the software from the Oracle Web site:

1. Download a Java SE 7 JDK, 1.7.0_01 or later, from http://www.oracle.com/technetwork/java/javase/downloads/index.html.
2. Install Java SE JDK according to the instructions provided by Oracle.
3. Make sure that the system PATH has been updated to point to this JDK.
4. Run a sample Java program to verify that the installation of the Java platform is correct.

Downloading Apache Ant

1. Download the latest Apache Ant from http://ant.apache.org/bindownload.cgi
2. Install the Apache Ant according to the instructions provided at http://ant.apache.org/manual/install.html
3. Make sure that the system PATH has been updated to point to this Apache ANT.

Obtaining System Manager user management web service API SDK

Please contact devconnect@avaya.com to System Manager UM WS SDK.

The various artifacts in this SDK distribution include the following:

Directory / File	Description
sdk/JavaSampleClient/conf	Includes client configuration files: **`config.properties`** and **`client.properties`**.
sdk/JavaSampleClient/lib	Includes the required dependency JARs. For example, **`jersey-bundle-1.4.jar`** , **`jsr311-api-1.1.1.jar`**, **`jaxb-impl.jar`** ,**`asm-3.1.jar`**
sdk/JavaSampleClient/src	Includes the RESTful webservice sample client source code.
sdk/JavaSampleClient/xsd	Includes XSD for reference. To find out more about XSD, see [1]
sdk/JavaSampleClient /build.xml	Includes an ANT script that compiles and builds a sample client component for execution.

user management web services component

User management web services component is installed on the System Manager server during the System Manager installation.

Third-party requirements

You need the following third-party JARs to compile and run the sample Java client:

- jersey-bundle-1.4.jar
- jsr311-api-1.1.1.jar
- jaxb-impl.jar
- asm-3.1.jar

The sample client application has all the above third-party JARs packaged in the lib folder.

Setting up and running a sample web services Java client

A test client with source code is included in the SDK. The test client demonstrates the basic interaction with System Manager user management web service. The client executes a single service request and displays the result, releases the session, and exits.

The sample code is located in the following directory:
`sdk/JavaSampleClient/src`

A trust store certificate is required to establish an SSL connection. For installing a System Manager certificate, see Installing System Manager Certificate.

To run the program, navigate to the **`sdk/JavaSampleClient/conf`** folder and update the **`config.properties`** file with the correct information for your System Manager Server, username, password, and trust store location.

The sample also provides another configuration file named **`client.properties.`** Using this property file, you can configure user operation, update mode, and delete type, criteria as mentioned in the section configuring your sample java client.

Next navigate to the **`sdk/JavaSampleClient/`** directory on the command line interface and execute: **`ant.`** The system executes the default command for the test application using the selected System Manager and login.

Configuring a sample web services Java client

The following is a list of configurable attributes with a brief description from **`config.properties`**:

Properties	Description

baseURL	https://<<smgr server name>>/web/mgmtwebservice For example, https://example.com/web/mgmtwebservice
trustStorePath	<<truststore file name>>
trustStorePassword	<<trust store password>>
Username	<<system manager user name>> For example, example@avaya.com
Password	<<system manager user password>> For example, examplepassword

The sample Java client provides a configuration file **`client.properties`**. This file enables you to set various properties such as operation, entity type and others as mentioned in below table.

Create User

You want to create a single user with all user data, you can set **`operation.type`** as **`create`**, **`entity.type`** as **`user`** and **`user.create.type`** as **`all`** in **`client.properties`**. With this property file settings, when you execute the sample client, a single user with all user data will be created. Default user(s) data is already available in the sample client for the Create user process. The property **`station.extension.start.The index`** is used for specifying station extension value for the user account created.

Update User

For update user using the sample client, you can specify the **`user.update.mode`** in the **`client.properties`** as **`merge/replace.`** You need to specify **`user.criteria`** in the **`client.properties.`** Based on the criteria the user list will be created and user's Middle name will be updated by the sample client.

Delete User

For delete user using the sample client, you can specify the **`user.delete.type`** in the **`client.properties`** as **`soft/permanent.`** You need to specify **`user.criteria`** in the **`client.properties.`** Based on the criteria the user list will be created and deleted by the sample client. **Note**: Ensure that you specify appropriate criteria for retrieving users.

Retrieve and Find User

For retrieve user using the sample client, you need to specify **`user.criteria`** in the **`client.properties.`** Based on the criteria the user list will be created and displayed on the console by the sample client.

The following is a list of configurable attributes with a brief description from **`client.properties`**:

Properties	Description
operation.type	Select the required operation : [create | update | lookup | find | delete]

entity.type	EntityType : [user]
user.create.type	User attributes: [mandatory | station | all] ▪ mandatory: User mandatory attributes ▪ station: User attribute and endpoint communication profile attributes ▪ all: User attribute and endpoint communication profile attributes
user.update.mode	update mode: [merge | replace]
user.delete.type	delete type: [soft | permanent] • soft: Softly delete, the user will persist in the system and can be restored • permanent: Permanently delete the user from the System Manager database and the same cannot be restored
user.criteria	startindex:equals:0#offsetindex:equals:10 o criteria structure: [propertyName:condition:value] o PropertyName: [startindex | offsetindex | loginName | givenName | surname | displayName] o Conditions: [equals | startsWith | endsWith | contains] o value: xxxxx o | represents OR o for example: loginName:startsWith:jh ▪ # is criteria de-limiter
station.extension start. Index	Endpoint station start range #725000

Chapter 3: Writing a client application

This chapter describes how to write an application using the web services.

- Understanding RESTful web services
- API (web services) details
- System Manager web service requests
  - Create User
  - Update User
  - Retrieve users with limited user data based on search criteria
  - Get User profile
  - Delete user
- Accessing the client API reference documentation
- Exceptions
- Security Considerations
  - Authentication
  - Generating certificates
- Sample Code
  - Sample code snippet to establish SSL connection and authentication
  - Sample code snippet to create, update and retrieve list of users

Understanding RESTful web services

The System Manager User Profile Management web services are designed as per the **REST** architecture.

Representational State Transfer (REST) is an architectural style that specifies constraints, such as the uniform interface, that if applied to a web service induces desirable properties, such as performance, scalability, and modifiability that enable services to work best on the Web. In the REST architectural style, data and functionality are considered resources, and these resources are accessed using Uniform Resource Identifiers (URIs), typically links on the web. The resources are acted upon by using a set of simple, well-defined operations. The REST architectural style constrains architecture to client-server architecture and is designed to use a stateless communication protocol, typically HTTP. In the REST architecture style, clients and servers exchange representations of resources using a standardised interface and protocol. These principles encourage RESTful applications to be simple, lightweight and have high performance.

RESTful web services typically map the four main HTTP methods to the operations they perform: create, retrieve, update, and delete. The following table shows a mapping of HTTP methods to the operations they perform.

User Management API (web services interface) details

This section includes the implementation details of the User Profile Management web services.

The System Manager user management web service provides the following methods:

- **String post(String entityType, String entityXMLData,String @DefaultValue create operationName)**
- **String update(String entityType, String entityXMLData, String updateMode)**
- **String lookup(String entityType, String searchCriteria)**
- **String find(String entityType, String uniqueID)**

The parameter entityType must be **user** for all user functionalities: Create User, Update User, Delete User, Lookup User, and Find User.

The entity data must be in XML format and must conform to the W3C XML standard as per the schema defined. For XML schema definition (XSD), see Appendix B.

The URL for the REST-based web services to be accessed is - **https://<System Manager Server FQDN>/web/mgmtwebservice/<URL parameter(s)>**

Starting SMGR 8.1.3 a new version of the REST API (referred to as version 2.0 in rest of the document) has been implemented which improves response format and adhering to standard REST API guidelines with regards to error handling in REST API.

URL for this new REST API is - **https://<System Manager Server FQDN>/web/v2/mgmtwebservice/<URL parameter(s) >**

This new API behaves the same as previous one (referred to as version 1.0 in rest of the document) in terms of the input parameters and their handling. The only difference between two API versions is the response format and the HTTP status codes.

Note: For backward compatibility previous URL will continue to work the way it used to work earlier.

System Manager web service requests (Version 1.0)

The URL for the REST-based web services to be accessed is - **`https://<System Manager Server FQDN>/web/mgmtwebservice/<URL parameter(s)>`**

This is version of the REST API that is around since release 6.3.x

Each request contains the **`username`** and **`password`** and System Manager performs authentication through the HTTP Basic authentication scheme. The client communicates with the server over SSL using the server certificate. See Sample code snippet to establish SSL connection and authentication.

Create user

The Create User operation creates one or more user profiles in System Manager with the given user data in XML format.

`String post(String entityType, String entityXMLData,String operationName)`

The Create User operation has the following components:

HTTP method	POST
URL	*https://host/web/mgmtwebservice* *the host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	Refer to Request Parameters section below.
Request Parameters	None.
Response	Refer to Response section below.

Request parameters

The Create User Call request takes the following parameters:

Name of the parameter	Data type	Type	Description
entityType	String	Form	Specifies the entity name. Expected value: **`user`**
entityXMLData	String	Form	Specifies User XML data conforming to the specified XSD. See Appendix A
operationNam e	String	Form	Specifies operation name. Expected value: create. The default value set for this parameter is create.

Response

The response contains the user status. The response generates a list of status (data type: String) messages.

For example,

```
Operation=Create User, LoginId, Action: Create,
Object=johnmiller@sip.com, Status:  Success
Operation=Create User, LoginId:, Action: Create,
Object=joemiller@sip.com, Status: Warning!"UPM_1102 user
with this LoginName already exists (maybe the user you are
adding has been soft deleted)"
```

Code snippet

The following is the Java client sample code to create a user with mandatory attributes using RESTful web services. The post request takes two FORM parameters: **entityType** and **entityXMLData**.

createsample.xml – The following is a sample xml file for Create User

```
<?xml version="1.0" encoding="UTF-8"?>
<tns:users xmlns:tns="http://xml.avaya.com/schema/import"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xml.avaya.com/schema/importuserimport
.xsd">     <tns:user>
           <authenticationType>Basic</authenticationType>
           <givenName>John</givenName>
           <loginName>jmiller@avaya.com</loginName>
           <surname>Miller</surname>
           <userPassword>mypassword</userPassword>
      </tns:user>
</tns:users>
```

Sample code – The following is a sample Java code to create a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
  private String uri = "https://example.com/web/mgmtwebservice";
…
      final InputStream inputStream = getClass().getClassLoader()
                        .getResourceAsStream("createsample.xml");
      final Reader reader = new InputStreamReader(inputStream);
      final BufferedReader br = new BufferedReader(reader);
      final StringBuffer line = new StringBuffer();
      String st = "";
      while ((st = br.readLine()) != null)
      {
            line.append(st + "\n");
      }
      Final String entityXmlData = line.toString();
      final Form form = new Form();
```

```
      form.add("entityType", "user");
      form.add("entityXMLData", entityXmlData);
      form.add("operationName", “create”); //optional

      String result = client.resource (uri).post(String.class,
   form);
…
```

The following statement adds **`entityType`**, **`entityXMLData`** and **`operationName`** parameter to the form class object respectively.

```
form.add("entityType", "user");
form.add("entityXMLData", entityXmlData);
form.add("operationName", “create”); //optional
```

`form` object is further passed to the post request and returns a list of status messages.

```
String result = client.resource (uri).post (String.class, form);
```

Sample Resultant stream for **`create`**:

```
 Operation=Create User, LoginId: , Action: Create,
Object=johnmiller@sip.com, Status:  Success
```

Note: For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

Update user

The update method updates one or more user profile in the System Manager with the given user data in XML format.

`String update (String entityType, String entityXMLData, String updateMode`

The **`updateMode`** parameter can be **`merge`** or **`replace`**. With the replace option, you can replace a user and data associated with the user. With the merge option, you can add data in collection value and update the single value user data. For example, add a contact to a contact list and update the last name of the user.

Web services client can retrieve a list of users using lookup API (see Retrieve users with minimal user data based on search criteria), modify user data, and then invoke update API to update user data.

The Update User operation has the following components:

HTTP method	PUT
URL	*https://host/web/mgmtwebservice* *host* is the FQDN of the System Manager host servicing the request.

Headers	Content-Type: application/x-www-form-urlencoded
Body	Refer to Request Parameters section below.
Request Parameters	None.
Response	Refer to Response section below.

Request parameters

The Update User Call request takes the following parameters:

Name of the parameter	Data type	Type	Description
entityType	String	Form	Specifies an entity name. Expected value: **user**
entityXMLData	String	Form	Specifies User XML data conforming to the specified XSD. See Appendix A
updateMode	String	Form	Specifies the type of update that you must perform. Possible values: **merge** or **replace**

Response

The response contains the user status and generates a list of status (data type: String) messages. For example,

```
Operation=Update User, LoginId: , Action: Update,
Object=johnmiller@sip.com, Status:  Success
Operation=Update User, LoginId: , Action: Update,
Object=joemiller@sip.com, Status: Error!"UPM_XXXX user with this
loginname does not exist."
```

Code snippet

The following is the Java client sample code to update a user with mandatory attributes using RESTful web services. The put request takes three FORM parameters: **`entityType, entityXMLData`** and **`updateMode`**.

updatesample.xml – The following is a sample xml file for Update User

```
<?xml version="1.0" encoding="UTF-8"?>
<delta:deltaUserList
xmlns:delta="http://xml.avaya.com/schema/deltaImport"
      xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://xml.avaya.com/schema/deltaImport
userdeltaimport.xsd ">
      <delta:userDelta>
            <loginName>jmiller@avaya.com</loginName>
            <middleName>Merged middle name</middleName>
```

```
      </delta:userDelta>
</delta:deltaUserList>
```

Sample code – The following is a sample code snippet to update a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
private String uri = "https://example.com/web/mgmtwebservice";
…
     final InputStream inputStream = getClass().getClassLoader()
                        .getResourceAsStream("updatesample.xml");
      final Reader reader = new InputStreamReader(inputStream);
      final BufferedReader br = new BufferedReader(reader);
      final StringBuffer line = new StringBuffer();
      String st = "";
      while ((st = br.readLine()) != null)
      {
            line.append(st + "\n");
      }
      Final String entityXmlData = line.toString();
      final Form form = new Form();
      form.add("entityType", "user");
      form.add("entityXMLData", entityXmlData);
      form.add("updateMode", "merge");
      String result = client.resource(uri).post(String.class,
form);
…
```

The following statement adds **`entityType`**, **`entityXMLData`**, and **`updateMode`** parameters to the form class object respectively.

```
form.add ("entityType", "user");
form.add ("entityXMLData", entityXmlData);
form.add ("updateMode", "merge");
```

`form` object is further passed to the put request and returns a list of status messages.

```
String result = client.resource (uri).put (String.class,
form);
```

Sample Resultant stream for *update*:

```
Operation=Update User, LoginId: , Action: Update, merge,
Object=jmiller@avaya.com, Status: Success
```

Note: For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

Retrieve users with minimal user data based on search criteria

The retrieve method enables search for one or more user profiles within System Manager with the given search criteria.

`String lookup (String entityType, String searchCriteria)`

The search criteria are based on the following user attributes: First Name, Last Name, Display Name, and Login name.

The lookup method has been tested and recommended to fetch 500 users at a time.

The Retrieve users operation has the following components:

HTTP method	GET
URL	*https://host/web/mgmtwebservice* *host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	Refer to Request Parameters section below.
Response	Refer to Response section below.

Request parameters

Name of the parameter	Data type	Type	Description
entityType	String	Query	Specifies an entity name. Expected value: **user**
searchCriteria	String	Query	Specifies the search criteria for listing the user profiles. These criteria should conform to Search Criteria XSD.

Response

The user profile search result includes one or more user profile data in XML format. The minimal user data in XML consists of First Name, Middle Name, Last Name, Display Name, source, and Login Name.

Sample Resultant stream content for **`lookup`**:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users
xsi:schemaLocation="http://xml.avaya.com/schema/import
```

```
userimport.xsd" xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_mm"
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns9="http://xml.avaya.com/schema/deltaImport"
xmlns:ns5="http://xml.avaya.com/schema/import_csm_abg"
xmlns:ns6="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns7="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm">
    <tns:user>
        <givenName>John</givenName>
        <loginName>jmiller@avaya.com</loginName>
        <source>TC_0101_1.xml</source>
        <sourceUserKey>none</sourceUserKey>
        <surname>Miller</surname>
        <userName>jmiller</userName>
    </tns:user>
</tns:users>
```

Code snippet

Sample code – The following is a sample code snippet to retrieve a list of users using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
private String uri = "https://example.com/web/mgmtwebservice";
…
     final InputStream inputStream = getClass().getClassLoader()
                .getResourceAsStream("searchcriteria.xml");
      final Reader reader = new InputStreamReader(inputStream);
      final BufferedReader br = new BufferedReader(reader);
      final StringBuffer line = new StringBuffer();
      String st = "";
      while ((st = br.readLine()) != null)
      {
            line.append(st + "\n");
      }
      String searchCriteria = line.toString();
      searchCriteria = searchCriteria.replaceAll("&", "&amp;")
            .replaceAll("<", "&lt;").replaceAll(">", "&gt;")
      .replaceAll("\"", "&quot;").replaceAll("'", "&apos;");

      String result =
      client.resource(uri).queryParam("entityType",
                "user").queryParam("searchCriteria",
            searchCriteria)
                get(String.class);
…
```

The last statement in the sample code to establish an SSL connection and authenticate the user, QUERY parameter **`entityType`** and **`searchCriteria`** is passed along with the GET request to the server.

Note: For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

Search criteria

You can search for users based on the following criteria:

User data:

User attributes	Description
First Name	Identifies the first name of the user. Searches are based on the first name. You can refine the search by specifying conditions such as equals, contains, starts with, or ends with the value of the user First Name. For example, you can specify the search criteria "First Name equals Joe" by selecting the equals condition and entering Joe in the First Name field.
Last Name	Identifies the last name of the user. Searches are based on the last name. You can refine the search by specifying conditions such as equals, contains, starts with, or ends with the value of the user Last Name. For example, you can specify the search criteria "Last Name equals Miller" by selecting the equals condition and entering Miller in the first name field.
Display Name	Identifies the display name of the user. Searches are based on Display name. You can refine the search by specifying conditions such as equals, contains, starts with, or ends with the value of the user Display Name. For example, you can specify the search criteria "Display Name equals Joe Miller" by selecting the equals condition and entering Joe Miller in the display name field.
Login Name	Identifies the log-in name of the user. You can refine the search by specifying conditions such as equals, contains, starts with, or ends with the value of the user Login Name. For example, you can specify the search criteria "Login Name equals jmiller@abc.com" by selecting the equals condition and entering jmiller@abc.com in the login name field.
Start Index	Sorts the database user record ID in ascending order.
Offset Index	Identifies the number of records that will be retrieved starting from the specified Start Index.

Operators:

Operators	Description
Equals	Refine the search criteria by using the equals operator. Equals imply having the same value as another.
StartsWith	Implies the value will preclude the same value as another.
endsWith	Implies the value will end with the same value as another.
Contains	Refine the search criteria by using the contains operator. Contains implies including or comprising the same value as another.

Restrictions:

Restrictions	Description
And	A Boolean operator that gives the value one only if all the operands are one and otherwise has a value of zero.
Or	A Boolean operator that gives the value one if at least one operand (or input) has a value of one and otherwise has a value of zero.

searchcriteria.xml – The following is a sample XML file to specify criteria to retrieve a list of users

```
  <?xml version="1.0" encoding="UTF-8"?>
  <tns:management-webservice-criteria
xmlns:tns="http://www.avaya.com/mgmt-web-criteria/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.avaya.com/mgmt-web-criteria/
mgmt-web-criteria.xsd">
        <criterialist>
                        <criteria>
            <propertyname>startindex</propertyname>
            <operator>equals</operator>
            <value>0</value>
            <restriction>and</restriction>
          </criteria>
          <criteria>
            <propertyname>offsetindex</propertyname>
            <operator>equals</operator>
            <value>10</value>
            <restriction>and</restriction>
          </criteria>
        </criterialist>
```

```
</tns:management-webservice-criteria>
```

Get user profile

The get user method enables retrieving a user profile with complete user data based on the login name.

`String find (String entityType, String uniqueID)`

The Get user operation has the following components:

HTTP method	GET
URL	*https://host/web/mgmtwebservice* *host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	Refer to Request Parameters section below.
Response	Refer to Response section below.

Request parameters

Name of the parameter	Data type	Type	Description
entityType	String	Path	Specifies an entity name. Expected value: **user**
uniqueID	String	Path	Specifies the User login name.

Response

The get user profile search result includes one user profile data in XML format. The XML contains complete user data and no filter is applied.

The response of the request consists of user profile details (all user attributes) based on the **uniqueId** sent as an input parameter.

Code snippet

The following is the Java client sample code to get the user profile using RESTful web services. The GET request URL takes two **`PATH`** parameters: **`entityType`** and **`uniqueId`**.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
private String uri = "https://example.com/web/mgmtwebservice";
…
String result = client.resource(uri).path("user").
path("jmiller@avaya.com").get(String.class);
…
```

The same code passes PATH parameter **`entityType`** and **`uniqueId`** along with the get request to the server.

Note: For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

Sample Resultant stream content of **`find`** user:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users
xsi:schemaLocation="http://xml.avaya.com/schema/import
userimport.xsd" xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_mm"
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns9="http://xml.avaya.com/schema/deltaImport"
xmlns:ns5="http://xml.avaya.com/schema/import_csm_abg"
xmlns:ns6="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns7="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm">
    <tns:user>
        <authenticationType>basic</authenticationType>
        <displayName>Miller, John</displayName>
        <displayNameAscii>Miller, John</displayNameAscii>

<isDuplicatedLoginAllowed>false</isDuplicatedLoginAllowed>
        <isEnabled>true</isEnabled>
        <isVirtualUser>false</isVirtualUser>
        <givenName>John</givenName>
        <loginName>jmiller@avaya.com</loginName>
        <preferredLanguage>en_US</preferredLanguage>
        <source>TC_0101_1.xml</source>
        <sourceUserKey>none</sourceUserKey>
        <status>provisioned</status>
        <surname>Miller</surname>
        <userName>jmiller</userName>
        <userPassword></userPassword>
        <roles>
            <role>End-User</role>
        </roles>
        <ownedContactLists>
            <contactList>
```

```
                <name>list-jmiller_avaya.com</name>
                <isPublic>false</isPublic>
                <contactListType>general</contactListType>
            </contactList>
        </ownedContactLists>
        <commProfileSet>
            <commProfileSetName>Primary</commProfileSetName>
            <isPrimary>true</isPrimary>
        </commProfileSet>
    </tns:user>
</tns:users>
```

Delete user

The delete method deletes a user profile based on the login name provided in XML as **`entityXMLData`**.

`String post(String entityType, String entityXMLData, String OperationName)`

The delete User operation has the following components:

HTTP method	POST
URL	*https://host/web/mgmtwebservice* *host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	Refer to Request Parameters section below.
Request Parameters	None.
Response	Refer to Response section below.

Request parameters

Name of the parameter	Data type	Type	Description
entityType	String	Form	Specifies an entity name. Expected value: **`user`**
entityXMLData	String	Form	Specifies User XML data conforming to delete XSD. For more information on user delete XSD, see Appendix B.
operationName	String	Form	Specifies operation name. Expected value: delete.

Response

Returns a list of status (data type: String) messages.

For example,

```
Operation=Delete User, LoginId: , Action: Delete,
Object=johnmiller@sip.com, Status:  Success
Operation=Delete User, LoginId: , Action: Delete,
Object=joemiller@sip.com, Status: Error!"UPM_XXXX user with this
loginname does not exist."
```

The following is the Java client sample code to delete a user profile using RESTful web services. The delete request URL takes two **FORM** parameters: **entityType** and **entityXMLData**.

Code snippet

deletesample.xml –– The following is a sample XML file to specify criteria to retrieve a list of users

```
  <?xml version="1.0" encoding="UTF-8"?>
<tns:deleteUsers
xmlns:tns="http://xml.avaya.com/schema/bulkdelete"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xml.avaya.com/schema/bulkdelete
       UserProfileSchemaDefinitionForBulkDelete.xsd ">
      <tns:deleteType>permanent</tns:deleteType>
      <tns:user>
            <tns:loginName>jmiller@avaya.com</tns:loginName>
      </tns:user>
</tns:deleteUsers>
```

Sample code – The following is a sample code snippet to delete a user using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate
the user >>
…
private String uri = https://example.com/web/mgmtwebservice";
…
  final InputStream inputStream = getClass().getClassLoader()
                .getResourceAsStream("deletesample.xml");
  final Reader reader = new InputStreamReader(inputStream);
  final BufferedReader br = new BufferedReader(reader);
  final StringBuffer line = new StringBuffer();
      String st = "";
      while ((st = br.readLine()) != null)
      {
            line.append(st + "\n");
      }
  Final String entityXmlData = line.toString();
  final Form form = new Form();
      form.add("entityType", "user");
      form.add("entityXMLData", entityXmlData);
      form.add("operationName", "delete"); //required
```

```
      String result = client.resource(uri).post(String.class,
form);
 …
```

Sample Resultant stream for *delete*:

```
Operation=Delete User, LoginId: , Action: Delete, permanent,
Object=jmiller@avaya.com, Status: Success!
```

Note: You can soft delete or permanently delete a user. When you choose to soft delete a user, you can restore this user at a later time.When you choose to permanently delete a user, the process is irreversible and the deleted user cannot be restored. To specify the delete option, you can provide a permanent or soft delete value in the XML **`entityXMLData`** parameter. For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

System Manager web service requests (Version 2.0)

Starting SMGR 8.1.3, a new version of the REST API has been implemented which improves response format and adhering to standard REST API guidelines with regards to error handling in REST API.

URL for this new REST API is - **`https://<System Manager Server FQDN>/web/v2/mgmtwebservice/<URL parameter(s) >`**

This new API behaves the same as previous one in terms of the input parameters and their handling. The only difference between two API versions is the response format and the HTTP status codes.

Each request contains the **`username`** and **`password`** and System Manager performs authentication through the HTTP Basic authentication scheme. The client communicates with the server over SSL using the server certificate. See Sample code snippet to establish SSL connection and authentication.

Create user

The Create User operation creates one or more user profiles in System Manager with the given user data in XML format.

`String post(String entityType, String entityXMLData,String operationName)`

The Create User operation has the following components:

HTTP method	POST
URL	*https://host/web/v2/mgmtwebservice*

	the host is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	Refer to Request Parameters section below.
Request Parameters	None.
Response	Refer to Response section below.

Request parameters

The Create User Call request takes the following parameters:

Name of the parameter	Data type	Type	Description
entityType	String	Form	Specifies the entity name. Expected value: **`user`**
entityXMLData	String	Form	Specifies User XML data conforming to the specified XSD. See Appendix A
operationNam e	String	Form	Specifies operation name. Expected value: create. The default value set for this parameter is create.

Response

Successful response does not contain any data, but the HTTP status code of the response will be 201.

In case of an error, an appropriate HTTP status code is sent back along with a response XML. The Response XML conforms to the XSD described in Appendix B

Responses can contain below status codes for different scenarios.

HTTP Status Code	HTTP Status Message	Description
201	NA	User created successfully. No content is sent back in the response.
400	Bad Request	All validation exceptions.
409	Conflict	User with same login name already exists.
500	Internal Server Error	Any errors other than validation exception or username conflict. One may also get a 500 response code for cases where multiple users are created using a single request and each user creation in

		the request results with different HTTP status code. Please refer to example below.

For example,

Successful response

```
HTTP/1.1 201
Connection: Closed
No Content
```

Error Response

```
HTTP/1.1 400
Content-Type: application/xml
Connection: Closed
<ns1:errorResponse xmlns:ns1="http://www.avaya.com/errorResponse/">
    <code>400</code>
    <status>Bad Request</status>
    <message>Invalid request data</message>
    <ns2:error xmlns:ns2="http://www.avaya.com/error/">
        <message>Invalid request data</message>
        <uniqueId>test123@avaya.com</uniqueId>
        <details>
            <message>
                <name>CsUser.commPassword</name>
                <reasonCode>RLUPM0154</reasonCode>
                <reasonText>Communication Profile Password is
required for the SIP handle assigned</reasonText>
            </message>
        </details>
    </ns2:error>
</ns1:errorResponse>


HTTP/1.1 409
Content-Type: application/xml
Connection: Closed

<ns1:errorResponse xmlns:ns1="http://www.avaya.com/errorResponse/">
    <code>409</code>
    <status>Conflict</status>
    <message>Request can not be processed due to issue in the
request message</message>
    <errors>
        <ns2:error xmlns:ns2="http://www.avaya.com/error/">
            <message>Request can not be processed due to issue in
the request message</message>
            <uniqueId>test1@avaya.com</uniqueId>
            <details>
                <message>
                    <reasonCode>UPM_1102</reasonCode>
                    <reasonText>User with this loginname already
exists ( maybe the user you are adding has been soft deleted )
</reasonText>
                </message>
            </details>
        </ns2:error>
        <ns2:error xmlns:ns2="http://www.avaya.com/error/">
            <message>Request can not be processed due to issue in
the request message</message>
            <uniqueId>test2@avaya.com</uniqueId>
            <details>
                <message>
                    <reasonCode>UPM_1102</reasonCode>
                    <reasonText>User with this loginname already
exists ( maybe the user you are adding has been soft deleted )
</reasonText>
                </message>
            </details>
        </ns2:error>
    </errors>
```

Code snippet

The following is the Java client sample code to create a user with mandatory attributes using RESTful web services. The post request takes two FORM parameters: **entityType** and **entityXMLData**.

createsample.xml – The following is a sample xml file for Create User

```
<?xml version="1.0" encoding="UTF-8"?>
<tns:users xmlns:tns="http://xml.avaya.com/schema/import"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xml.avaya.com/schema/importuserimport
.xsd">     <tns:user>
           <authenticationType>Basic</authenticationType>
           <givenName>John</givenName>
           <loginName>jmiller@avaya.com</loginName>
           <surname>Miller</surname>
           <userPassword>mypassword</userPassword>
      </tns:user>
</tns:users>
```

Sample code – The following is a sample Java code to create a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
  private String uri = "https://example.com/web/mgmtwebservice";
…
      final InputStream inputStream = getClass().getClassLoader()
                        .getResourceAsStream("createsample.xml");
      final Reader reader = new InputStreamReader(inputStream);
      final BufferedReader br = new BufferedReader(reader);
      final StringBuffer line = new StringBuffer();
      String st = "";
      while ((st = br.readLine()) != null)
      {
            line.append(st + "\n");
      }
      Final String entityXmlData = line.toString();
      final Form form = new Form();
      form.add("entityType", "user");
      form.add("entityXMLData", entityXmlData);
      form.add("operationName", "create"); //optional

      String result = client.resource (uri).post(String.class,
   form);
…
```

The following statement adds **`entityType`**, **`entityXMLData`** and **`operationName`** parameter to the form class object respectively.

```
form.add("entityType", "user");
form.add("entityXMLData", entityXmlData);
form.add("operationName", "create"); //optional
```

`form` object is further passed to the post request and returns a list of status messages.

```
String result = client.resource (uri).post (String.class, form);
```

Note: For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

Update user

The update method updates one or more user profile in the System Manager with the given user data in XML format.

```
String update (String entityType, String entityXMLData, String
updateMode
```

The **`updateMode`** parameter can be **`merge`** or **`replace`**. With the replace option, you can replace a user and data associated with the user. With the merge option, you can add data in collection value and update the single value user data. For example, add a contact to a contact list and update the last name of the user.

Web services client can retrieve a list of users using lookup API (see Retrieve users with minimal user data based on search criteria), modify user data, and then invoke update API to update user data.

The Update User operation has the following components:

HTTP method	PUT
URL	*https://host/web/v2/mgmtwebservice* *host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	Refer to Request Parameters section below.
Request Parameters	None.
Response	Refer to Response section below.

Request parameters

The Update User Call request takes the following parameters:

Name of the parameter	Data type	Type	Description
entityType	String	Form	Specifies an entity name. Expected value: **user**
entityXMLData	String	Form	Specifies User XML data conforming to the specified XSD. See Appendix A
updateMode	String	Form	Specifies the type of update that you must perform. Possible values: **merge** or **replace**

Response

Successful response does not contain any data, but the HTTP status code of the response will be 204.

In case of an error, an appropriate HTTP status code is sent back along with a response XML. The Response XML conforms to the XSD described in Appendix B

Responses can contain below status codes for different scenarios.

HTTP	HTTP Status	Description

Status Code	Message	
204	NA	User updated successfully. No content is sent back in the response.
400	Bad Request	All validation exceptions.
409	Conflict	User with same login name already exists.
500	Internal Server Error	Any errors other than validation exception or username conflict. One may also get a 500 response code for cases where multiple users are updated using a single request and each user update in the request results with different HTTP status code. Please refer to example below.

For example,

Successful response

```
HTTP/1.1 204
Connection: Closed
No Content
```

Error Response

```
HTTP/1.1 400
Content-Type: application/xml
Connection: Closed
<ns1:errorResponse
xmlns:ns1="http://www.avaya.com/errorResponse/">
    <code>400</code>
    <status>Bad Request</status>
    <message>Invalid request data</message>
      <errors>
        <ns2:error xmlns:ns2="http://www.avaya.com/error/>
        <message>Invalid request data</message>
        <uniqueId>test123@avaya.com</uniqueId>
        <details>
            <message>
                <name>CsUser.commPassword</name>
                <reasonCode>RLUPM0154</reasonCode>
                <reasonText>Communication Profile Password is
required for the SIP handle assigned</reasonText>
            </message>
        </details>
    </ns2:error>
</errors>
</ns1:errorResponse>

HTTP/1.1 409
Content-Type: application/xml
Connection: Closed
<ns1:errorResponse
xmlns:ns1="http://www.avaya.com/errorResponse/">
    <code>409</code>
    <status>Conflict</status>
    <message>Request can not be processed due to issue in the
request message</message>
    <errors>
        <ns2:error xmlns:ns2="http://www.avaya.com/error/>
            <message>Request can not be processed due to issue in
the request message</message>
            <details>
                <message>
                    <reasonCode>BI_2008</reasonCode>
                    <reasonText>No matching record found for Login
Name test@avaya.com</reasonText>
                </message>
            </details>
```

Code snippet

The following is the Java client sample code to update a user with mandatory attributes using RESTful web services. The put request takes three FORM parameters: **`entityType, entityXMLData`** and **`updateMode`**.

updatesample.xml – The following is a sample xml file for Update User

```
<?xml version="1.0" encoding="UTF-8"?>
<delta:deltaUserList
xmlns:delta="http://xml.avaya.com/schema/deltaImport"
      xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://xml.avaya.com/schema/deltaImport
userdeltaimport.xsd ">
      <delta:userDelta>
            <loginName>jmiller@avaya.com</loginName>
            <middleName>Merged middle name</middleName>
      </delta:userDelta>
</delta:deltaUserList>
```

Sample code – The following is a sample code snippet to update a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
private String uri = "https://example.com/web/mgmtwebservice";
…
     final InputStream inputStream = getClass().getClassLoader()
                        .getResourceAsStream("updatesample.xml");
      final Reader reader = new InputStreamReader(inputStream);
      final BufferedReader br = new BufferedReader(reader);
      final StringBuffer line = new StringBuffer();
      String st = "";
      while ((st = br.readLine()) != null)
      {
            line.append(st + "\n");
      }
      Final String entityXmlData = line.toString();
      final Form form = new Form();
      form.add("entityType", "user");
      form.add("entityXMLData", entityXmlData);
      form.add("updateMode", "merge");
      String result = client.resource(uri).post(String.class,
form);
…
```

The following statement adds **`entityType`**, **`entityXMLData`**, and **`updateMode`** parameters to the form class object respectively.

```
form.add ("entityType", "user");
form.add ("entityXMLData", entityXmlData);
form.add ("updateMode", "merge");
```

`form` object is further passed to the put request and returns a list of status messages.

```
String result = client.resource (uri).put (String.class,
form);
```

Note: For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

Retrieve users with minimal user data based on search criteria

The retrieve method enables search for one or more user profiles within System Manager with the given search criteria.

`String lookup (String entityType, String searchCriteria)`

The search criteria are based on the following user attributes: First Name, Last Name, Display Name, and Login name.

The lookup method has been tested and recommended to fetch 500 users at a time.

The Retrieve users operation has the following components:

HTTP method	GET
URL	*https://host/web/v2/mgmtwebservice* *host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	Refer to Request Parameters section below.
Response	Refer to Response section below.

Request parameters

Name of the parameter	Data type	Type	Description
entityType	String	Query	Specifies an entity name. Expected value: **user**
searchCriteria	String	Query	Specifies the search criteria for listing the user

			profiles. These criteria should conform to Search Criteria XSD.

Response

The user profile search result includes one or more user profile data in XML format. The minimal user data in XML consists of First Name, Middle Name, Last Name, Display Name, source, and Login Name.

In case of an error, an appropriate HTTP status code is sent back along with a response XML. The Response XML conforms to the XSD described in Appendix B

Responses can contain below status codes for different scenarios.

HTTP Status Code	HTTP Status Message	Description
200	NA	User retrieved successfully
400	Bad Request	All validation exceptions.
404	Not Found	User not found
500	Internal Server Error	Any errors other than validation exception or username conflict.

For example,

`Successful response`

```
HTTP/1.1 200
Connection: Closed
</errorResponse><?xml version="1.0" encoding="UTF-8"
standalone="yes"?>
<tns:users xsi:schemaLocation="http://xml.avaya.com/schema/import
userimport.xsd" xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns6="http://xml.avaya.com/schema/import_scopia"
xmlns:ns5="http://xml.avaya.com/schema/presence"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm"
xmlns:ns7="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns13="http://xml.avaya.com/schema/import_ce"
xmlns:ns9="http://xml.avaya.com/schema/import_csm_b5800"
xmlns:ns12="http://xml.avaya.com/schema/import_mmcs"
xmlns:ns11="http://xml.avaya.com/schema/import_csm_mm"
xmlns:ns10="http://xml.avaya.com/schema/import_mem_officelinx"
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns14="http://xml.avaya.com/schema/deltaImport">
    <tns:user>
        <authenticationType>basic</authenticationType>
        <displayName>test, test</displayName>
        <displayNameAscii>test, test</displayNameAscii>
        <isDuplicatedLoginAllowed>false</isDuplicatedLoginAllowed>
        <isEnabled>true</isEnabled>
        <isVirtualUser>false</isVirtualUser>
        <givenName>test</givenName>
        <givenNameAscii>test</givenNameAscii>
        <loginName>test@avaya.com</loginName>
        <preferredLanguage>en_US</preferredLanguage>
        <source>local.ws</source>
        <sourceUserKey>none</sourceUserKey>
                <name>list-test_avaya.com</name>
                <isPublic>false</isPublic>
                <contactListType>general</contactListType>
            </contactList>
        </ownedContactLists>
        <commProfileSet>
            <commProfileSetName>Primary</commProfileSetName>
            <isPrimary>true</isPrimary>
            <commProfileList>
                <commProfile xsi:type="ns6:ScopiaCommProfileType"
```

```
xmlns:ns6="http://xml.avaya.com/schema/import_scopia">

<commProfileType>scopiaProfile</commProfileType>
                    <ns6:scopiaUserId>36</ns6:scopiaUserId>
                    <ns6:password>12345</ns6:password>
                </commProfile>
            </commProfileList>
        </commProfileSet>
    </tns:user>
</tns:users>
```

Error Response

```
HTTP/1.1 404
Content-Type: application/xml
Connection: Closed
<ns1:errorResponse xmlns:ns1="http://www.avaya.com/errorResponse/
">
    <code>404</code>
    <status>Not Found</status>
    <message>Resource trying to request for is not available</mes
sage>
    <errors>
        <ns2:error xmlns:ns2="http://www.avaya.com/error/">
            <message>Resource trying to request for is not availa
ble</message>
            <details>
                <message>
                    <reasonText>Records not found for the specifi
ed criteria</reasonText>
                </message>
            </details>
        </ns2:error>
    </errors>
</ns1:errorResponse>
```

Sample Resultant stream content for **`lookup`**:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users
xsi:schemaLocation="http://xml.avaya.com/schema/import
userimport.xsd" xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_mm"
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns9="http://xml.avaya.com/schema/deltaImport"
xmlns:ns5="http://xml.avaya.com/schema/import_csm_abg"
xmlns:ns6="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns7="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm">
    <tns:user>
        <givenName>John</givenName>
        <loginName>jmiller@avaya.com</loginName>
        <source>TC_0101_1.xml</source>
        <sourceUserKey>none</sourceUserKey>
```

```
        <surname>Miller</surname>
        <userName>jmiller</userName>
    </tns:user>
</tns:users>
```

Code snippet

Sample code – The following is a sample code snippet to retrieve a list of users using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
private String uri = "https://example.com/web/mgmtwebservice";
…
     final InputStream inputStream = getClass().getClassLoader()
                .getResourceAsStream("searchcriteria.xml");
      final Reader reader = new InputStreamReader(inputStream);
      final BufferedReader br = new BufferedReader(reader);
      final StringBuffer line = new StringBuffer();
      String st = "";
      while ((st = br.readLine()) != null)
      {
            line.append(st + "\n");
      }
      String searchCriteria = line.toString();
      searchCriteria = searchCriteria.replaceAll("&", "&amp;")
            .replaceAll("<", "&lt;").replaceAll(">", "&gt;")
      .replaceAll("\"", "&quot;").replaceAll("'", "&apos;");

      String result =
      client.resource(uri).queryParam("entityType",
                "user").queryParam("searchCriteria",
            searchCriteria)
                get(String.class);
…
```

The last statement in the sample code to establish an SSL connection and authenticate the user, QUERY parameter **`entityType`** and **`searchCriteria`** is passed along with the GET request to the server.

Note: For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

Search criteria

You can search for users based on the following criteria:

User data:

User attributes	Description

First Name	Identifies the first name of the user. Searches are based on the first name. You can refine the search by specifying conditions such as equals, contains, starts with, or ends with the value of the user First Name. For example, you can specify the search criteria "First Name equals Joe" by selecting the equals condition and entering Joe in the First Name field.
Last Name	Identifies the last name of the user. Searches are based on the last name. You can refine the search by specifying conditions such as equals, contains, starts with, or ends with the value of the user Last Name. For example, you can specify the search criteria "Last Name equals Miller" by selecting the equals condition and entering Miller in the first name field.
Display Name	Identifies the display name of the user. Searches are based on Display name. You can refine the search by specifying conditions such as equals, contains, starts with, or ends with the value of the user Display Name. For example, you can specify the search criteria "Display Name equals Joe Miller" by selecting the equals condition and entering Joe Miller in the display name field.
Login Name	Identifies the log-in name of the user. You can refine the search by specifying conditions such as equals, contains, starts with, or ends with the value of the user Login Name. For example, you can specify the search criteria "Login Name equals jmiller@abc.com" by selecting the equals condition and entering jmiller@abc.com in the login name field.
Start Index	Sorts the database user record ID in ascending order.
Offset Index	Identifies the number of records that will be retrieved starting from the specified Start Index.

Operators:

Operators	**Description**
Equals	Refine the search criteria by using the equals operator. Equals imply having the same value as another.
StartsWith	Implies the value will preclude the same value as another.
endsWith	Implies the value will end with the same value as another.

Contains	Refine the search criteria by using the contains operator. Contains implies including or comprising the same value as another.

Restrictions:

Restrictions	Description
And	A Boolean operator that gives the value one only if all the operands are one and otherwise has a value of zero.
Or	A Boolean operator that gives the value one if at least one operand (or input) has a value of one and otherwise has a value of zero.

searchcriteria.xml – The following is a sample XML file to specify criteria to retrieve a list of users

```
  <?xml version="1.0" encoding="UTF-8"?>
  <tns:management-webservice-criteria
xmlns:tns="http://www.avaya.com/mgmt-web-criteria/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.avaya.com/mgmt-web-criteria/
mgmt-web-criteria.xsd">
        <criterialist>
                        <criteria>
            <propertyname>startindex</propertyname>
            <operator>equals</operator>
            <value>0</value>
            <restriction>and</restriction>
          </criteria>
          <criteria>
            <propertyname>offsetindex</propertyname>
            <operator>equals</operator>
            <value>10</value>
            <restriction>and</restriction>
          </criteria>
        </criterialist>
</tns:management-webservice-criteria>
```

Get user profile

The get user method enables retrieving a user profile with complete user data based on the login name. Starting version 8.1.12 of System Manager you can retrieve selective portions of user data using userAttributeCategory query parameter in the API.

`String find (String entityType, String uniqueID)`

The Get user operation has the following components:

HTTP method	GET
URL	*https://host/web/v2/mgmtwebservice* *host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	Refer to Request Parameters section below.
Response	Refer to Response section below.

Request parameters

Name of the parameter	Data type	Type	Description
entityType	String	Path	Specifies an entity name. Expected value: **user**
uniqueID	String	Path	Specifies the User login name.
userAttributeCategory	List<String>	Query	Specifies the user attribute category(s). This query parameter is optional, if you do not provide this parameter in the API then the complete data of a user will be fetched. Expected values are one or more of the following categories as comma separated values : **BASIC, COMMPROFILES, CONTACTLISTS, ADDRESS, MEMBERSHIPS.** For e.g. if you want to fetch the comm profiles and the memberships of a user then you can provide values like userAttributeCategory=COMMPROFILES, MEMBERSHIPS. The BASIC data of a user will always be fetched along with any of the user attribute that you are passing as a query parameter. See the description of each user attribute category here.

User Attribute Category

Name of the userAttributeCategory	Description
BASIC	Fetch the basic information associated with the user.
COMMPROFILES	Fetch the basic information, communication profiles and handles associated with the user.
CONTACTLISTS	Fetch the basic information and contacts associated with the user.

ADDRESS	Fetch the basic information and addresses associated with the user.
MEMBERSHIPS	Fetch the basic information, groups and roles associated with the user.

Response

The get user profile search result includes one user profile data in XML format. The XML contains complete user data and no filter is applied.

The response of the request consists of user profile details (all user attributes) based on the **uniqueId** sent as an input parameter.

Response when userAttributeCategory is BASIC

The get user profile search result includes one user profile data in XML format. The XML contains BASIC user data.

The response of the request consists of user profile details (only BASIC attributes) based on the **uniqueId** and **userAttributeCategory** sent as an input parameter.

For example,

Sample response

```
HTTP/1.1 200
Connection: Closed

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users xsi:schemaLocation="http://xml.avaya.com/schema/import userimport.xsd"
xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns6="http://xml.avaya.com/schema/import_scopia"
xmlns:ns5="http://xml.avaya.com/schema/presence"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm"
xmlns:ns7="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns13="http://xml.avaya.com/schema/import_ce"
xmlns:ns9="http://xml.avaya.com/schema/import_csm_b5800"
xmlns:ns12="http://xml.avaya.com/schema/import_mmcs"
xmlns:ns11="http://xml.avaya.com/schema/import_csm_mm"
xmlns:ns10="http://xml.avaya.com/schema/import_mem_officelinx"
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns14="http://xml.avaya.com/schema/deltaImport">
   <tns:user>
      <authenticationType>basic</authenticationType>
      <displayName>load24, test</displayName>
```

```
      <displayNameAscii>load24, test</displayNameAscii>
      <isDuplicatedLoginAllowed>false</isDuplicatedLoginAllowed>
      <isEnabled>true</isEnabled>
      <isVirtualUser>false</isVirtualUser>
      <givenName>test</givenName>
      <givenNameAscii>test</givenNameAscii>
      <loginName>loadtest24@avaya.com</loginName>
      <middleName>loadtest24 update test middle name</middleName>
      <preferredLanguage>en_US</preferredLanguage>
      <source>local</source>
      <sourceUserKey>none</sourceUserKey>
      <status>provisioned</status>
      <surname>load24</surname>
      <surnameAscii>load24</surnameAscii>
      <userName>loadtest24</userName>
      <userPassword></userPassword>
      <commPassword></commPassword>
      <localizedNames>
         <localizedName>
            <locale>English</locale>
            <name>load24</name>
         </localizedName>
      </localizedNames>
   </tns:user>
</tns:users>
```



```
HTTP/1.1 200
Connection: Closed

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users xsi:schemaLocation="http://xml.avaya.com/schema/import userimport.xsd"
xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns6="http://xml.avaya.com/schema/import_scopia"
xmlns:ns5="http://xml.avaya.com/schema/presence"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm"
xmlns:ns7="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns13="http://xml.avaya.com/schema/import_ce"
xmlns:ns9="http://xml.avaya.com/schema/import_csm_b5800"
xmlns:ns12="http://xml.avaya.com/schema/import_mmcs"
xmlns:ns11="http://xml.avaya.com/schema/import_csm_mm"
xmlns:ns10="http://xml.avaya.com/schema/import_mem_officelinx"
```

```
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns14="http://xml.avaya.com/schema/deltaImport">
  <tns:user>
    <authenticationType>basic</authenticationType>
    <displayName>load24, test</displayName>
    <displayNameAscii>load24, test</displayNameAscii>
    <isDuplicatedLoginAllowed>false</isDuplicatedLoginAllowed>
    <isEnabled>true</isEnabled>
    <isVirtualUser>false</isVirtualUser>
    <givenName>test</givenName>
    <givenNameAscii>test</givenNameAscii>
    <loginName>loadtest24@avaya.com</loginName>
    <middleName>loadtest24 update test middle name</middleName>
    <preferredLanguage>en_US</preferredLanguage>
    <source>local</source>
    <sourceUserKey>none</sourceUserKey>
    <status>provisioned</status>
    <surname>load24</surname>
    <surnameAscii>load24</surnameAscii>
    <userName>loadtest24</userName>
    <userPassword></userPassword>
    <commPassword></commPassword>
    <localizedNames>
      <localizedName>
        <locale>English</locale>
        <name>load24</name>
      </localizedName>
    </localizedNames>
    <commProfileSet>
      <commProfileSetName>Primary</commProfileSetName>
      <isPrimary>true</isPrimary>
      <handleList>
        <handle>
          <handleName>load24</handleName>
          <handleType>sip</handleType>
          <handleSubType>username</handleSubType>
          <domainName>gssmgr.com</domainName>
        </handle>
      </handleList>
      <commProfileList>
        <commProfile xsi:type="ns7:SessionManagerCommProfXML"
xmlns:ns7="http://xml.avaya.com/schema/import_sessionmanager">
          <commProfileType>SessionManager</commProfileType>
```

```
                <ns7:primarySM>SM_217</ns7:primarySM>
                <ns7:homeLocation>Pune</ns7:homeLocation>
                <ns7:maxSimultaneousDevices>1</ns7:maxSimultaneousDevices>

<ns7:blockNewRegistrationWhenMaxActive>false</ns7:blockNewRegistrationWhenM
axActive>
                <ns7:enabledisablecalllog>false</ns7:enabledisablecalllog>
                <ns7:policyType>fixed</ns7:policyType>
            </commProfile>
            <commProfile xsi:type="ns8:xmlStationProfile"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm">
                <commProfileType>CM</commProfileType>
                <ns8:cmName>CM_249</ns8:cmName>
                <ns8:prefHandleId></ns8:prefHandleId>
                <ns8:useExistingExtension>false</ns8:useExistingExtension>
                <ns8:extension>1229</ns8:extension>
                <ns8:setType>1408</ns8:setType>
                <ns8:securityCode></ns8:securityCode>
                <ns8:port>X</ns8:port>
                <ns8:deleteOnUnassign>true</ns8:deleteOnUnassign>
                <ns8:overRideEndpointName>true</ns8:overRideEndpointName>
                <ns8:dualRegistration>false</ns8:dualRegistration>
                <ns8:enhCallrInfodisplay>false</ns8:enhCallrInfodisplay>
                <ns8:lockMessages>false</ns8:lockMessages>
                <ns8:tn>1</ns8:tn>
                <ns8:cor>1</ns8:cor>
                <ns8:cos>1</ns8:cos>
                <ns8:tests>false</ns8:tests>
                <ns8:dataModule>false</ns8:dataModule>
                <ns8:speakerphone>2-way</ns8:speakerphone>
                <ns8:displayLanguage>english</ns8:displayLanguage>
                <ns8:personalizedRingingPattern>1</ns8:personalizedRingingPattern>
                <ns8:messageLampExt>1229</ns8:messageLampExt>
                <ns8:muteButtonEnabled>true</ns8:muteButtonEnabled>
                <ns8:ipSoftphone>false</ns8:ipSoftphone>
                <ns8:survivableCOR>internal</ns8:survivableCOR>
                <ns8:survivableTrunkDest>true</ns8:survivableTrunkDest>
                <ns8:voiceMailNumber></ns8:voiceMailNumber>
                <ns8:offPremisesStation>false</ns8:offPremisesStation>
                <ns8:displayModule>false</ns8:displayModule>
                <ns8:remoteOfficePhone>false</ns8:remoteOfficePhone>
                <ns8:lwcReception>spe</ns8:lwcReception>
                <ns8:lwcActivation>true</ns8:lwcActivation>
```

```
<ns8:lwcLogExternalCalls>false</ns8:lwcLogExternalCalls>
<ns8:cdrPrivacy>false</ns8:cdrPrivacy>
<ns8:redirectNotification>true</ns8:redirectNotification>
<ns8:perButtonRingControl>false</ns8:perButtonRingControl>
<ns8:bridgedCallAlerting>false</ns8:bridgedCallAlerting>
<ns8:bridgedIdleLinePreference>false</ns8:bridgedIdleLinePreference>

<ns8:confTransOnPrimaryAppearance>false</ns8:confTransOnPrimaryAppearance>
<ns8:customizableLabels>true</ns8:customizableLabels>
<ns8:expansionModule>false</ns8:expansionModule>
<ns8:ipVideoSoftphone>false</ns8:ipVideoSoftphone>
<ns8:activeStationRinging>single</ns8:activeStationRinging>
<ns8:switchhookFlash>false</ns8:switchhookFlash>
<ns8:ignoreRotaryDigits>false</ns8:ignoreRotaryDigits>
<ns8:h320Conversion>false</ns8:h320Conversion>
<ns8:serviceLinkMode>as-needed</ns8:serviceLinkMode>
<ns8:multimediaMode>basic</ns8:multimediaMode>
<ns8:mwiServedUserType></ns8:mwiServedUserType>
<ns8:emergencyLocationExt>1229</ns8:emergencyLocationExt>
<ns8:alwaysUse>false</ns8:alwaysUse>
<ns8:precedenceCallWaiting>false</ns8:precedenceCallWaiting>

<ns8:autoSelectAnyIdleAppearance>false</ns8:autoSelectAnyIdleAppearance>
<ns8:coverageMsgRetrieval>true</ns8:coverageMsgRetrieval>
<ns8:autoAnswer>none</ns8:autoAnswer>
<ns8:dataRestriction>false</ns8:dataRestriction>
<ns8:idleAppearancePreference>false</ns8:idleAppearancePreference>
<ns8:callWaitingIndication>false</ns8:callWaitingIndication>
<ns8:attCallWaitingIndication>false</ns8:attCallWaitingIndication>
<ns8:distinctiveAudibleAlert>false</ns8:distinctiveAudibleAlert>
<ns8:restrictLastAppearance>true</ns8:restrictLastAppearance>
<ns8:adjunctSupervision>false</ns8:adjunctSupervision>

<ns8:busyAutoCallbackWithoutFlash>false</ns8:busyAutoCallbackWithoutFlash>
<ns8:audibleMessageWaiting>false</ns8:audibleMessageWaiting>
<ns8:imsFeatureSequencing>false</ns8:imsFeatureSequencing>
<ns8:displayClientRedirection>false</ns8:displayClientRedirection>

<ns8:selectLastUsedAppearance>false</ns8:selectLastUsedAppearance>
<ns8:coverageAfterForwarding>s</ns8:coverageAfterForwarding>
<ns8:directIpIpAudioConnections>true</ns8:directIpIpAudioConnections>
<ns8:ipAudioHairpinning>false</ns8:ipAudioHairpinning>
<ns8:stationSiteData>
```

```
                <ns8:headset>false</ns8:headset>
                <ns8:speaker>false</ns8:speaker>
                <ns8:mounting>d</ns8:mounting>
                <ns8:cordLength>0</ns8:cordLength>
            </ns8:stationSiteData>
            <ns8:buttons>
                <ns8:number>1</ns8:number>
                <ns8:type>call-appr</ns8:type>
                <ns8:data1></ns8:data1>
                <ns8:data2></ns8:data2>
                <ns8:data3></ns8:data3>
                <ns8:data4></ns8:data4>
                <ns8:data5></ns8:data5>
                <ns8:data6></ns8:data6>
            </ns8:buttons>
            <ns8:buttons>
                <ns8:number>2</ns8:number>
                <ns8:type>call-appr</ns8:type>
                <ns8:data1></ns8:data1>
                <ns8:data2></ns8:data2>
                <ns8:data3></ns8:data3>
                <ns8:data4></ns8:data4>
                <ns8:data5></ns8:data5>
                <ns8:data6></ns8:data6>
            </ns8:buttons>
            <ns8:buttons>
                <ns8:number>3</ns8:number>
                <ns8:type>call-appr</ns8:type>
                <ns8:data1></ns8:data1>
                <ns8:data2></ns8:data2>
                <ns8:data3></ns8:data3>
                <ns8:data4></ns8:data4>
                <ns8:data5></ns8:data5>
                <ns8:data6></ns8:data6>
            </ns8:buttons>
            <ns8:nativeName/>
            <ns8:unconditionalInternalDest></ns8:unconditionalInternalDest>
            <ns8:unconditionalInternalActive>false</ns8:unconditionalInternalActive>
            <ns8:unconditionalExternalDest></ns8:unconditionalExternalDest>

<ns8:unconditionalExternalActive>false</ns8:unconditionalExternalActive>
            <ns8:busyInternalDest></ns8:busyInternalDest>
            <ns8:busyInternalActive>false</ns8:busyInternalActive>
```

```
                <ns8:busyExternalDest></ns8:busyExternalDest>
                <ns8:busyExternalActive>false</ns8:busyExternalActive>
                <ns8:noReplyInternalDest></ns8:noReplyInternalDest>
                <ns8:noReplyInternalActive>false</ns8:noReplyInternalActive>
                <ns8:noReplyExternalDest></ns8:noReplyExternalDest>
                <ns8:noReplyExternalActive>false</ns8:noReplyExternalActive>
                <ns8:sacCfOverride>n</ns8:sacCfOverride>
                <ns8:lossGroup>2</ns8:lossGroup>
                <ns8:timeOfDayLockTable></ns8:timeOfDayLockTable>
                <ns8:emuLoginAllowed>false</ns8:emuLoginAllowed>
                <ns8:ec500State>enabled</ns8:ec500State>
                <ns8:muteOnOffHookInSCMode>false</ns8:muteOnOffHookInSCMode>
                <ns8:type3pccEnabled>None</ns8:type3pccEnabled>
                <ns8:calculateRoutePattern>false</ns8:calculateRoutePattern>

<ns8:enableReachStaDomainControl>s</ns8:enableReachStaDomainControl>
                <ns8:multimediaEarlyAnswer>false</ns8:multimediaEarlyAnswer>
                <ns8:bridgedApprOrigRestr>false</ns8:bridgedApprOrigRestr>
                <ns8:callApprDispFormat>disp-param-default</ns8:callApprDispFormat>
                <ns8:ipPhoneGroupId></ns8:ipPhoneGroupId>
                <ns8:xid>false</ns8:xid>
                <ns8:stepClearing>false</ns8:stepClearing>
                <ns8:fixedTei>false</ns8:fixedTei>
                <ns8:endptInit>false</ns8:endptInit>
                <ns8:isShortCallingPartyDisplay>true</ns8:isShortCallingPartyDisplay>
                <ns8:displayCallerId>true</ns8:displayCallerId>

<ns8:callerIdMsgWaitingIndication>false</ns8:callerIdMsgWaitingIndication>
                <ns8:recallRotaryDigit>false</ns8:recallRotaryDigit>

<ns8:bridgingToneForThisExtension>n</ns8:bridgingToneForThisExtension>
                <ns8:terminalNumberPart1>0</ns8:terminalNumberPart1>
                <ns8:terminalNumberPart2>0</ns8:terminalNumberPart2>
                <ns8:terminalNumberPart3>0</ns8:terminalNumberPart3>
                <ns8:terminalNumberPart4>0</ns8:terminalNumberPart4>
                <ns8:systemId></ns8:systemId>
                <ns8:features></ns8:features>
                <ns8:features2></ns8:features2>
                <ns8:attendant>false</ns8:attendant>
                <ns8:ipHoteling>false</ns8:ipHoteling>
            </commProfile>
        </commProfileList>
    </commProfileSet>
```

```
    </tns:user>
</tns:users>
```

Response when userAttributeCategory is CONTACTLISTS

The get user profile search result includes one user profile data in XML format. The XML contains BASIC user data and the contacts associated with the user.

The response of the request consists of user profile details (only BASIC and CONTACTLISTS attributes) based on the **uniqueId** and **userAttributeCategory** sent as an input parameter.

For example,

Sample response

```
HTTP/1.1 200
Connection: Closed

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users xsi:schemaLocation="http://xml.avaya.com/schema/import userimport.xsd"
xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns6="http://xml.avaya.com/schema/import_scopia"
xmlns:ns5="http://xml.avaya.com/schema/presence"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm"
xmlns:ns7="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns13="http://xml.avaya.com/schema/import_ce"
xmlns:ns9="http://xml.avaya.com/schema/import_csm_b5800"
xmlns:ns12="http://xml.avaya.com/schema/import_mmcs"
xmlns:ns11="http://xml.avaya.com/schema/import_csm_mm"
xmlns:ns10="http://xml.avaya.com/schema/import_mem_officelinx"
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns14="http://xml.avaya.com/schema/deltaImport">
   <tns:user>
      <authenticationType>basic</authenticationType>
      <displayName>load24, test</displayName>
      <displayNameAscii>load24, test</displayNameAscii>
      <isDuplicatedLoginAllowed>false</isDuplicatedLoginAllowed>
      <isEnabled>true</isEnabled>
      <isVirtualUser>false</isVirtualUser>
      <givenName>test</givenName>
      <givenNameAscii>test</givenNameAscii>
      <loginName>loadtest24@avaya.com</loginName>
      <middleName>loadtest24 update test middle name</middleName>
      <preferredLanguage>en_US</preferredLanguage>
      <source>local</source>
      <sourceUserKey>none</sourceUserKey>
      <status>provisioned</status>
      <surname>load24</surname>
      <surnameAscii>load24</surnameAscii>
      <userName>loadtest24</userName>
      <userPassword></userPassword>
      <commPassword></commPassword>
      <localizedNames>
         <localizedName>
            <locale>English</locale>
            <name>load24</name>
         </localizedName>
```

Response when userAttributeCategory is ADDRESS

The get user profile search result includes one user profile data in XML format. The XML contains BASIC user data and the addresses associated with the user.

The response of the request consists of user profile details (only BASIC and ADDRESS attributes) based on the **uniqueId** and **userAttributeCategory** sent as an input parameter.

For example,

Sample response

```
HTTP/1.1 200
Connection: Closed

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users xsi:schemaLocation="http://xml.avaya.com/schema/import userimport.xsd"
xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns6="http://xml.avaya.com/schema/import_scopia"
xmlns:ns5="http://xml.avaya.com/schema/presence"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm"
xmlns:ns7="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns13="http://xml.avaya.com/schema/import_ce"
xmlns:ns9="http://xml.avaya.com/schema/import_csm_b5800"
xmlns:ns12="http://xml.avaya.com/schema/import_mmcs"
xmlns:ns11="http://xml.avaya.com/schema/import_csm_mm"
xmlns:ns10="http://xml.avaya.com/schema/import_mem_officelinx"
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns14="http://xml.avaya.com/schema/deltaImport">
   <tns:user>
      <authenticationType>basic</authenticationType>
      <displayName>load24, test</displayName>
      <displayNameAscii>load24, test</displayNameAscii>
      <isDuplicatedLoginAllowed>false</isDuplicatedLoginAllowed>
      <isEnabled>true</isEnabled>
      <isVirtualUser>false</isVirtualUser>
      <givenName>test</givenName>
      <givenNameAscii>test</givenNameAscii>
      <loginName>loadtest24@avaya.com</loginName>
      <middleName>loadtest24 update test middle name</middleName>
      <preferredLanguage>en_US</preferredLanguage>
      <source>local</source>
```

```
      <sourceUserKey>none</sourceUserKey>
      <status>provisioned</status>
      <surname>load24</surname>
      <surnameAscii>load24</surnameAscii>
      <userName>loadtest24</userName>
      <userPassword></userPassword>
      <commPassword></commPassword>
      <localizedNames>
         <localizedName>
            <locale>English</locale>
            <name>load24</name>
         </localizedName>
      </localizedNames>
      <address>
         <addressType>office</addressType>
         <name>Pune</name>
         <building></building>
         <localityName></localityName>
         <postalCode></postalCode>
         <room></room>
         <stateOrProvince></stateOrProvince>
         <country></country>
         <street></street>
         <businessphone></businessphone>
         <otherbusinessphone></otherbusinessphone>
         <fax></fax>
         <homephone></homephone>
         <otherhomephone></otherhomephone>
         <mobilephone></mobilephone>
         <othermobilephone></othermobilephone>
         <pager></pager>
         <pager2></pager2>
         <isPrivate>true</isPrivate>
      </address>
   </tns:user>
</tns:users>
```


Response when userAttributeCategory is MEMBERSHIPS

The get user profile search result includes one user profile data in XML format. The XML contains BASIC user data and the groups and roles associated with the user.

The response of the request consists of user profile details (only BASIC and MEMBERSHIPS attributes) based on the **uniqueId** and **userAttributeCategory** sent as an input parameter.

For example,

`Sample response`

```
HTTP/1.1 200
Connection: Closed

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users xsi:schemaLocation="http://xml.avaya.com/schema/import userimport.xsd"
xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns6="http://xml.avaya.com/schema/import_scopia"
xmlns:ns5="http://xml.avaya.com/schema/presence"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm"
xmlns:ns7="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns13="http://xml.avaya.com/schema/import_ce"
xmlns:ns9="http://xml.avaya.com/schema/import_csm_b5800"
xmlns:ns12="http://xml.avaya.com/schema/import_mmcs"
xmlns:ns11="http://xml.avaya.com/schema/import_csm_mm"
xmlns:ns10="http://xml.avaya.com/schema/import_mem_officelinx"
    <isEnabled>true</isEnabled>
    <isVirtualUser>false</isVirtualUser>
    <givenName>test</givenName>
    <givenNameAscii>test</givenNameAscii>
    <loginName>loadtest24@avaya.com</loginName>
    <middleName>loadtest24 update test middle name</middleName>
    <preferredLanguage>en_US</preferredLanguage>
    <source>local</source>
    <sourceUserKey>none</sourceUserKey>
    <status>provisioned</status>
    <surname>load24</surname>
    <surnameAscii>load24</surnameAscii>
    <userName>loadtest24</userName>
    <userPassword></userPassword>
    <commPassword></commPassword>
    <roles>
      <role>26f413c0-4a81-4451-83e8-f59382cc6d81</role>
      <role>End-User</role>
      <role>Collaboration.20Environment.20Auditor</role>
    </roles>
    <groups>
      <group>
        <name>TestGroup</name>
      </group>
    </groups>
    <localizedNames>
```

```
        <localizedName>
          <locale>English</locale>
          <name>load24</name>
        </localizedName>
      </localizedNames>
    </tns:user>
</tns:users>
```


To get a combination of one or more categories do the following:

For example,

when userAttributeCategory= COMMPROFILES, CONTACTLISTS, MEMBERSHIPS

Successful response


```
HTTP/1.1 200
Connection: Closed

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users xsi:schemaLocation="http://xml.avaya.com/schema/import userimport.xsd"
xmlns:ns4="http://xml.avaya.com/schema/import1" xmlns:ns3="http://xml.avaya.com/
schema/import_csm_agent" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:ns6="http://xml.avaya.com/schema/import_scopia" xmlns:ns5="http://
xml.avaya.com/schema/presence" xmlns:ns8="http://xml.avaya.com/schema/import_c
sm_cm" xmlns:ns7="http://xml.avaya.com/schema/import_sessionmanager" xmlns:ns
13="http://xml.avaya.com/schema/import_ce" xmlns:ns9="http://xml.avaya.com/sche
ma/import_csm_b5800" xmlns:ns12="http://xml.avaya.com/schema/import_mmcs" xm
lns:ns11="http://xml.avaya.com/schema/import_csm_mm" xmlns:ns10="http://xml.ava
ya.com/schema/import_mem_officelinx" xmlns:tns="http://xml.avaya.com/schema/imp
```

```
ort" xmlns:ns14="http://xml.avaya.com/schema/deltaImport">
  <tns:user>
    <authenticationType>basic</authenticationType>
    <displayName>load24, test</displayName>
    <displayNameAscii>load24, test</displayNameAscii>
    <isDuplicatedLoginAllowed>false</isDuplicatedLoginAllowed>
    <isEnabled>true</isEnabled>
    <isVirtualUser>false</isVirtualUser>
    <givenName>test</givenName>
    <givenNameAscii>test</givenNameAscii>
    <loginName>loadtest24@avaya.com</loginName>
    <middleName>loadtest24 update test middle name</middleName>
    <preferredLanguage>en_US</preferredLanguage>
    <source>local</source>
    <sourceUserKey>none</sourceUserKey>
    <status>provisioned</status>
    <surname>load24</surname>
    <surnameAscii>load24</surnameAscii>
    <userName>loadtest24</userName>
    <userPassword></userPassword>
    <commPassword></commPassword>
    <roles>
      <role>26f413c0-4a81-4451-83e8-f59382cc6d81</role>
      <role>End-User</role>
      <role>Collaboration.20Environment.20Auditor</role>
    </roles>
    <groups>
      <group>
        <name>TestGroup</name>
      </group>
    </groups>
    <localizedNames>
      <localizedName>
        <locale>English</locale>
        <name>load24</name>
      </localizedName>
    </localizedNames>
    <ownedContactLists>
      <contactList>
        <name>list-loadtest24_avaya.com</name>
        <isPublic>false</isPublic>
        <members>
          <memberUser>loadtest7@avaya.com</memberUser>
```

```
        <isFavorite>false</isFavorite>
        <isSpeedDial>false</isSpeedDial>
        <isPresenceBuddy>false</isPresenceBuddy>
        <priorityLevel>0</priorityLevel>
    </members>
    <members>
        <memberUser>loadtest6@avaya.com</memberUser>
        <isFavorite>false</isFavorite>
        <isSpeedDial>false</isSpeedDial>
        <isPresenceBuddy>false</isPresenceBuddy>
        <priorityLevel>0</priorityLevel>
    </members>
    <members>
        <memberUser>loadtest5@avaya.com</memberUser>
        <isFavorite>false</isFavorite>
        <isSpeedDial>false</isSpeedDial>
        <isPresenceBuddy>false</isPresenceBuddy>
        <priorityLevel>0</priorityLevel>
    </members>
    <members>
        <memberUser>loadtest4@avaya.com</memberUser>
        <isFavorite>false</isFavorite>
        <isSpeedDial>false</isSpeedDial>
        <isPresenceBuddy>false</isPresenceBuddy>
        <priorityLevel>0</priorityLevel>
    </members>
    <members>
        <memberUser>loadtest3@avaya.com</memberUser>
        <isFavorite>false</isFavorite>
        <isSpeedDial>false</isSpeedDial>
        <isPresenceBuddy>false</isPresenceBuddy>
        <priorityLevel>0</priorityLevel>
    </members>
    <members>
        <memberUser>loadtest2@avaya.com</memberUser>
        <isFavorite>false</isFavorite>
        <isSpeedDial>false</isSpeedDial>
        <isPresenceBuddy>false</isPresenceBuddy>
        <priorityLevel>0</priorityLevel>
    </members>
    <members>
        <memberUser>loadtest1@avaya.com</memberUser>
        <isFavorite>false</isFavorite>
```

```
            <isSpeedDial>false</isSpeedDial>
            <isPresenceBuddy>false</isPresenceBuddy>
            <priorityLevel>0</priorityLevel>
        </members>
        <members>
            <memberUser>jayesh1@avaya.com</memberUser>
            <isFavorite>false</isFavorite>
            <isSpeedDial>false</isSpeedDial>
            <isPresenceBuddy>false</isPresenceBuddy>
            <priorityLevel>0</priorityLevel>
        </members>
        <contactListType>general</contactListType>
    </contactList>
</ownedContactLists>
<ownedContacts>
    <contact>
        <displayName>load24, test</displayName>
        <displayNameAscii>load24, test</displayNameAscii>
        <givenName>test</givenName>
        <givenNameAscii>test</givenNameAscii>
        <preferredLanguage>en_US</preferredLanguage>
        <isPublic>false</isPublic>
        <source>local</source>
        <sourceUserKey>none</sourceUserKey>
        <surname>load24</surname>
        <surnameAscii>load24</surnameAscii>
        <ContactAddress>
            <address>Pune</address>
            <contactCategory>office</contactCategory>
            <contactType>phone</contactType>
        </ContactAddress>
        <addresses>
            <addressType>office</addressType>
            <name>Pune</name>
            <isPrivate>true</isPrivate>
        </addresses>
    </contact>
</ownedContacts>
<commProfileSet>
    <commProfileSetName>Primary</commProfileSetName>
    <isPrimary>true</isPrimary>
    <handleList>
        <handle>
```

```
                <handleName>load24</handleName>
                <handleType>sip</handleType>
                <handleSubType>username</handleSubType>
                <domainName>gssmgr.com</domainName>
            </handle>
        </handleList>
        <commProfileList>
            <commProfile xsi:type="ns7:SessionManagerCommProfXML"  xmlns:ns7="
http://xml.avaya.com/schema/import_sessionmanager">
                <commProfileType>SessionManager</commProfileType>
                <ns7:primarySM>SM_217</ns7:primarySM>
                <ns7:homeLocation>Pune</ns7:homeLocation>
                <ns7:maxSimultaneousDevices>1</ns7:maxSimultaneousDevices>
                <ns7:blockNewRegistrationWhenMaxActive>false</ns7:blockNewRegistr
ationWhenMaxActive>
                <ns7:enabledisablecalllog>false</ns7:enabledisablecalllog>
                <ns7:policyType>fixed</ns7:policyType>
            </commProfile>
            <commProfile xsi:type="ns8:xmlStationProfile"  xmlns:ns8="http://xml.avaya
.com/schema/import_csm_cm">
                <commProfileType>CM</commProfileType>
                <ns8:cmName>CM_249</ns8:cmName>
                <ns8:prefHandleId></ns8:prefHandleId>
                <ns8:useExistingExtension>false</ns8:useExistingExtension>
                <ns8:extension>1229</ns8:extension>
                <ns8:setType>1408</ns8:setType>
                <ns8:securityCode></ns8:securityCode>
                <ns8:port>X</ns8:port>
                <ns8:deleteOnUnassign>true</ns8:deleteOnUnassign>
                <ns8:overRideEndpointName>true</ns8:overRideEndpointName>
                <ns8:dualRegistration>false</ns8:dualRegistration>
                <ns8:enhCallrInfodisplay>false</ns8:enhCallrInfodisplay>
                <ns8:lockMessages>false</ns8:lockMessages>
                <ns8:tn>1</ns8:tn>
                <ns8:cor>1</ns8:cor>
                <ns8:cos>1</ns8:cos>
                <ns8:tests>false</ns8:tests>
                <ns8:dataModule>false</ns8:dataModule>
                <ns8:speakerphone>2-way</ns8:speakerphone>
                <ns8:displayLanguage>english</ns8:displayLanguage>
                <ns8:personalizedRingingPattern>1</ns8:personalizedRingingPattern>
                <ns8:messageLampExt>1229</ns8:messageLampExt>
                <ns8:muteButtonEnabled>true</ns8:muteButtonEnabled>
```

```
<ns8:ipSoftphone>false</ns8:ipSoftphone>
<ns8:survivableCOR>internal</ns8:survivableCOR>
<ns8:survivableTrunkDest>true</ns8:survivableTrunkDest>
<ns8:voiceMailNumber></ns8:voiceMailNumber>
<ns8:offPremisesStation>false</ns8:offPremisesStation>
<ns8:displayModule>false</ns8:displayModule>
<ns8:remoteOfficePhone>false</ns8:remoteOfficePhone>
<ns8:lwcReception>spe</ns8:lwcReception>
<ns8:lwcActivation>true</ns8:lwcActivation>
<ns8:lwcLogExternalCalls>false</ns8:lwcLogExternalCalls>
<ns8:cdrPrivacy>false</ns8:cdrPrivacy>
<ns8:redirectNotification>true</ns8:redirectNotification>
<ns8:perButtonRingControl>false</ns8:perButtonRingControl>
<ns8:bridgedCallAlerting>false</ns8:bridgedCallAlerting>
<ns8:bridgedIdleLinePreference>false</ns8:bridgedIdleLinePreference>
<ns8:confTransOnPrimaryAppearance>false</ns8:confTransOnPrimaryA
ppearance>
<ns8:customizableLabels>true</ns8:customizableLabels>
<ns8:expansionModule>false</ns8:expansionModule>
<ns8:ipVideoSoftphone>false</ns8:ipVideoSoftphone>
<ns8:activeStationRinging>single</ns8:activeStationRinging>
<ns8:switchhookFlash>false</ns8:switchhookFlash>
<ns8:ignoreRotaryDigits>false</ns8:ignoreRotaryDigits>
<ns8:h320Conversion>false</ns8:h320Conversion>
<ns8:serviceLinkMode>as-needed</ns8:serviceLinkMode>
<ns8:multimediaMode>basic</ns8:multimediaMode>
<ns8:mwiServedUserType></ns8:mwiServedUserType>
<ns8:emergencyLocationExt>1229</ns8:emergencyLocationExt>
<ns8:alwaysUse>false</ns8:alwaysUse>
<ns8:precedenceCallWaiting>false</ns8:precedenceCallWaiting>
<ns8:autoSelectAnyIdleAppearance>false</ns8:autoSelectAnyIdleAppea
rance>
<ns8:coverageMsgRetrieval>true</ns8:coverageMsgRetrieval>
<ns8:autoAnswer>none</ns8:autoAnswer>
<ns8:dataRestriction>false</ns8:dataRestriction>
<ns8:idleAppearancePreference>false</ns8:idleAppearancePreference>
<ns8:callWaitingIndication>false</ns8:callWaitingIndication>
<ns8:attCallWaitingIndication>false</ns8:attCallWaitingIndication>
<ns8:distinctiveAudibleAlert>false</ns8:distinctiveAudibleAlert>
<ns8:restrictLastAppearance>true</ns8:restrictLastAppearance>
<ns8:adjunctSupervision>false</ns8:adjunctSupervision>
<ns8:busyAutoCallbackWithoutFlash>false</ns8:busyAutoCallbackWitho
utFlash>
```

```
<ns8:audibleMessageWaiting>false</ns8:audibleMessageWaiting>
<ns8:imsFeatureSequencing>false</ns8:imsFeatureSequencing>
<ns8:displayClientRedirection>false</ns8:displayClientRedirection>
<ns8:selectLastUsedAppearance>false</ns8:selectLastUsedAppearance
>
<ns8:coverageAfterForwarding>s</ns8:coverageAfterForwarding>
<ns8:directIpIpAudioConnections>true</ns8:directIpIpAudioConnections
>
<ns8:ipAudioHairpinning>false</ns8:ipAudioHairpinning>
<ns8:stationSiteData>
   <ns8:headset>false</ns8:headset>
   <ns8:speaker>false</ns8:speaker>
   <ns8:mounting>d</ns8:mounting>
   <ns8:cordLength>0</ns8:cordLength>
</ns8:stationSiteData>
<ns8:buttons>
   <ns8:number>1</ns8:number>
   <ns8:type>call-appr</ns8:type>
   <ns8:data1></ns8:data1>
   <ns8:data2></ns8:data2>
   <ns8:data3></ns8:data3>
   <ns8:data4></ns8:data4>
   <ns8:data5></ns8:data5>
   <ns8:data6></ns8:data6>
</ns8:buttons>
<ns8:buttons>
   <ns8:number>2</ns8:number>
   <ns8:type>call-appr</ns8:type>
   <ns8:data1></ns8:data1>
   <ns8:data2></ns8:data2>
   <ns8:data3></ns8:data3>
   <ns8:data4></ns8:data4>
   <ns8:data5></ns8:data5>
   <ns8:data6></ns8:data6>
</ns8:buttons>
<ns8:buttons>
   <ns8:number>3</ns8:number>
   <ns8:type>call-appr</ns8:type>
   <ns8:data1></ns8:data1>
   <ns8:data2></ns8:data2>
   <ns8:data3></ns8:data3>
   <ns8:data4></ns8:data4>
   <ns8:data5></ns8:data5>
```

```
                <ns8:data6></ns8:data6>
            </ns8:buttons>
            <ns8:nativeName/>
            <ns8:unconditionalInternalDest></ns8:unconditionalInternalDest>
            <ns8:unconditionalInternalActive>false</ns8:unconditionalInternalActive>
            <ns8:unconditionalExternalDest></ns8:unconditionalExternalDest>
            <ns8:unconditionalExternalActive>false</ns8:unconditionalExternalActiv
e>
            <ns8:busyInternalDest></ns8:busyInternalDest>
            <ns8:busyInternalActive>false</ns8:busyInternalActive>
            <ns8:busyExternalDest></ns8:busyExternalDest>
            <ns8:busyExternalActive>false</ns8:busyExternalActive>
            <ns8:noReplyInternalDest></ns8:noReplyInternalDest>
            <ns8:noReplyInternalActive>false</ns8:noReplyInternalActive>
            <ns8:noReplyExternalDest></ns8:noReplyExternalDest>
            <ns8:noReplyExternalActive>false</ns8:noReplyExternalActive>
            <ns8:sacCfOverride>n</ns8:sacCfOverride>
            <ns8:lossGroup>2</ns8:lossGroup>
            <ns8:timeOfDayLockTable></ns8:timeOfDayLockTable>
            <ns8:emuLoginAllowed>false</ns8:emuLoginAllowed>
            <ns8:ec500State>enabled</ns8:ec500State>
            <ns8:muteOnOffHookInSCMode>false</ns8:muteOnOffHookInSCMode>
            <ns8:type3pccEnabled>None</ns8:type3pccEnabled>
            <ns8:calculateRoutePattern>false</ns8:calculateRoutePattern>
            <ns8:enableReachStaDomainControl>s</ns8:enableReachStaDomainC
ontrol>
            <ns8:multimediaEarlyAnswer>false</ns8:multimediaEarlyAnswer>
            <ns8:bridgedApprOrigRestr>false</ns8:bridgedApprOrigRestr>
            <ns8:callApprDispFormat>disp-param-default</ns8:callApprDispFormat>
            <ns8:ipPhoneGroupId></ns8:ipPhoneGroupId>
            <ns8:xid>false</ns8:xid>
            <ns8:stepClearing>false</ns8:stepClearing>
            <ns8:fixedTei>false</ns8:fixedTei>
            <ns8:endptInit>false</ns8:endptInit>
            <ns8:isShortCallingPartyDisplay>true</ns8:isShortCallingPartyDisplay>
            <ns8:displayCallerId>true</ns8:displayCallerId>
            <ns8:callerIdMsgWaitingIndication>false</ns8:callerIdMsgWaitingIndicati
on>
            <ns8:recallRotaryDigit>false</ns8:recallRotaryDigit>
            <ns8:bridgingToneForThisExtension>n</ns8:bridgingToneForThisExtens
ion>
            <ns8:terminalNumberPart1>0</ns8:terminalNumberPart1>
            <ns8:terminalNumberPart2>0</ns8:terminalNumberPart2>
```

```
                <ns8:terminalNumberPart3>0</ns8:terminalNumberPart3>
                <ns8:terminalNumberPart4>0</ns8:terminalNumberPart4>
                <ns8:systemId></ns8:systemId>
                <ns8:features></ns8:features>
                <ns8:features2></ns8:features2>
                <ns8:attendant>false</ns8:attendant>
                <ns8:ipHoteling>false</ns8:ipHoteling>
            </commProfile>
        </commProfileList>
    </commProfileSet>
  </tns:user>
</tns:users>
```

In case of an error, an appropriate HTTP status code is sent back along with a response XML. The Response XML conforms to the XSD described in Appendix B

Responses can contain below status codes for different scenarios.

HTTP Status Code	HTTP Status Message	Description
400	Bad Request	All validation exceptions.
404	Not Found	User not found
500	Internal Server Error	Any errors other than validation exception or username conflict.

For example,

Successful response

```
HTTP/1.1 200
Connection: Closed

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users xsi:schemaLocation="http://xml.avaya.com/schema/import
userimport.xsd" xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns6="http://xml.avaya.com/schema/import_scopia"
xmlns:ns5="http://xml.avaya.com/schema/presence"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm"
xmlns:ns7="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns13="http://xml.avaya.com/schema/import_ce"
xmlns:ns9="http://xml.avaya.com/schema/import_csm_b5800"
xmlns:ns12="http://xml.avaya.com/schema/import_mmcs"
xmlns:ns11="http://xml.avaya.com/schema/import_csm_mm"
xmlns:ns10="http://xml.avaya.com/schema/import_mem_officelinx"
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns14="http://xml.avaya.com/schema/deltaImport">
    <tns:user>
        <authenticationType>basic</authenticationType>
        <displayName>test, test</displayName>
        <displayNameAscii>test, test</displayNameAscii>
        <isDuplicatedLoginAllowed>false</isDuplicatedLoginAllowed>
        <isEnabled>true</isEnabled>
        <isVirtualUser>false</isVirtualUser>
        <givenName>test</givenName>
        <givenNameAscii>test</givenNameAscii>
        <loginName>test123@avaya.com</loginName>
        <preferredLanguage>en_US</preferredLanguage>
        <source>local.ws</source>
        <sourceUserKey>none</sourceUserKey>
        <status>provisioned</status>
        <surname>test</surname>
        <surnameAscii>test</surnameAscii>
        <userName>test123</userName>
        <userPassword></userPassword>
        <commPassword></commPassword>
        <roles>
            <role>End-User</role>
        </roles>
        <ownedContactLists>
            <contactList>
                <name>list-test123_avaya.com</name>
```

Error Response

```
HTTP/1.1 404
Content-Type: application/xml
Connection: Closed
<ns1:errorResponse xmlns:ns1="http://www.avaya.com/errorResponse/"
>
    <code>404</code>
    <status>Not Found</status>
    <message>Resouce trying to request for is not
available</message>
    <errors>
    <ns2:error xmlns:ns2="http://www.avaya.com/error/">
        <message>Resouce trying to request for is not
available</message>
        <details>
            <message>
                <reasonText>No entity found for query</reasonText>
            </message>
        </details>
    </error>
    </errors>
</ns1:errorResponse>
```

Code snippet

The following is the Java client sample code to get the user profile using RESTful web services. The GET request URL takes two **PATH** parameters: **entityType** and **uniqueId**.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
private String uri = "https://example.com/web/mgmtwebservice";
…
String result = client.resource(uri).path("user").
path("jmiller@avaya.com").get(String.class);
…
```

The same code passes PATH parameter **entityType** and **uniqueId** along with the get request to the server.

Note: For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

Sample Resultant stream content of **find** user:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<tns:users
xsi:schemaLocation="http://xml.avaya.com/schema/import
userimport.xsd" xmlns:ns4="http://xml.avaya.com/schema/import1"
xmlns:ns3="http://xml.avaya.com/schema/import_csm_mm"
xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ns9="http://xml.avaya.com/schema/deltaImport"
xmlns:ns5="http://xml.avaya.com/schema/import_csm_abg"
xmlns:ns6="http://xml.avaya.com/schema/import_sessionmanager"
xmlns:ns7="http://xml.avaya.com/schema/import_csm_agent"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns8="http://xml.avaya.com/schema/import_csm_cm">
    <tns:user>
        <authenticationType>basic</authenticationType>
        <displayName>Miller, John</displayName>
        <displayNameAscii>Miller, John</displayNameAscii>

<isDuplicatedLoginAllowed>false</isDuplicatedLoginAllowed>
        <isEnabled>true</isEnabled>
        <isVirtualUser>false</isVirtualUser>
        <givenName>John</givenName>
        <loginName>jmiller@avaya.com</loginName>
        <preferredLanguage>en_US</preferredLanguage>
        <source>TC_0101_1.xml</source>
        <sourceUserKey>none</sourceUserKey>
        <status>provisioned</status>
        <surname>Miller</surname>
        <userName>jmiller</userName>
        <userPassword></userPassword>
        <roles>
            <role>End-User</role>
        </roles>
        <ownedContactLists>
            <contactList>
                <name>list-jmiller_avaya.com</name>
                <isPublic>false</isPublic>
                <contactListType>general</contactListType>
            </contactList>
        </ownedContactLists>
        <commProfileSet>
            <commProfileSetName>Primary</commProfileSetName>
            <isPrimary>true</isPrimary>
        </commProfileSet>
    </tns:user>
</tns:users>
```

Delete user

The delete method deletes a user profile based on the login name provided in XML as **`entityXMLData`**.

`String post(String entityType, String entityXMLData, String OperationName)`

The delete User operation has the following components:

HTTP method	POST
URL	*https://host/web/v2/mgmtwebservice* *host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	Refer to Request Parameters section below.
Request Parameters	None.
Response	Refer to Response section below.

Request parameters

Name of the parameter	Data type	Type	Description
entityType	String	Form	Specifies an entity name. Expected value: **`user`**
entityXMLData	String	Form	Specifies User XML data conforming to delete XSD. For more information on user delete XSD, see Appendix B.
operationName	String	Form	Specifies operation name. Expected value: delete.

Response

Successful response does not contain any data, but the HTTP status code of the response will be 204.

In case of an error, an appropriate HTTP status code is sent back along with a response XML. The Response XML conforms to the XSD described in Appendix B

Responses can contain below status codes for different scenarios.

HTTP Status Code	HTTP Status Message	Description

204	NA	User updated successfully. No content is sent back in the response.
400	Bad Request	All validation exceptions.
409	Conflict	User with same login name already exists.
500	Internal Server Error	Any errors other than validation exception or username conflict. One may also get a 500 response code for cases where multiple users are updated using a single request and each user update in the request results with different HTTP status code. Please refer to example below.

For example,

Successful response

```
HTTP/1.1 204
Connection: Closed
No Content
```

Error response

```
HTTP/1.1 400
Content-Type: application/xml
Connection: Closed
<ns1:errorResponse xmlns:ns1="http://www.avaya.com/errorResponse/">
    <code>400</code>
    <status>Bad Request</status>
    <message>Invalid request data</message>
    <errors>
        <ns2:error xmlns:ns2="http://www.avaya.com/error/">
            <message>Invalid request data</message>
            <details>
                <message>
                    <reasonText>Invalid XML Data - The end-tag for e
lement type "tns:deleteType" must end with a '&gt;' delimiter.</reas
onText>
                </message>
            </details>
        </ns2:error>
    </errors>
</ns1:errorResponse>


HTTP/1.1 409
Content-Type: application/xml
Connection: Closed

<ns1:errorResponse xmlns:ns1="http://www.avaya.com/errorResponse/">
    <code>409</code>
    <status>Conflict</status>
                </message>
            </details>
        </ns2:error>
    </errors>
</ns1:errorResponse>

HTTP/1.1 500
Content-Type: application/xml
Connection: Closed
<ns1:errorResponse xmlns:ns1="http://www.avaya.com/errorResponse/">
    <code>500</code>
    <status>Internal Server Error</status>
    <message>Request can not be processed due to some error in
server</message>
<errors>
  <ns2:error xmlns:ns2="http://www.avaya.com/error/">
        <message>Request can not be processed due to some error in
```

```
server</message>
        <uniqueId>test@avaya.com</uniqueId>
        <details>
            <message>
                <reasonCode>STACOMMPROFILE0006</reasonCode>
                <reasonText>Unassign of Endpoint "4900000" from user
"test@avaya.com" failed. Cause of failure: 1 00000000 41d5 Bridged
appearances must be removed before removal</reasonText>
            </message>
        </details>
    </ns2:error>
</errors>
</ns1:errorResponse>
```

The following is the Java client sample code to delete a user profile using RESTful web services. The delete request URL takes two **FORM** parameters: **entityType** and **entityXMLData**.

Code snippet

deletesample.xml -– The following is a sample XML file to specify criteria to retrieve a list of users

```
  <?xml version="1.0" encoding="UTF-8"?>
<tns:deleteUsers
xmlns:tns="http://xml.avaya.com/schema/bulkdelete"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xml.avaya.com/schema/bulkdelete
       UserProfileSchemaDefinitionForBulkDelete.xsd ">
```

```
      <tns:deleteType>permanent</tns:deleteType>
      <tns:user>
            <tns:loginName>jmiller@avaya.com</tns:loginName>
      </tns:user>
</tns:deleteUsers>
```

Sample code – The following is a sample code snippet to delete a user using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate
the user >>
…
private String uri = https://example.com/web/mgmtwebservice";
…
  final InputStream inputStream = getClass().getClassLoader()
                .getResourceAsStream("deletesample.xml");
  final Reader reader = new InputStreamReader(inputStream);
  final BufferedReader br = new BufferedReader(reader);
  final StringBuffer line = new StringBuffer();
      String st = "";
      while ((st = br.readLine()) != null)
      {
            line.append(st + "\n");
      }
  Final String entityXmlData = line.toString();
  final Form form = new Form();
      form.add("entityType", "user");
      form.add("entityXMLData", entityXmlData);
      form.add("operationName", "delete"); //required
      String result = client.resource(uri).post(String.class,
form);
 …
```

Note: You can soft delete or permanently delete a user. When you choose to soft delete a user, you can restore this user at a later time.When you choose to permanently delete a user, the process is irreversible and the deleted user cannot be restored. To specify the delete option, you can provide a permanent or soft delete value in the XML **`entityXMLData`** parameter. For a sample code snippet to establish an SSL connection and authenticate the user, see Sample code to establish SSL connection.

Self provisioning password feature (web services interface) details

This section includes the implementation details of the User Communication Profile Password Management web services.

The URL for the REST-based web services to be accessed is
`https://<System Manager Server FQDN>/selfprovisioning/ws/<URL parameter(s)>`

NOTE: Each request is secure and client should be a validate certificate based Authenticated entity only. The client communicates with the server over SSL using the server certificate. See Sample code snippet to establish SSL connection and authentication.

SIP Password Change

This operation allows user to change SIP Communication Profile password using HTTP POST methods.

The operation has the following components:

HTTP method	PUT
URL	*https://<host>/selfprovisioning/ws* *the host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	See Request Parameters
Response	See Response

Request parameters

The SIP Password Change request takes the following parameters:

Name of the parameter	Data type	Type	Description
handle	String	Form Parameter	Login name of a user to change SIP communication profile password.
currentpassword	String	Form Parameter	Current SIP Communication Profile password.
newpassword	String	Form Parameter	New SIP Communication Profile password.

Response

The response contains the SIP Communication Profile password change operation status. The response generates a status (data type: String) messages.

For example,
`SUCCESS : SIP Password Changed Successfully.`

Code snippet

The following is the Java client sample code to update last login using RESTful web services. The GET request takes form parameters: **handle, currentpassword, newpassword.**

Sample code – The following is a sample Java code to create a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
  private String uri = "https://example.com/
selfprovisioning/ws/SIP";
…

      String result = client.resource (uri) .queryParam("handle",
"johnmiller@sip.com");
      .queryParam("currentpassword", "123456");
      .queryParam("newpassword", "Avaya123$");
      .put(String.class);


…
```

Endpoint Password Change

This operation allows user to change H323/Endpoint security code using HTTP POST methods.

The operation has the following components:

HTTP method	PUT
URL	*https://<host>/selfprovisioning/ws* *the host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	See Request Parameters
Response	See Response

Request parameters

The Endpoint Password Change request takes the following parameters:

Name of the parameter	Data type	Type	Description
handle	String	Form Parameter	Login name of a user to change Endpoint security code.
currentpassword	String	Form Parameter	Current security code of Endpoint.
newpassword	String	Form Parameter	New security code of Endpoint.

Response

The response contains the Endpoint security code change operation status. The response generates a status (data type: String) messages.

For example,

```
SUCCESS : H323 Password Changed Successfully.
```

Code snippet

The following is the Java client sample code to update last login using RESTful web services. The GET request takes form parameters: **handle, currentpassword, newpassword.**

Sample code – The following is a sample Java code to create a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
  private String uri = "https://example.com/
selfprovisioning/ws/H.323";
…

      String result = client.resource (uri) .queryParam("handle",
"johnmiller@sip.com");
      .queryParam("currentpassword", "123456");
      .queryParam("newpassword", "Avaya123$");
      .put(String.class);


…
```

Agent Password Change

This operation allows user to change Agent Communication password using HTTP POST methods.

The operation has the following components:

HTTP method	PUT
URL	*https://<host>/selfprovisioning/ws* *the host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	See Request Parameters
Response	See Response

Request parameters

The Agent Password Change request takes the following parameters:

Name of the parameter	Data type	Type	Description

handle	String	Form Parameter	Login name of an Agent to change communication profile password.
currentpassword	String	Form Parameter	Agent's current Communication Profile password.
newpassword	String	Form Parameter	New Communication Profile password for Agent.

Response

The response contains the Agent Communication Profile password change operation status. The response generates a status (data type: String) messages.

For example,

```
SUCCESS : CM Agent Password Changed Successfully.
```

Code snippet

The following is the Java client sample code to update last login using RESTful web services. The GET request takes form parameters: **handle, currentpassword, newpassword.**

Sample code – The following is a sample Java code to create a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
  private String uri = "https://example.com/
selfprovisioning/ws/Agent";
…

      String result = client.resource (uri) .queryParam("handle",
"johnmiller@sip.com");
      .queryParam("currentpassword", "123456");
      .queryParam("newpassword", "Avaya123$");
      .put(String.class);


…
```

Messaging Password Change

This operation allows user to change Messaging Communication profile password using HTTP POST methods.

The operation has the following components:

HTTP method	PUT
URL	*https://<host>/selfprovisioning/ws* *the host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	See Request Parameters
Response	See Response

Request parameters

The Messaging Communication profile password change request takes the following parameters:

Name of the parameter	Data type	Type	Description
handle	String	Form Parameter	Login name of a user to change Messaging communication profile password.
currentpassword	String	Form Parameter	User's current Messaging Communication Profile password.
newpassword	String	Form Parameter	New Messaging Communication Profile password of user.

Response

The response contains the Messaging Communication Profile password change operation status. The response generates a status (data type: String) messages.

For example,

```
SUCCESS : Messaging Password Changed Successfully.
```

Code snippet

The following is the Java client sample code to update last login using RESTful web services. The GET request takes form parameters: **handle, currentpassword, newpassword.**

Sample code – The following is a sample Java code to create a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
  private String uri = "https://example.com/
selfprovisioning/ws/Messaging";
…

     String result = client.resource (uri)
      .queryParam("handle", "johnmiller@sip.com");
      .queryParam("currentpassword", "123456");
      .queryParam("newpassword", "Avaya123$");
      .put(String.class);


…
```

Reset Password

This operation allows user to reset Communication profile password using HTTP GET methods.
Reset password API sends the activation link on users email account to activate the new communication profile password.

The operation has the following components:

HTTP method	GET
URL	*https://<host>/selfprovisioning/ws* *the host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	See Request Parameters
Response	See Response

Request parameters

The Reset Communication profile password request takes the following parameters:

Name of the parameter	Data type	Type	Description
handle	String	Form Parameter	Handle of a user to reset communication profile password.

Response

The response contains the Reset Communication Profile password operation status. The response generates a status (data type: String) messages.

For example,

```
SUCCESS : Password Activation link has been emailed to registered email
address.
```

Code snippet

The following is the Java client sample code to update last login using RESTful web services. The GET request takes form parameters: **handle.**

Sample code – The following is a sample Java code to create a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
  private String uri = "https://example.com/
selfprovisioning/ws/Reset ";
…

      String result = client.resource (uri)
        .queryParam("handle", "johnmiller@sipdomain.com");
        .get(String.class);


…
```

Activate Password

This operation allows user to activate new Communication profile password using HTTP GET methods.
Until user doesn't click the activation link new Communication Profile password will not be set and activated.

The operation has the following components:

HTTP method	GET
URL	*https://<host>/selfprovisioning/ws* *the host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	See Request Parameters
Response	See Response

Request parameters

The Activate Communication profile password request takes the following parameters:

Name of the parameter	Data type	Type	Description
uuid	String	Query Parameter	UUID of a user to activate a communication profile password.

Response

The response contains the Activate Communication Profile password operation status. The response generates a status (data type: String) messages.

For example,

```
SUCCESS : New Communication Profile Password has been activated
successfully.
```

Code snippet

The following is the Java client sample code to update last login using RESTful web services. The GET request takes form parameters: **uuid**

Sample code – The following is a sample Java code to create a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
  private String uri = "https://example.com/
selfprovisioning/ws/Activate";
…

      String result = client.resource (uri)
        .queryParam("uuid", "cad7255e-9956-4e8b-8b6e-
   7c17cef1200b");
        .get(String.class);


…
```

Changing Multiple Communication Profile Passwords in single API

This operation allows user to activate new Communication profile password using HTTP GET methods.
Until user doesn't click the activation link new Communication Profile password will be set and activated.

The operation has the following components:

HTTP method	POST
URL	*https://<host>/selfprovisioning/ws* *the host* is the FQDN of the System Manager host servicing the request.
Headers	Content-Type: application/x-www-form-urlencoded
Body	None.
Request Parameters	See Request Parameters
Response	See Response

Request parameters

The Activate Communication profile password request takes parameters in APPLICATION_JSON form:

Input Stream (APPLICATION_JSON) format	Input stream sample.

{
"users": {
"type": "array",
	{
	"users": [
	{

```
                "minLength": "1",
                "items": {
                    "type": "object",
                    "properties": {
                        "handle": {"type":
"string"},
                        "commprofiles": {
                            "type": "array",
                            "minLength": "1",
                            "items": {
                                "type": "object",
                                "properties": {
                                    "profiletype":
{"type": "string", "enum": [ "SIP", "H.323",
"Agent", "Messaging" ]},

"currentpassword": {"type": "string"},
                                    "newpassword":
{"type": "string"}
                                },
                                "required":
["profiletype", "currentpassword",
"newpassword"]
                            }
                        }
                    },
                    "required": ["handle",
"commprofiles"]
                }
            }
        }
```

```
                "handle": "u1@avaya.com",
                "commprofiles": [
                    {
                        "profiletype": "SIP",
                        "currentpassword":
"welcome",
                        "newpassword":
"Admin123"
                    },
                    {
                        "profiletype": "Agent",
                        "currentpassword":
"4321",
                        "newpassword": "5678"
                    }
                ]
            },
            {
                "handle": "u2@avaya.com",
                "commprofiles": [
                    {
                        "profiletype": "H.323",
                        "currentpassword":
"5678",
                        "newpassword": "4321"
                    },
                    {
                        "profiletype":
"Messaging",
                        "currentpassword":
"7045",
                        "newpassword": "9420"
                    }
                ]
            }
        ]
    }
```


Response

The response contains the Activate Communication Profile password operation status. The response generates a status (data type: String) messages.

For example,
SUCCESS.

Code snippet

The following is the Java client sample code to update last login using RESTful web services. The GET request takes form parameters: **uuid**

Sample code – The following is a sample Java code to create a user with mandatory attributes using RESTful web services.

```
<< code snippet to establish SSL connection and authenticate the
user >>
…
  private String uri = "https://example.com/
selfprovisioning/ws/PasswordManager";
…

String jsonInput = "{ "
                      + "\"users\": [ "
                            + "{ \"handle\": \"u1@avaya.com\","
                                  + "\"commprofiles\": [ "
                                        +
"{ \"profiletype\": \"SIP\", \"currentpassword\": \"welcome\", \
"newpassword\": \"Admin123\" },"
                                        +
"{ \"profiletype\": \"Agent\", \"currentpassword\": \"4321\", \"
newpassword\": \"5678\" }"

                              + "] "
                         + "}, "
                         + "{ \"handle\": \"u2@avaya.com\","
                            + "\"commprofiles\": [ "
                            +
"{ \"profiletype\": \"H.323\", \"currentpassword\": \"5678\", \"
newpassword\": \"4321\" }, "
                            +
"{ \"profiletype\": \"Messaging\", \"currentpassword\": \"7045\"
, \"newpassword\": \"9420\" } "
                            + "] "
                      +"} "
                   + "] "
                + "}";

   String result = client.resource ( uri
          .type("application/json")
          .post(String.class, jsonInput);

…
```

Exceptions

User management web services methods support CRUD operation for one or more user records. The returned object of exposed Web methods has status details containing the error or info message depending on the operation status. In the scenario where multiple user records are expected, if there are errors for certain records, the returned status object will contain information of the failed records.

The following table lists the exceptions that you may encounter when using the System Manager UM WS along with their potential solutions:

For error code, see Appendix D.

System Manager UM WS exceptions 193911.1.0.doc	
Exception	**Potential cause and solution**
A user with this loginname already exists	Occurs when a Create User call receives a call for creating a user that already exists in System Manager. Send your request again after removing this user data.
Error! I/O error while reading the XML file; root cause Unexpected character'd' (code 100) in prolog; expected '<' at [row,col {unknown-source}]: [1,1]	Occurs when an invalid XML is passed in the call. Verify the XML against the defined XSD.
Invalid criteria argument	Occurs when a lookup call receives an invalid XML as search criteria. Verify the search criteria XML against the defined XSD.
Authorization error	Occurs when the user sent for authentication does not have permissions to execute the user management operations. Verify the user permissions before executing the client.
Invalid access	Occurs when the user sent for authentication fails to be authenticated. Specify a valid user before executing the client.

Entity type not supported	Occurs when entity type in create, update user, delete a user, lookup and find user input is not the user. Verify that the **`User`** is being passed as the entity type for the calls.
Null argument- entitytype	Occurs when the null value is being passed as an entity type. Verify that the **`User`** is being passed as the entity type for the calls.
Update mode not supported	Occurs when Update User call receives a call for updating user without update mode specified. Verify that **`merge`** or **`replace`** is being passed as update mode.
Unable to find method	Occurs when an invalid method call is done.For a list of methods, see API (web services interface) Details.

Multi-tenancy considerations

System Manager Release 6.3.4 introduces the multi-tenancy feature.

A multi-tenant System Manager lets tenants share the same instance of System Manager, while allowing them to do user management to fit their needs as if it runs on a dedicated environment. A Tenant is the customer who uses multi-tenant System Manager.

To support multi-tenancy in UPM web services, four elements have been introduced in the XSD.These elements are optional. These Elements are

Attributes	Description
Tenant	Name of the tenant the user belongs
organizationUnitLevelOne	Name of the organizationUnitLevelOne.For e.g. Site
organizationUnitLevelTwo	Name of the organizationUnitLevelTwo.For e.g. Department
organizationUnitLevelThree	Name of the organizationUnitLevelThree.For e.g. Team
createTenantIfNotAlreadyPresent	If the flag is set as true, the Tenant will be created if the same does not already existing in the system

Note: A flag has been added as an attribute. If the flag is set as true, the tenant will be created if the same does not already exist in the system

Using UPM web services, you can associate a tenant to a user using create/update user API.For removing a tenant associated with the user, you need to delete the user and re-create the user with the new tenant information.

System Manager will implicitly create a new Tenant (if it does not exist) when associating tenant with User using the Web Services.

Deleting the User will not delete the Tenant associated with the User.

Security considerations

Your application development organization must provide the appropriate amount of security for your particular application or recommend appropriate security measures to your application customers for the deployment of your application or both. Therefore, you must be aware of the security measures that System Manager web services already takes.

Each UM WS request contains the **username** and **password** and System Manager performs authentication through the HTTP Basic authentication scheme. The client communicates with the server over SSL using the server certificate. This certificate is used by the client to authenticate the System Manager server and set up a secure connection, after which the authentication of the client is done through userid/password.

1. System Manager UM WS requires a valid ***username*** and ***password*** to access all the web services. A user account must exist in the System Manager and be able to log in to System Manager Console.

- All the web services must be secured by using the secure SSL port 443.The user management web services can be accessed only on port 443 over SSL.
- System Manager UM WS authorises an authenticated user using role-based access control (RBAC). You can use predefined roles for user management web services CRUD operation such as Security Administrator and System Administrator. You can create Customised Roles (Read/Write permission) as per the RBAC guidelines. For example, customise a role which only has permission to create or delete a user.

To learn more about security, see Sample code snippet to establish SSL connection and authentication.

Authentication

System Manager UM Webservices is authenticated just like System Manager Web UI/Interface. For the UM webservices you need to pass a username and password for authentication. This user on the System Manager should have permission to manage users (read, create, modify and delete) and manage (read, create, modify and delete) elements (RTS_Administration). Certificate-based user authentication is not supported for System management UM WS APIs.

Generating certificates

The System Manager UM WS requires adding trusted root certificate to communicate over TLS at secured port 443. Following steps must be carried out to install trusted certificate on a webservice client.

1. Download Root CA certificate

Following steps must be carried out, depending upon whether System Manager is using internal CA or third-party CA issued certificate, to download root CA certificate.

System Manager Internal CA certificate are in use.

System Manager Internal CA can be setup as a **root CA** or as a **subordinate/intermediate CA** of another CA. In all such cases use the following steps to download the CA certificate file.

Download the System Manager Root Certificate using the System Manager Console.
To download the certificate:

- On the System Manager Web Console, click **Services > Security > Certificates > Authority**.
- In the left navigation pane, click **CA Structure & CRLs.**
- Click **Download PEM File** of the **Root CA** for CA with name **tmdefaultca**.
- Save the file as **trust-cert.pem.**

Third-party CA Issued certificate are in use.

System Manager supports installation of third-party CA issued identity certificate for TLS communication. If third-party CA issued certificate are being used for System Manager, **trust-cert.pem** should contain the third-party root CA certificate.

A root CA certificate refers to the self-signed certificate, the topmost certificate in the certificate chain/hierarchy.

For System Manager certificate management refer Avaya Aura® System Manager Certificate Management.

2. Convert the exported PEM file from System Manager to JKS Keystore.
To convert, run following command:
`keytool -importcert -file trust-cert.pem -keystore trust.jks`

where **`trust-cert.pem`** is the exported CA certificate file from System Manager Console and **`trust.jks`** is the output file name.

Note: By default, Java supports "JKS" key store type for SSL handshake. To use a different format of trust store for SSL handshake with user management web services, install a new provider like bouncy castle that supports different formats of trust store such as PEM, PKCS12, CER, etc. The default value for the "keystore.type" property is JKS.

3. Assign **`javax.net.ssl.trustStore`** property value points to the trust store certificate file location required to establish an SSL connection.

Sample code

Sample code to establish SSL connection

The following is the sample code to establish an SSL connection and authenticate the user:

```
…
      import java.net.URI;
      import java.util.Properties;
      import javax.ws.rs.core.UriBuilder;
      import com.sun.jersey.api.client.Client;
      import com.sun.jersey.api.client.WebResource;
      import com.sun.jersey.api.client.config.ClientConfig;
      import
com.sun.jersey.api.client.config.DefaultClientConfig;
      import
com.sun.jersey.api.client.filter.HTTPBasicAuthFilter;
      import com.sun.jersey.api.representation.Form;
      …
      Properties properties = System.getProperties();

properties.put("javax.net.ssl.trustStore","trustfile.jks");
         properties.put("javax.net.ssl.trustStorePassword",
           "trustpassword" );
      config = new DefaultClientConfig();
      client = Client.create(config);
      client.addFilter(new HTTPBasicAuthFilter("username",
      "userpassword");
      uri = UriBuilder.formUri(baseUri);
      …
```

javax.net.ssl.trustStore property value points to the trust store certificate file location required to establish an SSL connection.

Sample code snippet to create, update and retrieve users

The following is the sample code for creating, updating, and retrieving users based on criteria and deleting users. See Appendix C for code files.

```
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.Reader;
import java.io.StringBufferInputStream;
import java.io.StringWriter;
import java.net.URI;
import java.util.Properties;

import javax.ws.rs.core.UriBuilder;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.OutputKeys;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.TransformerFactoryConfigurationError;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.w3c.dom.Text;
import org.xml.sax.SAXException;

import com.sun.jersey.api.client.Client;
import com.sun.jersey.api.client.WebResource;
import com.sun.jersey.api.client.config.ClientConfig;
import com.sun.jersey.api.client.config.DefaultClientConfig;
import com.sun.jersey.api.client.filter.HTTPBasicAuthFilter;
import com.sun.jersey.api.representation.Form;

/**
 *
 * @author
 *
 */
public class WebServiceSampleClient
{
```

```
      private ClientConfig config;
      private Client client;
      private String truststoreFileName = "trustStore.jks";
      private String truststorePassword = "trustpassword";
      private String username = "username";
      private String password = "password";
      private String uri = "https://example.com/web/mgmtwebservice";

       /**
       *
       * @param args
       */
      public static void main(String[] args)
      {
            try
            {
                 WebServiceSampleClient webServiceClient = new
WebServiceSampleClient();

               // Create new user
               String entityXmlData = "createsample.xml";
               String result = webServiceClient.postUser("user",

      webServiceClient.getXMLStream(entityXmlData));
               System.out.println(result);

               // Update user

               entityXmlData = "updatesample.xml";
               String updateResult = webServiceClient.putUser("user",

      webServiceClient.getXMLStream(entityXmlData), "merge");
               System.out.println(updateResult);

               String searchCriteria = "searchcriteriasample.xml";
               String criteria =
webServiceClient.getXMLStream(searchCriteria)
                           .replaceAll("&", "&amp;").replaceAll("<",
"&lt;")
                           .replaceAll(">", "&gt;").replaceAll("\"",
"&quot;")
                           .replaceAll("'", "&apos;");

             // Criteria based user lookup
                 String lookupresult = webServiceClient.lookup("user",
criteria);
                 System.out.println(lookupresult);

            // Criteria based update user
                 webServiceClient.putUserCriteriaBased("user", criteria,
```

```
"merge");

         // Find user attributes
            String findresult = webServiceClient.find("user",
                              "jmiller@avaya.com");
            System.out.println(findresult);

        // Delete user
            entityXmlData = "deletesample.xml";
            String deleteresult = webServiceClient.delete("user",

      webServiceClient.getXMLStream(entityXmlData));
            System.out.println(deleteresult);

                  webServiceClient.destroy();
            } catch (IOException e)
            {
                  e.printStackTrace();
            } catch (ParserConfigurationException e)
            {
                  e.printStackTrace();
            } catch (SAXException e)
            {
                  e.printStackTrace();
            } catch (TransformerException e)
            {
                  e.printStackTrace();
            }
      }
}
```

C Debugging

To debug your application, you must rely on:

1. **Exceptions** that your application receives and logs. See Exceptions.
2. **Server-side logs** found at: **`/var/log/Avaya/mgmt/mgmtwebservice/`**

Getting support

The Web Services Interface is available to partners and customers that want to develop user management utilities to work with the new System Manager API. Basic support for operational issues will be provided by Avaya Client Services (ACS) and a user management web services programmer guide will be available on the Avaya Tech Support Site. No application development support will be provided by ACS.

Customers and Partners looking for applications development support can access developer support through the Avaya DevConnect Program. Registered level membership in the DevConnect Program is free for Avaya customers, channel partners and commercial application developers, and provides access to technical documentation, educational and other Avaya Aura System Manager UM WS related materials.

The DevConnect Program offers for-pay, developer-oriented technical support as part of Enhanced Level memberships at the Support-enabled and Technology Partner levels. For further information please refer to current DevConnect Membership and Program information at www.avaya.com/devconnect, or contact Avaya DevConnect by email at devconnect@avaya.com.

DevConnect Technology Partners may also register Avaya Aura System Manager-enabled solutions and undertake Compliance Testing activities of interoperable solutions through the DevConnect Program.

Appendix A – Sample code

For the Java sample client code and resource files that are also part of the System Manager User management SDK, see SDK artifacts.

Web service Sample Client	WebServiceClient.java
Create user sample XML	Sample_Create_User.xml
Update user sample XML	Sample_Update_User.xml
Search Criteria sample XML	SearchCriteria- sample.xml
Delete user sample XML	Sample_Delete_User.xml

Appendix B – XSD as attachments

To access the following XML Schema Definitions (XSD) from the SDK, see SDK artifacts

User	userdeltaimport.xsd userimport.xsd
Station Profile	stationCommProfile.xsd
Messaging Profile	messagingCommProfile.xsd
Session Manager Profile	session_manager-comm_prof.xsd
User Management Lookup Criteria	User Management Lookup_criteria.xsd
Delete User	UserProfileSchemaDefinitionForBulkDelete.xsd

Error for v2 API	error.xsd
Error Response for v2 API	errorResponse.xsd

Appendix C – HTTP verbs

This appendix lists the specific methods supported:

- POST
  - Used for creation and deletion. This request contains XML as XML will appear in a bulk import XML file for a given user.
  - The HTTP response contains the HTTP status code 201 (created) and the URI contained standard location header.
  - For delete, the HTTP response contains the HTTP status code 200.
- GET
  - Used for retrieval.
  - The HTTP response contains the HTTP status code 200 with results as parameters of the response.
  - Two variations to this GET request:
    - When the URI is used (as returned in a POST), just the single user is returned.
    - When searching for multiple resources, the search is performed with HTTP parameters in the URL.
- PUT
  - Used for an update.
  - The HTTP response contains the HTTP status code 200.
  .

Appendix D – Error Code

This appendix lists the specific error codes methods supported in the UM WS.

# ERROR CODE
WE_000000= WebService: {0}
WE_000001= WebService: Configuration file error, {0}
WE_000002= WebService: Entity type not supported, {0} : {1}
WE_000003= WebService: Unable to load the class, {0}
WE_000004= WebService: Invalid access, {0}
WE_000005= WebService: Unable to find method, {0}
WE_000006= WebService: Invalid method argument, {0}
WE_000007= WebService: No access to the method, {0}
WE_000008= WebService: Method invocation exception, {0}
WE_000009= WebService: Invalid object, {0}
WE_000010= WebService: Unable to load component for entity, {0}
WE_000011= WebService: Class can not be initialized, {0}
WE_000012= WebService: Invalid converter, {0}
WE_000013= WebService: No converter available for entityType: {0}
WE_000014= WebService: Invalid validator, {0}
WE_000015= WebService: No validator available for entityType: {0}
WE_000016= WebService: Null argument, {0}
WE_000017= WebService: Invalid argument, {0}
WE_000018= WebService: Update mode not supported, {0}
WE_000019= WebService: {0} "{1}" already exists.
WE_000020= WebService: Cannot connect to database.
WE_000022= WebService: Invalid criteria argument, {0}

WE_000021= WebService: Entity process. Operation={0}, LoginId: {1}, ClientHost: {2}, Action: {3}, Result: {4}, Object={5}

# WARNING CODE
WE_100000= WebService: Configuration file close error, {0}

# AUDIT CODE
WA_200000= WebService: Entity process. Operation={0}, LoginId: {1}, ClientHost: {2}, Action: {3}, Result: {4}, Object={5}

# INFO CODE
WI_300000= WebService: Entity process. Operation={0}, LoginId: {1}, ClientHost: {2}, Action: {3}, Result: {4}, Object={5}

# DEBUG CODE
WD_400000= WebService: Entity process. Operation={0}, LoginId: {1}, ClientHost: {2}, Action: {3}, Result: {4}, Object={5}

ERROR CODE

BI_1056=No file to upload
BI_1057=No header line found
BI_1058=No terminating quote
BI_1059=CSV line is too long
BI_1060=Illegal column name {0}
BI_1061=Import job {0} is scheduled
BI_1062=Cancellation request is initialized
BI_1063=Error while scheduling the job
BI_1064=Validation error
BI_1065=I/O error while reading the XML file; root cause {0}
BI_1066=Import error {0}
BI_1067=Job must be in the running state
BI_1068=Illegal value {0} for column {1}
BI_1069=Import completed
BI_1070=Unsupported property {0}
BI_1071=Failed to parse XML user: {0} invalid XML file
BI_1072=Unable to load the XML schema for file import; root cause {0}
BI_1073=Internal Error while creating the XML parser; root cause {0}
BI_1074=XML not conforming to {0} schema; cause - Bad Root Element {1}
BI_1075=Missing mandatory columns {0}
BI_1076={0} {1} not found

BI_1077=Failure in lookup of {0} EJBs
BI_1078=Unable to lookup remote interface of UPM EJBs
BI_1079=Special character present in {0}
BI_1080=The Password field is not present. Setting the default password {0}
BI_1081=File extension {0} is not supported for bulk import
BI_1082=Error while reading file/folder {0}
BI_1083=Error while creating folder {0}
BI_1084=The file {0} is of zero length.
BI_1085={0} reference is null
BI_1086=No user record is present for importing. Invalid XML file {0}
BI_1087={0} "{1}" already exists.
BI_1088= DataBase Operation Error. Root Cause {0}
BI_1089= Rule Validation fails, {0}
BI_1090= General server error; please check the logs {0}
BI_1091= Invalid date; the scheduled date should be greater than the current date.
BI_1092= Failed to parse System Presence XML: {0} invalid XML file
BI_1093= No System presence record is present for import. Invalid XML file {0}
BI_1094= General server error; please check the logs
BI_1095= Scheduled bulk import job failed, {0}
BI_1096= Failure while loading EP atrifact {0} for UPM EP
BI_1097= Failure while loading converter {0} class.
BI_1098= Failure while reading the input file {0}
BI_1099= Error while scheduling the bulkimport job
BI_2000= Contact list member {0} does not exist
BI_2001= User type {0} does not exist
BI_2002= Role type {0} does not exist
BI_2003= Unable to access property {0}

BI_2004= Virtual user {0} cannot be deleted
BI_2005= Error during SecureStore initialization
BI_2006 =Complete import operation supports Skip option only
BI_2007= Partial import operation supports Merge and Replace options only
BI_2008= No matching record found for {0} {1}
BI_2009=Cannot connect to the database
BI_2010=Presentity and Watcher cannot be the same {0}
BI_2011=isSpeedDial is set to True; but SpeedDialContactAddress is not found for memberContact {0}.
BI_2012=isSpeedDial is set to True; but SpeedDialHandle is not found for memberUser {0}.
BI_2013=isSpeedDial is set to True; but SpeedDialEntry is not found for memberUser {0}.
BI_2014=isSpeedDial is set to False; but SpeedDialHandle or speedDialEntry found for memberUser {0}.
BI_2015=isSpeedDial is set False; but SpeedDialContactAddress or speedDialEntry found for memberContact {0}.
BI_2016=User "{0}" is already marked as soft delete.
BI_2017=Invalid delete option - {0}
BI_2018=Delete operation failed for {0} cause {1}
BI_2019=Cannot perform {0} operation , {1} is already asscoiated with {2}
BI_2020=Duplicate entry exists for {0} with the name {1}
BI_2021=Error file creation failed; root cause {0}
BI_2022=TimeZone validation failure; "{0}" is either not a valid timezone or not a valid format
BI_2023={0} transaction timed out or aborted
BI_2024={0} transaction is not completed or is marked for rollback
BI_3000=Mandatory field missing,{0}
BI_2025=Rule validation failed. User {0} with the system administrator role is having a blank password.
BI_2026=Locale {0} is not supported.
BI_2027=The {0} is not a valid G13 preferred language locale, setting to the default value {1}
BI_2028=Rule validation failed. Enterprise user {0}, can not be deleted.
BI_2029=The preferred language is not present in input file, setting to the default value {0}

AUDIT CODE

AUDBI1001=BulkImport job is started. Operation={0}, {1}, {2}, {3}, Result={4}, Configuration={5}
AUDBI1002=Bulkimport job is scheduled successfully. Operation={0}, {1}, {2}, {3}, Result={4}, Configuration={5}
AUDBI1003=Bulkimport job is executed successfully. Operation={0}, {1}, {2}, {3}, Result={4}, Configuration={5}
AUDBI5001=Bulkimport job is not scheduled successfully. Operation={0}, {1}, {2}, {3}, Result={4}, Configuration={5}
AUDBI5002=Bulkimport job execution failed. Operation={0}, {1}, {2}, {3}, Result={4}, Configuration={5}

ERROR CODE

OPUPM0001=Cannot connect to server {0}.
OPUPM0006=Invoking execute method for extended class: {0}
OPUPM0007=Finished execution of execute method for extended class: {0}

UPM_1000=This CsAddress is Enterprise data and cannot be removed
UPM_1001={0} cannot be null or empty
UPM_1002=Object Not Found: {0}
UPM_1003=Persistence Exception: {0}
UPM_1004=User object passed in is stale. Retry operation with fresh user instance.
UPM_1008=Exception: {0}
UPM_1011={0} cannot be null
UPM_1012=Cannot find domain with name={0} and type={1}
UPM_1013=Failed to retrieve domain(s) from database
UPM_1016={0} is Enterprise data and cannot be changed
UPM_1020=User is not marked deleted.
UPM_1021=Exception accessing authorization ID: {0}
UPM_1022=System owned entities must be saved before being assigned to a user
UPM_1025=Incorrect password
UPM_1026=Not able to find the user. Please cancel this operation and retry.
UPM_1027=Cannot delete a System level attribute set: {0}
UPM_1028=No attribute category was found with name: {0}
UPM_1029=Attribute set named {0} was not found
UPM_1030=Could not create {0} scoped attribute set because parent scope({1}) is invalid.
UPM_1031=Attribute {0} was not found in {1} {2}
UPM_1032=AttributeSet {0} does not have a group scope
UPM_1033=AttributeSet {0} does not have the correct scope. AttributeSet scope: {1}, specified scope: {2}
UPM_1034=The attribute {0} does not belong to any set already assigned to the user {1}
UPM_1035=No attribute set was found with container named '{0}' for user '{1}'
UPM_1036=Source attribute set must be System or Group Scope
UPM_1037=Destination attribute set already exists
UPM_1038=Source attribute set is not at System Level and does not have a parent
UPM_1039=Attribute {0} is not cloneable
UPM_1040={0} user of clone attribute sets may not be deleted
UPM_1041=Destination user of clone attribute sets may not have existing attribute sets
UPM_1042=Cannot update soft deleted user.

AUDIT CODE

AUDUPM0001=LoginID={0}, ClientHost={1}, ObjectType={2}, ObjectName={3}, Action={4}, Result={5}, Values: {6}
AUDUPM0022=LoginID={0}, ClientHost={1}, ObjectType={2}, ObjectName={3}, Action={4}, Result={5}, Values: {6}
AUDUPM0023=Deleting dependent object {0} as a result of deleting user {1}
AUDUPM0024=LoginID={0}, ClientHost={1}, ObjectType={2}, ObjectName={3}, Action={4}, Result={5}, Values: {6}
SECUPM0024=Password for user {0} changed by {1}
AUTZUPM001=Authorization error

AUTZUPM002=Null cannot be passed as a parameter to the Authorization Interceptor
AUDDSE0001=LoginID={0}, ClientHost={1}, ObjectType={2}, ObjectName={3}, Action={4}, Result={5}
AUDDSE0002=LoginID={0}, ClientHost={1}, ObjectType={2}, ObjectName={3}, Action={4}, Result={5}

ERROR CODE

UPM_1056=No file to upload
UPM_1057=No header line found
UPM_1058=No terminating quote
UPM_1059=CSV line is too long
UPM_1060=Illegal column name {0}
UPM_1061=Import job {0} queued
UPM_1062=Cancellation request queued
UPM_1063=Error while queueing job
UPM_1064=Validation error
UPM_1065=I/O error {0}
UPM_1066=Import error {0}
UPM_1067=Job must be queued or in progress to be cancelled
UPM_1068=Illegal value {0} for column {1}
UPM_1069=Import completed
UPM_1070=Unsupported property {0}
UPM_1071=Failed to parse XML user: {0}
UPM_1072=Unable to load XML schema for file import
UPM_1073=Internal Error while creating XML parser
UPM_1074=Bad XML root element {0}
UPM_1075=Missing mandatory columns {0}
UPM_1076={0} {1} not found

UPM_1077=Unable to lookup local interface of UPM EJBs
UPM_1078=Unable to lookup remote interface of UPM EJBs
UPM_1079=Contact list not found: {0}
UPM_1080=Contact list owner not found: {0}
UPM_1081=Contact not found: {0}
UPM_1082=Contact owner not found: {0}
UPM_1083=Contact Member not found: {0}
UPM_1084=The contact object being updated does not have a valid id, retrieve the contact through the getContact method prior to updating the contact object and calling updateContact
UPM_1085=Could not retrieve contact {0} with contact list members
UPM_1086=The attributes givenName, surname and displayName are non-nullable fields
UPM_1087=The owner loginName for a public contact must be either CsUser.SYSTEM_USER or null
UPM_1088=A private contact may not updated to be made public
UPM_1089=Database already contains contact with the same displayName and owner {0}
UPM_1090=Database already contains contact list with the same Name and owner {0}
UPM_1092=Contact address category string passed did not match a valid category value

UPM_1093=Contact address type string passed did not match a valid contact type value
UPM_1094=Illegal Argument passed. {0} cannot be passed null or a Collection of size 0.
UPM_1095=Invalid AvPersonID: {0} Could not retrieve the relevant CsPerson
UPM_1096=Cannot update the display name of a contact to be the same as another contact in the users set of owned contacts
UPM_1097=Cannot use negative values for search start index and page offset
UPM_1099=The owner attribute cannot be changed after the entity has been saved
UPM_1100=Contact being updated was not found in users set of owned contacts. {0}

UPM_1101=Database connection error: {0}
UPM_1102={0} with this {1} already exists {2}
UPM_1103=EJBTransactionRolledbackException: {0}
UPM_1104={0} cannot be deleted as it is still referenced by other objects
UPM_1105=Private address cannot be added/updated/deleted through this API
UPM_1106=Could not get the UPM profile from SPM, Invalid Profile
UPM_1107=Could not get the UPM profile from SPM, Invalid Attribute
UPM_1108=Role {0} is already assigned to user {1}
UPM_1109=Trying to add a private contact {0} that is not present in the users {1} set of owned contacts
UPM_1110=Failed to initialize context
UPM_1112=Could not retrieve contact list specified by AvContactListID
UPM_1113=Could not retrieve specified Contact list member
UPM_1114=A null or empty presentity cannot be passed
UPM_1115=A null or empty priority cannot be passed
UPM_1116=A null or empty watcher cannot be passed
UPM_1117=A null rule type cannot be passed
UPM_1120=User Rule cannot be created without a presentity
UPM_1121=Cannot create a rule of type ALL
UPM_1122=Rule {0} could not be recognized
UPM_1124=Trying to override an unoverrideable attribute: {0}
UPM_1125=Attribute {0} is read only and cannot be set again
UPM_1126=Attribute {0} does not meet the minimum length of {1}
UPM_1127=Attribute {0} exceeds the maximum length of {1}
UPM_1128=Method {0} cannot be called on the rule {1}
UPM_1130=A {0} already exists with these criteria
UPM_1131=A {0} does not exist with these criteria
UPM_1132=The correct search class must be used in the findContacts criteria. Valid classes are CsUser, CsContact and CsPerson
UPM_1133=The CsUser class must be used as the criteria search class in the findUsers method
UPM_1134=The CsPerson or CsContact class must be used as the criteria search class in the findContacts method
UPM_1135=A {0} cannot be added to a rule of type {1}
UPM_1136=Invalid class {0}. Valid search classes for searchCsPresACRule extend CsPresAcRule
UPM_1137=Priority cannot be added to a CsPresACRule of type {0}
UPM_1138=The contact {0} already exists in the users contact list
UPM_1139=A {0} must be added to a rule of type {1}
UPM_1140=Contact Address Type must be populated.
UPM_1141=CsPresInfoType with label {0} does not exist.
UPM_1142=The presentity of an existing ACL cannot be updated.

UPM_1143=Role {0} is not assigned to user {1}
UPM_1144=Private contact information has already been created for the contact {0} and the owner {1}
UPM_1145=The entity being updated must have a valid id attribute
UPM_1146=Cannot update the contact person attribute of a CsPrivateContactInfo object to be the same as another contact in the users set of private contact info records
UPM_1147=Cannot find the CsPrivateContactInfo object {0} for the owner {1}
UPM_1148=Cannot add user in its own contact list.
UPM_1149=Rule must contain either a watcher or an externalWatcherURI. Both should not exists together.
UPM_1150=contactPerson for a csPrivateContactInfo record cannot be updated.
UPM_1151=PrivateContactInfo cannot be added for a private contact.
UPM_1152=contactPerson for PrivateContactInfo cannot be null.
UPM_1155=Updating SIP credential failed for user {0}.
UPM_1156=User retrieval failed for user {0}.

RLUPM0002=Rule validatePassword: userPassword cannot be null
RLUPM0003=Rule validatePassword: userPassword is invalid
RLUPM0004=Rule validateLoginName: loginName cannot be null
RLUPM0005=Login name must not contain embedded spaces
RLUPM0006=Rule validateGivenName: givenname cannot be null
RLUPM0007=Rule validatePassword: failed to hash userPassword
RLUPM0008=Rule fillDefaultPersonValues: problem converting display name to ascii
RLUPM0009=Self-deletion not permitted for logged-in user

RLUPM0010=Cannot delete the last Admin User
RLUPM0011=Rule markDeleteEnterprise: Enterprise users can only be deleted from the enterprise source
RLUPM0012=Rule updateCheckAdminUser: Cannot remove the SystemAdmin role from the last Admin User
RLUPM0013=Rule validateDisplayName: displayName cannot be null
RLUPM0014=Rule validateDisplayNameAscii: displayNameAscii cannot be null
RLUPM0015=Rule validateSource: source cannot be null
RLUPM0016=Rule validateAuthenticationType: AuthenticationTypeEnum cannot be null
RLUPM0017=Rule validatePreferredLanguage: preferredLanguage is not valid
RLUPM0018=Rule delete_defaults\: unable to delete property\:
RLUPM0019=Rule deleteContacts\:

RLUPM0020=Rule deleteAddresses\:
RLUPM0021=Rule modifyAdddressFields\:
RLUPM0022=Rule addAddressFields\:
RLUPM0023=Rule addressTypeEnums\:
RLUPM0024=Rule defaultUpdate\: unable to update property\:
RLUPM0025=Rule modifyContacts\:
RLUPM0026=Rule addContacts\:
RLUPM0027=Rule defaultAdd\: unable to set value\:
RLUPM0029=Rule modifySecurityIdentity

RLUPM0031=Rule addSecurityIdentity\: Failed to add securityIdentity\:
RLUPM0032=Rule validateobjectguid: objectguid cannot be empty or null
RLUPM0033=Rule modifyobjectguid: objectguid cannot be be modified

RLUPM0034=Rule validateloginname: samaccountname, uid, userprincipalname cannot both be empty or null
RLUPM0035=Rule validatePassword: password for virtual user cannnot be updated
RLUPM0036=Virtual user cannnot be deleted
RLUPM0037=Rule hibernateValidator: entity validation error: {0}
RLUPM0038=Rule validateACL: A rule must contain at least one CsPresInfoTypeAccess
RLUPM0039=Rule validateACL: A CsPresACRule can only contain one entry for each CsPresInfoType

RLUPM0041=Rule validateCommValues: commProfile type not equal to application system type
RLUPM0042=Rule validateCommValues: application system type for commProfile has been deleted or is not assignable
RLUPM0044=Rule validateCommValues: handle type and subtype do not match
RLUPM0046=Rule validateCommValues: invalid format for handle parameters
RLUPM0047=Rule validateCommValues: numeric handle must have subtype=username
RLUPM0048=Rule validateCommValues: duplicate numeric handle exists for another user
RLUPM0049=Rule validateCommValues: invalid domain for handle with subtype=username or e164

RLUPM0051=Rule validateContactValues: contact list member may not have both the speedDialHandle and speedDialContactAddress set at the same time
RLUPM0052=Rule validateContactValues: A public contact must have the System user as the contact owner
RLUPM0053=Rule validateContactValues: Trying to add a private contact to the contact list of a different user
RLUPM0054=Rule validateACL: A user rule must contain a presentity
RLUPM0055=Rule validateACL: Rule must contain a CsPresACPriority
RLUPM0056=Rule validateACL: Rule must contain a watcher {0}
RLUPM0057=Rule validateAddress: A shared address must be first created through the UserMgmtModel API before being associated with a person
RLUPM0059=Rule validateACL: System Rule creation failed as a System Rule with this priority already exists
RLUPM0060=Rule validateACL: This watcher already exists in another System ACL
RLUPM0061=Rule validateACL: A System Default ACL already exists
RLUPM0062=Rule validateACL: Rule must contain either a watcher or an externalWatcherURI. Both should not exists together.

# Address validation codes
RLUPM0100=Invalid {0} address: cannot be null or empty
RLUPM0101=Invalid {0} address "{1}": must start with "{2}"
RLUPM0102=Invalid SIP address "{0}": cannot contain more than one "@" symbol
RLUPM0103=Invalid SIP address "{0}": SIP userinfo part is not valid
RLUPM0104=Invalid {0} address "{1}": symbol "{2}" appears more then once
RLUPM0105=Invalid SIP address "{0}": SIP header part is not valid
RLUPM0106=Invalid {0} address "{1}": Domain Part length should not be greater then 255 characters
RLUPM0107=Invalid {0} address "{1}": Domain Name Part is not valid
RLUPM0108=Invalid {0} address "{1}": invalid IP address
RLUPM0109=Invalid SIP address "{0}": SIP parameters part is not valid

RLUPM0110=Invalid {0} address "{1}": length should not be greater then 320 characters
RLUPM0111=Invalid {0} address "{1}": must contain symbol "@"
RLUPM0112=Invalid {0} address "{1}": Local Part should be 1-64 characters in length
RLUPM0113=Invalid {0} address "{1}": Local Part should not contain ".." or start/end with "."
RLUPM0114=Invalid {0} address "{1}": Local Part is not valid
RLUPM0115=Invalid {0} address "{1}": invalid format
RLUPM0117=Invalid phone address "{0}": invalid format
RLUPM0118=Invalid XMPP address "{0}": Authority Component part "{1}" is not valid
RLUPM0119=Invalid XMPP address "{0}": Node Identifier part can have only one "@" symbol
RLUPM0120=Invalid XMPP address "{0}": Node Identifier part "{1}" is not valid
RLUPM0121=Invalid XMPP address "{0}": Fragment Identifier part can have only one "#" symbol
RLUPM0122=Invalid XMPP address "{0}": Fragment Identifier part "{1}" is not valid
RLUPM0123=Invalid XMPP address "{0}": Query Component part "{1}" is not valid
RLUPM0124=Invalid XMPP address "{0}": Resource Identifier part "{1}" is not valid
RLUPM0126=Invalid username handle "{0}": format must comply with the userinfo portion of SIP URI

RLUPM0127=Invalid contact address associated with contact "{0}"
RLUPM0128={0} cannot have a private contact as a watcher
RLUPM0129=Invalid E164 address "{0}": must contain 2-15 digits
RLUPM0130=Rule validateCommValues: handle "{0}" already assigned to another user
RLUPM0131=Rule validateCommValues\: E164 handle "{0}" already assigned to current user
RLUPM0132=Rule validateCommValues: cannot move {0} from one {1} to another
RLUPM0133=Rule validateLoginName\: loginName cannot be null or empty
RLUPM0134=Rule validateLoginName: invalid loginName - must have format "username@domain"
RLUPM0135=Rule validateLoginName: invalid loginName - username part may only contain alphanumeric characters, '-', '_', '%', '!', '~', '*', '(', ')', '=', '+', '$' ',', ';', ':' and '.'
RLUPM0136=Rule validateLoginName: invalid loginName - domain name part is not valid
RLUPM0137=Invalid CsPresInfoTypeAccess: Cannot contain a null {0}
RLUPM0138=Invalid Sametime address "{0}": must be 1-255 characters in length
RLUPM0139=Rule validateContactValues: contact list member may not have a negative speed dial value
RLUPM0141=Rule validateContactValues: contact list member that represents an internal user must have a null contact address property
RLUPM0142=Rule validateContactValues\: contact list member that represents an external contact must have a null handle address property
RLUPM0143=Invalid CsAddress "{0}" associated with the CsPrivateContactInfo "{1}", the CsAddress object must be private
RLUPM0144=Rule validateLoginName: the login name {0} already exists in the system. The user may be in the soft deleted users list.
RLUPM0145=Cannot share private contact information between different contacts. The private contact information is already assigned to another contact {0}.
RLUPM0146="{0}" is not a valid G13 Locale. Please enter a valid G13 Locale.
RLUPM0147=Cannot contain more than 7 localized Name.
RLUPM0148="{0}" is not a supported Locale. Please enter supported Locale.

RLUPM0149=Special characters are not allowed for Localized Name.
RLUPM0150=Duplicate Locales are not allowed.
RLUPM0151={0} specified cannot be null or empty.
RLUPM0152=User Authentication type cannot change from Enterprise to Basic.
RLUPM0153=Invalid Language Preference "{0}".
RLUPM0154=Communication Profile Password is required for the SIP handle assigned
RLUPM0155=Invalid pager address "{0}": invalid format
RLUPM0156=Rule validateSurname: surname cannot be null

Appendix E – Must Read

1. **Use FQDN while gaining access to System Manager**

If you access the System Manager using an IP address, the generated certificate does not contains the IP address, but a hostname.
By default, using java implementation, there is an error when you will try to access the REST API as the hostname differs for the one in the certificate.

2. **Reccommendation: Use following namespace to avoid errors**

```
<?xml version="1.0" encoding="UTF-8"?>
     <!--  Root Element 'Users' represent collection of
user (containing 1 or more users)-->
<tns:users xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xml.avaya.com/schema/import
userimport.xsd" >

  <tns:user>
    <authenticationType>Basic</authenticationType>
    <givenName>John</givenName>
    <loginName>jmiller@avaya.com</loginName>
    <surname>Miller</surname>
    <userPassword>mypassword</userPassword>
  </tns:user>
</tns:users>
```

3. **Making retrieved user data compatible for updating**

XML file format contains the user records that you have retrieved using the System Manager Web services. You require this procedure because retrieve users data is not conforming to update user XML Schema Definition. For more information, see XML Schema Definition for Retrieve and Update users : Basic Attributes..

Before you begin

Retrieve the users using the web services.

Procedure

1.Perform the following steps:

a.Locate the following content in the generated XML:

<tns:users xmlns:tns="http://xml.avaya.com/schema/import"

xmlns:ns3="http://xml.avaya.com/schema/import1"

xmlns:ns4="http://xml.avaya.com/schema/deltaImport"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xml.avaya.com/schema/import userimport.xsd">

b.Modify tns:users to tns:deltaUserList.

c.Remove tns="http://xml.avaya.com/schema/import".

d.Modify ns4="http://xml.avaya.com/schema/deltaImport" to tns="http://xml.avaya.com/schema/deltaImport"

e.Modify xsi:schemaLocation="http://xml.avaya.com/schema/import userimport.xsd"> to xsi:schemaLocation="http://xml.avaya.com/schema/deltaImport userdeltaimport.xsd ">

After you modify the XML file as instructed in Step b through Step e, the content in Step a changes to:

<tns:deltaUserList xmlns:ns3="http://xml.avaya.com/schema/import1"

xmlns:tns="http://xml.avaya.com/schema/deltaImport"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xml.avaya.com/schema/deltaImport userdeltaimport.xsd ">

2.Replace all instances of:

◦<tns:user> with <tns:userDelta>

◦</tns:user> with </tns:userDelta>

◦<tns:users> with <tns:deltaUserList>

◦</tns:users> with </tns:deltaUserList>

Next Steps

You can now make the updates in the XML and update the user attributes in the database using web services.

4. **Not to use namespace as xPath query**

It is recommended not to use namespace as the xPath queries. The namespace are dynamically formed to well-form the XML response.Business logic in your client should not depend upon these namespace.

5. Webservice Request should be generated as ‘well-formed XML’ since SMGR do validate request XML, it’s not mandatory to have namespace as ‘ns’.

Appendix F – XML Schema Definitions

Note: You cannot use the following characters as is in the XML file. To use the characters in the import of XML files, make the following modifications:

•Less-than character (<) as <

•Ampersand character (&) as &

•Greater-than character (>) as >

•Double-quote character (") as "

•Apostrophe or single-quote character (') as '

When you copy the XML schema from the document you must take care of the line breaks.

XML Schema Definition for retrieving and creating User(s): Basic Attributes

Note: The attributes highlighted in Grey are introduced in System Manager 6.3.4 and 6.3.8.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema xmlns:tns="http://xml.avaya.com/schema/import"
xmlns:ext="http://xml.avaya.com/schema/import"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://xml.avaya.com/schema/import" version="3.0">
      <xs:annotation>
            <xs:documentation xml:lang="en"This Schema defines schema for
bulk import and export of Users. Root Element 'Users' represent collection
of user (containing 1 or more users)/xs:documentation>
      </xs:annotation>
      <xs:element name="secureStore" type="tns:xmlSecureStore"/>
      <xs:element name="user" type="tns:xmlUser"/>
      <xs:element name="users">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element name="secureStore"
type="tns:xmlSecureStore" minOccurs="0"/>
                        <xs:element name="user" type="tns:xmlUser"
minOccurs="0" maxOccurs="unbounded"/>
                  </xs:sequence>
            </xs:complexType>
      </xs:element>
```

```
      <xs:complexType name="xmlUser">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                        --- UserOrganizationDetails: This defines the tenant
and its organizational hierarchy like name of site, team, and department.
                       --- UserProvisionRuleName : This defines the rules
for auto provisioning of user.
                        ---authenticationType: This defines the type of
authentication the user undergoes at runtime to gain access to the system.
Possible values are basic or enterprise. Although this attribute is
mandatory, the value entered is ignored by System Manager. It sets the
value depending on how the user is provisioned in the system. On user
addition via web service, it will always be set as "basic". On user update
via web service, the current value in System Manager will not be updated.
                        ---description:A text description of the user. Human
readable description of this user instance.
                        ---displayName:The localized name of a user to be
used when displaying. It will typically be the localized full name. This
value may be provisioned from the user enterprise directory entry. If it
does not exist, synchronization rules can be used to populate it for other
fields e.g. Surname, GivenName, or LoginName.
                        ---displayNameAscii:This corresponds to the Console
attribute-Endpoint Display Name.The full text name of the user represented
in ASCII. It is used to support display (e.g. endpoints) that cannot
handle localized text
                        ---dn:The distinguished name of the user. The DN is a
sequence of relative distinguished names (RDN) connected by commas. An RDN
is an attribute with an associated value in the form of attribute=value,
normally expressed in a UTF-8 string format.The dn can be used to identify
the user and may be used for authentication subject mapping. Note the dn
is changeable.
                        ---isDuplicatedLoginAllowed:A boolean indicator
showing whether this user is allowed a duplicate concurrent logins.A true
stipulates that the user is allow to have duplicate logins. Default value
is true.
                        ---isEnabled:A boolean indicator showing whether or
not the user is active. Users with AuthenticationType=Basic will fail if
this value is false.This attribute can be used to disable access between
login attempts. A running session login will not be revocable.
Alternatively the administrator can always modify the password to disable
the user from logging in.A true stipulates this is an active user, a false
used for a disabled  user. Default value is false.
                        ---isVirtualUser:A boolean indicator showing whether
or not the record is being used for a non-human entity such as an
application, service, software agent, etc. This is to be used where the
entity will behave as a user and needs to have subset of the user profile
populated. If the entity does not behave as a user and has a different
trust relationship e.g. a trust certificate it should not be treated as a
virtual user. A virtual user can represent an Avaya or external non-human
entity. This attribute is provided as a convenience to track such
accounts.A true stipulates this is a virtual users, a false is used for
```

```
human users. Default value is false.
                      ---givenName:The first name of the user.
                      ---honorific:The personal title used to address a
user. This is typically a social title and not the work title which is
contained in the title attribute. This attribute can map to
"PersonalTitle".
                      ---loginName:This is the unique system login name
given to the user. It can take the form of username@domain or just
username.This may vary across customers. It can be used to help provision
default user handles in the CSHandle table.The username is an alphanumeric
value that must comply with the userinfo related portion of a URI as
described in rfc2396. However, it is further restricted as ASCII
characters with only the "_" and "." special characters supported.  This
is the rfc2798 "uid" attribute.
                     ---newLoginName: This is the unique system login name
given to the user. It can take the form of username@domain or just
username.This may vary across customers. This attribute help you to
specify a new login name incase of updating user login name.
                      ---employeeNo:Employee number of user.
                      ---department:Department of employee.
                      ---organization:Organization of employee.
                      ---middleName:The middle name of the user
                      ---managerName:Text name of the user manager. This is
a free formed field and does not require the user manager to also be a
user of the solution. This attribute was requested to support reporting
needs.
                      ---preferredGivenName:The preferred first name of the
user.
                      ---preferredLanguage:The individual's preferred
written or spoken language.Values will conform to rfc4646 and the reader
should refer to rfc4646 for syntax.  This format uses the ISO standard
Language (ISO-639) and region (ISO-3166) codes In the absence of a value
the client locale should be used, if no value is set, en-US should be
defaulted.
                      ---source:Free format text field that identifies the
entity that created this user record.  The format of this field will be
either a IP Address/Port or a name representing an enterprise LDAP or
Avaya.
                      ---sourceUserKey:The key of the user from the source
system. If the source is an Enterprise Active Directory server, this value
with be the objectGUID.
                      ---status:This information is to help manage
provisioning activities such as correcting or completing the provisioning
of a user instance. It can also signify that approval is needed
(PENDINGAUTHZ) before a user account is sufficiently configured to be a
valid user (PROVISIONED). Possible Values:
AUTHPENDING;PENDINGAUTHZ;PROVISIONED
                      ---suffix:The text appended to a name e.g. Jr., III.

                      ---surname:The user's last name, also called the
family name.
```

```
                        ---timeZone:The preferred time zone of the user. For
example: (-12:0)International Date Line West.The application consuming
this information would need to know how to translate e.g. in Java it would
be TimeZone.getTimeZone("Europe/Moscow");In the absence of a value the
local services timezone will be used.
                        ---title:The job function of a person in their
organizational context.
                        ---userName:This is the username portion of the
loginName field. It is an alphanumeric value that must comply with the
userinfo related portion of a URI as described in rfc2396. However, it is
further restricted as ASCII characters with only the "_" and "." special
characters supported.  This is the rfc2798 "uid" attribute.
                        ---userPassword:The encrypted password for this
user's account.A null password is used when the user is authenticated by
the enterprise such as with a separate source such as the enterprise LDAP.
                        ---commPassword:The encrypted  subscriber or
communication password with which the user logs can use to authentication
with on to any CommProfile SIP and non SIP.  This attribute is meant to be
a shared across different communication profiles and thus different
communication services.
                        ---userType:This enumerates the possible primary user
application types. A User can be associated with multiple user types.
Possible values are ADMINISTRATOR;COMMUNICATION
USER;AGENT;SUPERVISOR;RESIDENT EXPERT;SERVICE TECHNICIAN;LOBBY PHONE
                        ---roles:Text name of a role.This value needs to pre-
exist in SMGR DB
                        ---localizedNames:localized name of user.
                        ---address:The address of the user.
                        ---securityIdentity:The SecurityIdentity is used to
hold any additional identities for a user that can be used for
authentication such as their loginName, Kerberos account name, or their
X509 certificate name.
                        ---ownedContactLists:It is a collection of internal
or external contacts. ContactList is owned by a specific user and has a
name that a unique name within the context of its owner.
                        ---ownedContacts:It represents a non Avaya
application user (external) contact. Contacts can be collected together
along with User entities into a contact list. Contacts can be created by
an administrator or an end user.
                        ---presenceUserACL:These are personal rules defined
by presentities themselves on who can monitor their presence information.
There may be several entries in the list for a given presentity, each
entry corresponding to one watcher.
                        ---commProfileSet:A user will have a default
commprofile set.A commprofile set can exist without any handles or
commprofiles referencing it. I.e. you can create a commprofile set without
needing to also create either a handle or a commprofile.A commprofile set
can contain multiple commprofiles, but only one of each specific type.
This is enforced by having the CommProfile uniqueness constraint include
type, commprofile_set_id.
                  </xs:documentation>
```

```
            </xs:annotation>
            <xs:sequence>
<           <xs:element name="UserOrganizationDetails"
type="tns:UserOrganizationDetailsType" maxOccurs="1" minOccurs="0" />
              <xs:element name="UserProvisionRules" minOccurs="0">
                <xs:complexType>
                   <xs:sequence>
                            <xs:element name="UserProvisionRuleName"
type="xs:string" minOccurs="0" maxOccurs="unbounded" />
                        </xs:sequence>
                        </xs:complexType>
                       </xs:element>
                  <xs:element name="authenticationType" type="xs:string"/>
                  <xs:element name="description" type="xs:string"
minOccurs="0"/>
                  <xs:element name="displayName" type="xs:string"
minOccurs="0"/>
                  <xs:element name="displayNameAscii" type="xs:string"
minOccurs="0"/>
                  <xs:element name="dn" type="xs:string" minOccurs="0"/>
                  <xs:element name="isDuplicatedLoginAllowed"
type="xs:boolean" minOccurs="0"/>
                  <xs:element name="isEnabled" type="xs:boolean"
minOccurs="0"/>
                  <xs:element name="isVirtualUser" type="xs:boolean"
minOccurs="0"/>
                  <xs:element name="givenName" type="xs:string"/>
      <xs:element name="givenNameAscii" type="xs:string" minOccurs="0"
                       maxOccurs="1" />
                  <xs:element name="honorific" type="xs:string"
minOccurs="0"/>
                  <xs:element name="loginName">
                       <xs:simpleType>
                            <xs:restriction base="xs:string">
                                 <xs:maxLength value="128"/>
                            </xs:restriction>
                       </xs:simpleType>
                  </xs:element>
                  <!- updated for 6.3.8 -->
                  <xs:element name="newLoginName" minOccurs="0"
maxOccurs="1">
                    <xs:simpleType>
                      <xs:restriction base="xs:string">
                           <xs:maxLength value="128" />
                      </xs:restriction>
                    </xs:simpleType>
                  </xs:element>
                  <xs:element name="employeeNo" type="xs:string"
                       minOccurs="0" maxOccurs="1">
                  </xs:element>
                  <xs:element name="department" type="xs:string"
```

```
minOccurs="0"
                      maxOccurs="1">
                </xs:element>
                <xs:element name="organization" type="xs:string"
                      minOccurs="0" maxOccurs="1">
                </xs:element>
                <xs:element name="middleName" type="xs:string"
minOccurs="0"/>
                <xs:element name="managerName" type="xs:string"
minOccurs="0"/>
                <xs:element name="preferredGivenName" type="xs:string"
minOccurs="0"/>
                <xs:element name="preferredLanguage" type="xs:string"
minOccurs="0"/>
                <xs:element name="source" type="xs:string" minOccurs="0"
                      maxOccurs="1"/>
                <xs:element name="sourceUserKey" type="xs:string"
minOccurs="0"
                      maxOccurs="1"/>
                <xs:element name="status" type="xs:string" minOccurs="0"/>
                <xs:element name="suffix" type="xs:string" minOccurs="0"/>
                <xs:element name="surname" type="xs:string"/>
                <xs:element name="surnameAscii" type="xs:string"
minOccurs="0"
                      maxOccurs="1" />
                <xs:element name="timeZone" type="xs:string"
minOccurs="0"/>
                <xs:element name="title" type="xs:string" minOccurs="0"/>
                <xs:element name="userName" type="xs:string" minOccurs="0"
                      maxOccurs="1"/>
                <xs:element name="userPassword" type="xs:string"
minOccurs="0"/>
                <xs:element name="commPassword" type="xs:string"
minOccurs="0"/>
                <xs:element name="userType" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
                <xs:element name="roles" minOccurs="0">
                      <xs:complexType>
                            <xs:sequence>
                                  <xs:element name="role" type="xs:string"
minOccurs="0" maxOccurs="unbounded"/>
                            </xs:sequence>
                      </xs:complexType>
                </xs:element>
                <xs:element name="localizedNames"
type="tns:xmLocalizedNames" minOccurs="0" maxOccurs="1"/xs:element>
                <xs:element name="address" type="tns:xmlAddress"
minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="securityIdentity"
type="tns:xmlSecurityIdentity" minOccurs="0" maxOccurs="unbounded"/>
                <!-- Contact list Entries -->
```

```
                  <xs:element name="ownedContactLists" minOccurs="0">
                        <xs:complexType>
                              <xs:sequence>
                                    <xs:element name="contactList"
type="tns:xmlContactList"/>
                              </xs:sequence>
                        </xs:complexType>
                  </xs:element>
                  <xs:element name="ownedContacts" minOccurs="0">
                        <xs:complexType>
                              <xs:sequence>
                                    <xs:element name="contact"
type="tns:xmlContact" maxOccurs="unbounded"/>
                              </xs:sequence>
                        </xs:complexType>
                  </xs:element>
                  <!-- Presence ACL User Entries -->
                  <xs:element name="presenceUserACL"
type="tns:xmlPresUserACLEntryType" minOccurs="0" maxOccurs="unbounded"/>
                  <xs:element name="commProfileSet"
type="tns:xmlCommProfileSetType" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
      </xs:complexType>
      <xs:complexType name="xmlSecurityIdentity">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                        ---SecurityIdentity:Represents the possible external
identities that a user may have for the purpose of authentication. The
type and format of an identity depends on the external Identity Provider
and can include X.509 certificates or Kerberos user accounts
                        ---identity:The unique external identity of the user.
This is a free text field and no format is enforced. The format will
depend on the identity type. Kerberos user account can take the form of:
username@domainName
e.g. jsmith@acme.org
                        ---realm:The name of the security domain that this
identity is valid in.
                        ---type:The text representation of the type of
identity. Possible values are: principalname,X509 and Kerberos
                  </xs:documentation>
            </xs:annotation>
            <xs:sequence>
                  <xs:element name="identity" type="xs:string"/>
                  <xs:element name="realm" type="xs:string" minOccurs="0"/>
                  <xs:element name="type" type="xs:string"/>
            </xs:sequence>
      </xs:complexType>
      <xs:complexType name="xmlPresInfoTypeAccessType">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                        ---PresInfoTypeAccess: For the purpose of access
```

```
control, presence information is partitioned into several areas called
Presence Info Types. Examples of Presence Info Types would be "Telephony
Presence", "Instant Messaging Presence", "Calendar Presence", or "Full
Presence".
                      ---infoType:This defines the different classes of
presence information.
                      ---access:Presence access type possible values:
ALLOW, BLOCK,
                </xs:documentation>
            </xs:annotation>
            <xs:sequence>
                <xs:element name="infoType"
type="tns:xmlPresInfoTypeType"/>
                <xs:element name="access" type="xs:string" minOccurs="0"/>
            </xs:sequence>
      </xs:complexType>
      <xs:complexType name="xmlPresACRuleType">
            <xs:annotation>
                <xs:documentation xml:lang="en">
                      ---ACRuleType:This  contains rules that are similar
to a User ACL in the sense that its entries define access between
individual presentities and watchers. However this rule is managed by the
administrator as opposed to presentities themselves. Entries of Enforced
User ACL can also be defined with different priorities. Entries with
higher priority will have more weight than entries with lower priority.
                      ---infotypeaccess:This is a link between acl entries,
presence info types, and access actions.
                </xs:documentation>
            </xs:annotation>
            <xs:sequence>
                <xs:element name="infoTypeAccess"
type="tns:xmlPresInfoTypeAccessType" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
      </xs:complexType>
      <xs:complexType name="xmlPresUserACLEntryType">
            <xs:annotation>
                <xs:documentation xml:lang="en">
                            ---UserACLEntry:These are personal rules
defined by presentities themselves on who can monitor their presence
information. There may be several entries in the list for a given
presentity, each entry corresponding to one watcher.
                            ---watcherLoginName: LoginName,if the watcher
is a user.
                            ---watcherDisplayName:DisplayName,if the
watcher is a contact.
                </xs:documentation>
            </xs:annotation>
            <xs:complexContent>
                <xs:extension base="tns:xmlPresACRuleType">
                      <xs:sequence>
                            <xs:choice>
```

```
                                    <xs:element name="watcherLoginName"
type="xs:string" minOccurs="0"/>
                                    <xs:element name="watcherDisplayName"
type="xs:string" min
Occurs="0"/>
                              </xs:choice>
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="xmlPresInfoTypeType">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                        ---PresInfoType:Entries that define the difference
classes of presence information.
                        ---label:A unique string that names this info type
(e.g. "Telephony Presence").
                        ---filter:Internal definition of which part of
presence information is covered by this info type. The value of this field
should be treated as opaque string; it is maintained and used only by
Presence services.
                        ---specFlags:This field is empty for regular info
types, but for special info types it contains a comma-separated list of
keywords that identify these types. In this version only FULL that
represents full presence information is supported.
            </xs:documentation>
            </xs:annotation>
            <xs:sequence>
                  <xs:element name="label" type="xs:string"/>
                  <xs:element name="filter" type="xs:string"/>
                  <xs:element name="specFlags" type="xs:string"
minOccurs="0"/>
            </xs:sequence>
      </xs:complexType>
      <!-- Contact List entries -->
      <xs:complexType name="xmlContactList">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                        ---ContactList:The ContactList is a collection of
personal or public groups containing external contacts and/or Avaya users.
                        ---name:The text name of the list. This in the
context of the owner must be unique.
                        ---description:A free text description of this
member.
                        ---isPublic:Defines if the contact is public or
personal. Default = false.
                        ---members:Represents the list of users or contacts
that belong to contact list
                        ---contactListType:Specifies the type categorizing
this list.
                  </xs:documentation>
```

```
            </xs:annotation>
            <xs:sequence>
                  <xs:element name="name" type="xs:string"/>
                  <xs:element name="description" type="xs:string"
minOccurs="0"/>
                  <xs:element name="isPublic" type="xs:boolean"/>
                  <xs:element name="members" type="tns:xmlContactListMember"
minOccurs="0" maxOccurs="unbounded"/>
                  <xs:element name="contactListType" type="xs:string"/>
            </xs:sequence>
      </xs:complexType>
      <xs:complexType name="xmlContactListMember">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                        ---ContactListMember:It supports many to many
relationship between user, Contact and ContactList.
                        ---memberContact:This represents the name of the
Contact.A ContactListMember can either be a Contact or User
                        ---speedDialContactAddress:A Contact Address added as
a favorite entry
                        ---memberUser:This represents the loginname of the
User.A ContactListMember can either be a Contact or User
                        ---speedDialHandle:A handle added as a favorite entry
                        ---isFavorite:A boolean indicator that reflects
whether this contact is a favorite entry. If true, the value of entryindex
would show which position to place this entry in any display.
                        ---isSpeedDial:Each contact list member can also be
flagged as a favorite (a.k.a. speed dial)
                        ---speedDialEntry:For either a presence buddy or
favorite entry, a specific communication address to use can be pointed to.
                        ---isPresenceBuddy:Each contact list member can also
be flagged as a presence buddy
                        ---label:A free text short word or phrase for
classifying this contact list member.
                        ---altLabel:A free text short word or phrase for
classifying this contact. This is similar to label, but it is used to
store alternate language representations.
                        ---description:A free text description of this
member.
                  </xs:documentation>
            </xs:annotation>
            <xs:sequence>
                  <xs:choice>
                        <xs:sequence>
                              <xs:element name="memberContact"
type="xs:string" minOccurs="0"/>
                              <xs:element name="speedDialContactAddress"
type="tns:xmlContactAddress" minOccurs="0"/>
                        </xs:sequence>
                        <xs:sequence>
                              <xs:element name="memberUser" type="xs:string"
```

```
minOccurs="0"/>
                              <xs:element name="speedDialHandle"
type="tns:xmlHandle" minOccurs="0"/>
                        </xs:sequence>
                  </xs:choice>
                  <xs:element name="isFavorite" type="xs:boolean"/>
                  <xs:element name="isSpeedDial" type="xs:boolean"/>
                  <xs:element name="speedDialEntry" type="xs:int"
minOccurs="0"/>
                  <xs:element name="isPresenceBuddy" type="xs:boolean"/>
                  <xs:element name="label" type="xs:string" minOccurs="0"/>
                  <xs:element name="altLabel" type="xs:string"
minOccurs="0"/>
                  <xs:element name="description" type="xs:string"
minOccurs="0"/>
                  <xs:element name="priorityLevel" type="xs:int"
minOccurs="0"/>
            </xs:sequence>
      </xs:complexType>
      <xs:complexType name="xmlContactAddress">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                        ---address:A fully qualified URI for interacting with
this contact. Any addresses added to this table should contain a qualifier
e.g. sip, sips, tel, mailto. The address should be syntactically valid
based on the qualifier. It must be possible to add via the GUI and
Interface. The application must do validation.
                        ---altLabel:A free text description for classifying
this contact. This is similar to ContactLabel, but it is used to store
alternate language representations.
                        ---contactCategory:It represents the category of this
entry e.g. Home, Office, Mobile.
                        ---contactType:It represents the type of contact this
entry e.g. phone, SIP, IM, Email.
                        ---label:A free text description for classifying this
contact.
                  </xs:documentation>
            </xs:annotation>
            <xs:sequence>
                  <xs:element name="address" type="xs:string"/>
                  <xs:element name="altLabel" type="xs:string"
minOccurs="0"/>
                  <xs:element name="contactCategory" type="xs:string"/>
                  <xs:element name="contactType" type="xs:string"/>
                  <xs:element name="label" type="xs:string" minOccurs="0"/>
            </xs:sequence>
      </xs:complexType>
      <xs:complexType name="xmlAddress">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                              ---addressType:Specifies the role of the
```

```
address. Examples:  Home, business.
                            ---name:The Name property defines the unique
label by which the address is known. Default format for user specific
address should include user name place address type.
                            ---building:The name or other designation of a
structure
                            ---localityName:The name of a locality, such as
a city,    county or other geographic region.
                            ---postalCode:A code used by postal services to
route mail to a destination. In the United States this is the zip code.
                            ---room:Name or designation of a room.
                            ---stateOrProvince:The full name of a state or
province.
                            ---country:A country.
                            ---street:The physical address of the object
such as an address for package delivery
                            ---postalAddress:A free formed text area for
the complete physical delivery address. It may be used in place of the
specific fields in this table.
                            ---isPrivate:A boolean indicator to specify if
this address could be shared across multiple users.True is private, false
is sharable. Default is false.
                 </xs:documentation>
            </xs:annotation>
            <xs:sequence>
                 <xs:element name="addressType" type="xs:string"/>
                 <xs:element name="name" type="xs:string"/>
                 <xs:element name="building" type="xs:string"
minOccurs="0"/>
                 <xs:element name="localityName" type="xs:string"
minOccurs="0"/>
                 <xs:element name="postalCode" type="xs:string"
minOccurs="0"/>
                 <xs:element name="room" type="xs:string" minOccurs="0"/>
                 <xs:element name="stateOrProvince" type="xs:string"
minOccurs="0"/>
                 <xs:element name="country" type="xs:string"
minOccurs="0"/>
                 <xs:element name="street" type="xs:string" minOccurs="0"/>

                 <!-- Additional Attribute Support LDAP -->
                 <xs:element name="businessphone" type="xs:string"
minOccurs="0"/>
                 <xs:element name="otherbusinessphone" type="xs:string"
minOccurs="0"/>
                 <xs:element name="fax" type="xs:string" minOccurs="0"/>
                 <xs:element name="homephone" type="xs:string"
minOccurs="0"/>
                 <xs:element name="otherhomephone" type="xs:string"
minOccurs="0"/>
                 <xs:element name="mobilephone" type="xs:string"
```

```
minOccurs="0"/>
                 <xs:element name="othermobilephone" type="xs:string"
minOccurs="0"/>
                 <xs:element name="pager" type="xs:string" minOccurs="0"/>
                 <xs:element name="pager2" type="xs:string" minOccurs="0"/>
                 <!-- Additional Attribute Support LDAP - End -->


                 <xs:element name="postalAddress" minOccurs="0">
                      <xs:simpleType>
                           <xs:restriction base="xs:string">
                                <xs:maxLength value="1024"/>
                           </xs:restriction>
                      </xs:simpleType>
                 </xs:element>
                 <xs:element name="isPrivate" type="xs:boolean"
minOccurs="0"/>
            </xs:sequence>
      </xs:complexType>
      <xs:complexType name="xmlContact">
            <xs:annotation>
                 <xs:documentation xml:lang="en">
                      ---Contact:An entity that represents a non Avaya
application user (external) contact. Contacts can be collected together
along with User entities into a contact list. Contacts can be created by
an administrator or an end user. Contacts have name attributes, and owner,
and can be public or personal.A contact also includes one or more contact
addresses that can be used for establishing an interaction with the
contact. Contacts can be designated as being a user presence buddy or
added as a favorite entry (i.e. speed dial).
                      ---company:The organization that the contact belongs
to.
                      ---description:A free text field containing human
readable text providing information on this entry.
                      ---displayName:The localized name of a contact to be
used when displaying. It will typically be the localized full name. This
value may be provisioned from the user enterprise directory entry. If it
does not exist, synchronization rules can be used to populate it for other
fields e.g. Surname, GivenName, or LoginName.
                      ---displayNameAscii:The full text name of the contact
represented in ASCII. It is used to support display (e.g. endpoints) that
cannot handle localized text.
                      ---dn:The distinguished name of the user. The DN is a
sequence of relative distinguished names (RDN) connected by commas. An RDN
is an attribute with an associated value in the form of attribute=value,
normally expressed in a UTF-8 string format.The dn can be used to uniquely
identify this record. Note the dn is changeable.
                      ---givenName:The first name of the contact.
                      ---initials:Initials of the contact
                      ---middleName:The middle name of the contact.
                      ---preferredGivenName:The nick name of the contact.
```

```
                        ---preferredLanguage:The individual's preferred
written or spoken language. Values will conform to rfc4646 and the reader
should refer to rfc4646 for syntax.  This format uses the ISO standard
Language (ISO-639) and region (ISO-3166) codes In the absence of a value
the client locale should be used, if no value is set, en-US should be
defaulted.
                        ---isPublic:Defines if the contact is public or
personal. Default = false.
                        ---source:Free format text field that identifies the
entity that created this user record.  The format of this field will be
either a IP Address/Port or a name representing an enterprise LDAP or
Avaya.
                        ---sourceUserKey:The key of the user from the source
system. If the source is an Enterprise Active Directory server, this value
with be the objectGUID.
                        ---suffix:The text appended to a name e.g. Jr., III.
                        ---surname:The user's last name, also called the
family name.
                        ---title:The job function of a person in their
organizational context.Examples: supervisor, manager
                        ---ContactAddress:Represents a contact address.
                        ---addresses:A fully qualified URI for interacting
with this contact. Any addresses added to this table should contain a
qualifier e.g. sip, sips, tel, mailto. The address should be syntactically
valid based on the qualifier. It must be possible to add via the GUI and
Interface. The application must do validation.
                 </xs:documentation>
           </xs:annotation>
           <xs:sequence>
                 <xs:element name="company" type="xs:string"
minOccurs="0"/>
                 <xs:element name="description" type="xs:string"
minOccurs="0"/>
                 <xs:element name="displayName" type="xs:string"/>
                 <xs:element name="displayNameAscii" type="xs:string"/>
                 <xs:element name="dn" type="xs:string" minOccurs="0"/>
                 <xs:element name="givenName" type="xs:string"/>
      <xs:element name="givenNameAscii" type="xs:string" minOccurs="0"
                        maxOccurs="1" />
                 <xs:element name="initials" type="xs:string"
minOccurs="0"/>
                 <xs:element name="middleName" type="xs:string"
minOccurs="0"/>
                 <xs:element name="preferredGivenName" type="xs:string"
minOccurs="0"/>
                 <xs:element name="preferredLanguage" type="xs:string"
minOccurs="0"/>
                 <xs:element name="isPublic" type="xs:boolean"/>
                 <xs:element name="source" type="xs:string"/>
                 <xs:element name="sourceUserKey" type="xs:string"/>
                 <xs:element name="suffix" type="xs:string" minOccurs="0"/>
```

```
                <xs:element name="surname" type="xs:string"/>
                <xs:element name="surnameAscii" type="xs:string"
minOccurs="0" maxOccurs="1" />
                <xs:element name="title" type="xs:string" minOccurs="0"/>
                <xs:element name="ContactAddress"
type="tns:xmlContactAddress" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element name="addresses" type="tns:xmlAddress"
minOccurs="0" maxOccurs="unbounded"/>
           </xs:sequence>
     </xs:complexType>
     <xs:complexType name="xmlHandle">
           <xs:annotation>
                <xs:documentation xml:lang="en">
                      ---HandleName:This is the name given to the user to
allow communication to be established with the user. It is an alphanumeric
value that must comply with the userinfo related portion of a URI as
described in rfc2396. However, it is further restricted as ASCII
characters with only the + prefix to signify this is an E.164 handle and
"_" and "." special characters supported.Note, the handle plus domain can
be used to construct a user Address of Record.
                      ---handleType:The value reflecting the type of handle
this is. Possible values are  sip,smtp,ibm,and xmpp.
                      ---handleSubType:This is an additional qualify on the
handle type to help specify which private subsystem this handle belongs
to.Possible values are
e164,username,msrtc,googletalk,jabber,ibmsametime,lotousnotes,msexchage.
                      ---domainName:The text name of the domain.
                </xs:documentation>
           </xs:annotation>
           <xs:sequence>
                <xs:element name="handleName" type="xs:string"/>
                <xs:element name="handleType" type="xs:string"/>
                <xs:element name="handleSubType" type="xs:string"/>
                <xs:element name="domainName" type="xs:string"
minOccurs="0"/>
           </xs:sequence>
     </xs:complexType>
     <xs:complexType name="xmlCommProfileType">
           <xs:sequence>
                <xs:element name="commProfileType" type="xs:string"/>
                <xs:element name="commProfileSubType" type="xs:string"
maxOccurs="1" minOccurs="0"/xs:element>
           </xs:sequence>
     </xs:complexType>
     <xs:complexType name="xmlCommProfileSetType">
           <xs:annotation>
                <xs:documentation xml:lang="en">
                                  ---commProfileSetName:The unique name of
this CommProfile. This is used to aid in the lookup of the CommProfile
                                  ---isPrimary:A boolean value indicating
whether Communication profile is primary or not.
```

```
                              </xs:documentation>
            </xs:annotation>
            <xs:sequence>
                  <xs:element name="commProfileSetName" type="xs:string"/>
                  <xs:element name="isPrimary" type="xs:boolean"/>
                  <xs:element name="handleList" minOccurs="0">
                        <xs:annotation>
                              <xs:documentation xml:lang="en">
                                          ---handleList:List of handles
                                          ---handle:A user address of record
(AOR) is represented by a combination of a handle (userpart) and domain
(domainpart).The entity that contains the userinfo part of an address that
can be used to establish an interaction with a user. A user can have
multiple handles.
                                    </xs:documentation>
                        </xs:annotation>
                        <xs:complexType>
                              <xs:sequence>
                                    <xs:element name="handle"
type="tns:xmlHandle" maxOccurs="unbounded"/>
                              </xs:sequence>
                        </xs:complexType>
                  </xs:element>
                  <xs:element name="commProfileList" minOccurs="0">
                        <xs:annotation>
                              <xs:documentation xml:lang="en">
                                    ---commProfileList:List of communication
profile
                                    ---commProfile:A communication profile  is
an entity that supports communication interactions established through
Avaya Communication Services. A communication profile is used to represent
a user subscription to a product specific communication subsystem and
contains its specific configuration needs for the user.
                              </xs:documentation>
                        </xs:annotation>
                        <xs:complexType>
                              <xs:sequence>
                                    <xs:element name="commProfile"
type="tns:xmlCommProfileType" maxOccurs="unbounded"/>
                              </xs:sequence>
                        </xs:complexType>
                  </xs:element>
            </xs:sequence>
      </xs:complexType>
      <xs:complexType name="ForgeinCommProfileType">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                        ---ForeignCommProfileType:A ForeignCommProfile is
used to represent a user address information when routing to that address
is controlled by a non Avaya system or Avaya applications not using this
User CIM to populate their handles and aliases.
```

```
                        ---csEncryptionKeyId:The service will be responsible
for using this key and the secure store library API when encrypting and
decrypting the password field when respectively set or accessed by an
authorized client.
                        ---servicePassword:.Password is an optional field if
an Avaya application needs to authenticate with the foreign service. This
field will be stored using a reversible encryption algorithm. The key will
be specified through a reference to EncryptionKey.
                  </xs:documentation>
            </xs:annotation>
            <xs:complexContent>
                  <xs:extension base="ext:xmlCommProfileType">
                        <xs:sequence>
                              <xs:element name="csEncryptionKeyId"
type="xs:long" minOccurs="0"/>
                              <xs:element name="servicePassword"
type="xs:string" minOccurs="0"/>
                              <xs:element name="serviceData" type="xs:string"
minOccurs="0"/>
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="xmlSecureStore">
            <xs:annotation>
                  <xs:documentation xml:lang="en">
                        ---SecureStore:The Entity is used to persist the
secure store. Each application can have a single secure store and the
application name used to represent the secure store must be unique.

                        ---passwordEncrypted :This section gets generated by
the encryption util which encrypts the userPassword and CommPassword.
                  </xs:documentation>
            </xs:annotation>
            <xs:sequence>
                  <xs:element name="secureStoreData"
type="xs:base64Binary"/>
                  <xs:element name="passwordEncrypted" type="xs:boolean"/>
            </xs:sequence>
      </xs:complexType>


      <xs:complexType name="xmlLocalizedName">
            <xs:sequence>
                  <xs:element name="locale" type="xs:string" minOccurs="1"
                        maxOccurs="1">
                  </xs:element>
                  <xs:element name="name" type="xs:string" minOccurs="1"
maxOccurs="1"/xs:element>
            </xs:sequence>
      </xs:complexType>
```

```
      <xs:complexType name="xmLocalizedNames">
            <xs:sequence>
                  <xs:element name="localizedName"
type="tns:xmlLocalizedName" minOccurs="0" maxOccurs="7"/xs:element>
            </xs:sequence>
      </xs:complexType>
      <!—To support multi-tenancy →
      <xs:complexType name="UserOrganizationDetailsType">
      <xs:sequence>
 <xs:element name="tenant" maxOccurs="1" minOccurs="1">
  <xs:complexType>
     <xs:attribute name="name" type="xs:string" use="required" />
     <xs:attribute name="createTenantIfNotAlreadyPresent" type="xs:boolean"
use="required" />
   </xs:complexType>
  </xs:element>
  <xs:element name="organizationUnitLevelOne" type="xs:string"
maxOccurs="1" minOccurs="0" />
   <xs:element name="organizationUnitLevelTwo" type="xs:string"
maxOccurs="1" minOccurs="0" />
   <xs:element name="organizationUnitLevelThree" type="xs:string"
maxOccurs="1" minOccurs="0" />
  </xs:sequence>
 </xs:complexType>
</xs:schema>
```

XML Schema Definition for updating User(s): Basic Attributes

Note: The attributes highlighted in Grey are introduced in System Manager 6.3.4 and 6.3.8.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<xs:schema xmlns:delta="http://xml.avaya.com/schema/deltaImport"
xmlns:base="http://xml.avaya.com/schema/import"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://xml.avaya.com/schema/deltaImport" version="1.0">

      <xs:import namespace="http://xml.avaya.com/schema/import"
schemaLocation="userimport.xsd"/>


      <xs:element name="userDelta" type="delta:xmlUserDelta"/>
```

```
      <xs:element name="deltaUserList" type="delta:xmlDeltaUserList"/>

      <xs:complexType name="xmlDeltaUserList">
            <xs:sequence>
                  <xs:element name="secureStore"
type="base:xmlSecureStore"></xs:element>
                  <xs:element name="userDelta" type="delta:xmlUserDelta"
minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
      </xs:complexType>

      <xs:complexType name="xmlUserDelta">
            <xs:sequence>
            <!—To support Tenant →
           <xs:element name="UserOrganizationDetails"
type="base:UserOrganizationDetailsType" maxOccurs="1" minOccurs="0" />
            <!—To support user provisioning rule →
            <xs:element name="UserProvisionRules" minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                 <xs:element name="UserProvisionRuleName" type="xs:string"
minOccurs="0" maxOccurs="unbounded" />
                </xs:sequence>
              </xs:complexType>
             </xs:element>
                <xs:element name="authenticationType"
                      type="xs:string" minOccurs="0" maxOccurs="1" />
                  <xs:element name="description" type="xs:string"
                      minOccurs="0" />
                  <xs:element name="displayName" type="xs:string"
                      minOccurs="0" />
                  <xs:element name="displayNameAscii" type="xs:string"
                      minOccurs="0" />
                  <xs:element name="dn" type="xs:string" minOccurs="0" />
                  <xs:element name="isDuplicatedLoginAllowed"
                      type="xs:boolean" minOccurs="0" />
                  <xs:element name="isEnabled" type="xs:boolean"
minOccurs="0"
                      maxOccurs="1" />
                  <xs:element name="isVirtualUser" type="xs:boolean"
                      minOccurs="0" />
                  <xs:element name="givenName" type="xs:string"
maxOccurs="1"
                      minOccurs="0" />
      <xs:element name="givenNameAscii" type="xs:string" maxOccurs="1"
                      minOccurs="0" />
                  <xs:element name="honorific" type="xs:string"
minOccurs="0" />
                  <xs:element name="loginName" type="xs:string"
maxOccurs="1"
                      minOccurs="1" />
```

```
                  <!— To update user login name -->
                  <!- Updated for 6.3.8: newLoginName: This is the unique
system login name given to the user. It can take the form of
username@domain or just username.This may vary across customers. This
attribute help you to specify a new login name incase of updating user
login name.  -->
                   <xs:element name="newLoginName" minOccurs="0"
maxOccurs="1">
                    <xs:simpleType>
                      <xs:restriction base="xs:string">
                           <xs:maxLength value="128" />
                      </xs:restriction>
                    </xs:simpleType>
                  </xs:element>
                  <xs:element name="middleName" type="xs:string"
                        minOccurs="0" />
                  <xs:element name="managerName" type="xs:string"
                        minOccurs="0" />
                  <xs:element name="preferredGivenName" type="xs:string"
                        minOccurs="0" />
                  <xs:element name="preferredLanguage" type="xs:string"
                        minOccurs="0" />
                  <xs:element name="source" type="xs:string" minOccurs="0"
                        maxOccurs="1" />
                  <xs:element name="sourceUserKey" type="xs:string"
                        minOccurs="0" maxOccurs="1" />
                  <xs:element name="status" type="xs:string"
                        minOccurs="0" />
                  <xs:element name="suffix" type="xs:string"
minOccurs="0" />
                  <xs:element name="surname" type="xs:string" minOccurs="0"
                        maxOccurs="1" />
                  <xs:element name="surnameAscii" type="xs:string"
minOccurs="0" maxOccurs="1" />
                  <xs:element name="timeZone" type="xs:string" minOccurs="0"
/>
                  <xs:element name="title" type="xs:string" minOccurs="0" />
                  <xs:element name="userName" type="xs:string" maxOccurs="1"
                        minOccurs="0" />
                  <xs:element name="userPassword" type="xs:string"
                        minOccurs="0" />
                  <xs:element name="commPassword" type="xs:string"
                        minOccurs="0" />
                  <xs:element name="userType" type="xs:string"
                        minOccurs="0" maxOccurs="unbounded" />
                  <xs:element name="roles" minOccurs="0">
                        <xs:complexType>
                              <xs:sequence>
                                    <xs:element name="role" type="xs:string"
                                          minOccurs="0"
maxOccurs="unbounded" />
```

```
                              </xs:sequence>
                        </xs:complexType>
                  </xs:element>
                  <xs:element name="address" type="base:xmlAddress"
                        minOccurs="0" maxOccurs="unbounded" />
                  <xs:element name="securityIdentity"
                        type="base:xmlSecurityIdentity" minOccurs="0"
maxOccurs="unbounded" />
                  <!-- Contact list Entries -->
                  <xs:element name="ownedContactLists" minOccurs="0"
                        maxOccurs="1">
                        <xs:complexType>
                              <xs:sequence>
                                    <xs:element name="contactList"
                                          type="base:xmlContactList"
maxOccurs="1" />
                              </xs:sequence>
                        </xs:complexType>
                  </xs:element>
                  <xs:element name="ownedContacts" minOccurs="0">
                        <xs:complexType>
                              <xs:sequence>
                                    <xs:element name="contact"
type="base:xmlContact"
                                          maxOccurs="unbounded" />
                              </xs:sequence>
                        </xs:complexType>
                  </xs:element>
                  <!-- Presence ACL User Entries -->
                  <xs:element name="presenceUserACL"
                        type="base:xmlPresUserACLEntryType" minOccurs="0"
                        maxOccurs="unbounded" />
                  <xs:element name="commProfileSet"
                        type="base:xmlCommProfileSetType"
maxOccurs="unbounded" minOccurs="0">
                  </xs:element>
            </xs:sequence>
      </xs:complexType>
</xs:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Session Manager

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  xmlns:smgr="http://xml.avaya.com/schema/import"
```

```
      targetNamespace="http://xml.avaya.com/schema/import_sessionmanager"
                 elementFormDefault="qualified">

<!-- This is the XML schema for the "Session Manager Profile".  It
     defines this profile inside of an XML document that defines
     a user record (see userimport.xsd) -->

<xsd:import namespace="http://xml.avaya.com/schema/import"
            schemaLocation="userimport.xsd"/>

<xsd:complexType name="SessionManagerCommProfXML">

    <xsd:complexContent>
        <xsd:extension base="smgr:xmlCommProfileType" >

      <xsd:sequence>
        <!--
         The following attributes are the names of objects that must
         already be administered in System Manager before performing
         the user import.

         The relative order here cannot be changed because it would
         break backwards compatibility with existing XML documents
         that could be used for an import.
        -->

        <!-- Name of the primary Session Manager (required) -->
        <xsd:element name="primarySM" type="xsd:string" minOccurs="0" />

        <!--  Name of the secondary Session Manager (optional) -->
          <xsd:element name="secondarySM" type="xsd:string"
minOccurs="0" />

          <!-- Name of the Termination Application Sequence (optional)
               - administered under
                Session Manager / Application Configuration / Application
Sequences -->
          <xsd:element name="terminationAppSequence" type="xsd:string"
minOccurs="0" />

          <!-- Name of the Origination Application Sequence (optional)
               - administered under
                 Session Manager / Application Configuration / Application
Sequences -->
          <xsd:element name="originationAppSequence" type="xsd:string"
minOccurs="0" />

          <!-- Name of the Conference Factory Set (optional)
              - administered under
                Session Manager / Application Configuration / Conference
Factories  -->
```

```
          <xsd:element name="confFactorySet" type="xsd:string"
minOccurs="0" />

          <!-- Name of the Survivability Server (optional)
             - usually the name of a Branch Session Manager, but can be
               any non-CM SIP Entity -->
          <xsd:element name="survivabilityServer" type="xsd:string"
minOccurs="0" />

          <!-- Name of the Home Location (required)
                 - administered under Routing / Locations -->
          <xsd:element name="homeLocation" type="xsd:string"
minOccurs="1" />

          <!-- The maximum number of endpoints that can be simultaneously
registered
               using this Session Manager Profile. (optional)
               - The value is an integer between 1 and 10 and
                 defaults to 1 if not specified.  -->
          <xsd:element name="maxSimultaneousDevices" minOccurs="0">
           <xsd:simpleType>
                      <xsd:restriction base="xsd:integer">
                            <xsd:minInclusive value="1" />
                            <xsd:maxInclusive value="10" />
                      </xsd:restriction>
                 </xsd:simpleType>
          </xsd:element>

        <!--
          If true, new registrations will be blocked for this Session
Manager
          Profile if the maximum number of simultaneously registered
endpoints
          (see "maxSimultaneousDevices" above) is currently registered.  If
          false, an existing registration will be terminated to allow a new
          registration for this Session Manager Profile. (optional)
           - the value defaults to false if not specified
                         -->
          <xsd:element name="blockNewRegistrationWhenMaxActive"
minOccurs="0">
                 <xsd:simpleType>
                      <xsd:restriction base="xsd:boolean" />
                 </xsd:simpleType>
          </xsd:element>
                <!--
                  Flag to enable/disable the offline Call Logging feature
for the SIP users
                         -->
          <xsd:element name="enabledisablecalllog" minOccurs="0">
                 <xsd:simpleType>
                      <xsd:restriction base="xsd:boolean" />
```

```
                  </xsd:simpleType>
          </xsd:element>

          <!-- Name of the Emergency Termination Application Sequence
(optional)
                - administered under
                 Session Manager / Application Configuration / Emergency
Application Sequences -->
          <xsd:element name="emergencyTerminationAppSequence"
type="xsd:string"  minOccurs="0" />

          <!-- Name of the Emergency Origination Application Sequence
(optional)
                - administered under
                  Session Manager / Application Configuration / Emergency
Application Sequences -->
          <xsd:element name="emergencyOriginationAppSequence"
type="xsd:string" minOccurs="0" />

            <!-- Policy Type (required only when policy based SM
assignment global settings is enabled) -->
        <xsd:element name="policyType" type="xsd:string" default="fixed"
minOccurs="0" />

         <!-- Name of the third Session Manager -->
        <xsd:element name="thirdSM" type="xsd:string" minOccurs="0" />

         <!-- Name of the fourth Session Manager -->
        <xsd:element name="fourthSM" type="xsd:string" minOccurs="0" />

          <!-- Name of the primary fixed-region (mandatory only in case of
fixed-region policy type minimum occurance is kept zero but is validated
in AsmComProfDataRules, as this field is not applicable in case of fixed
policy type ) -->
        <xsd:element name="primaryFixedRegion" type="xsd:string"
minOccurs="0" />

          <!-- Name of the secondary fixed-region (applicable only in case
of fixed-region policy type, optional) -->
        <xsd:element name="secondaryFixedRegion" type="xsd:string"
minOccurs="0" />

        <!-- Name of the third fixed-region (applicable only in case of
fixed-region policy type, optional) -->
        <xsd:element name="thirdFixedRegion" type="xsd:string"
minOccurs="0" />

         <!-- Name of the fourth fixed-region (applicable only in case of
fixed-region policy type, optional) -->
        <xsd:element name="fourthFixedRegion" type="xsd:string"
minOccurs="0" />
```

```
     </xsd:sequence>

        </xsd:extension>
    </xsd:complexContent>
 </xsd:complexType>


</xsd:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Breeze

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                xmlns:smgr="http://xml.avaya.com/schema/import"
                targetNamespace="http://xml.avaya.com/schema/import_ce"
                elementFormDefault="qualified">

<!-- This is the XML schema for the "CE Profile".  It
     defines this profile inside of an XML document that defines
     a user record (see userimport.xsd) -->

<xsd:import namespace="http://xml.avaya.com/schema/import"
            schemaLocation="userimport.xsd"/>

<xsd:complexType name="CeCommProfXML">
    <xsd:complexContent>
        <xsd:extension base="smgr:xmlCommProfileType">

      <xsd:sequence>
        <!--
         The following attributes are the names of objects that must
         already be administered in System Manager before performing
         the user import.

         The relative order here cannot be changed because it would
         break backwards compatibility with existing XML documents
         that could be used for an import.
        -->

        <!--  Name of the secondary Session Manager (optional) -->
          <xsd:element name="serviceProfile" type="xsd:string"
minOccurs="1" />
```

```
      </xsd:sequence>

         </xsd:extension>
    </xsd:complexContent>
 </xsd:complexType>

</xsd:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Endpoint

Note: The attributes highlighted in Grey are introduced in System Manager 7.0.

Note: The attributes marked in Green are introduced in System Manager 7.0.1.

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

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:one="http://xml.avaya.com/schema/import" elementFormDefault="qualified"

targetNamespace="http://xml.avaya.com/schema/import_csm_cm"
xmlns:csm="http://xml.avaya.com/schema/import_csm_cm">

<xs:import namespace="http://xml.avaya.com/schema/import"
schemaLocation="userimport.xsd"/>


<!--Changes in xsd file need to generate jaxb src using this xsd-->

<xs:complexType name="xmlStationProfile">

        <xs:complexContent>

        <xs:extension base="one:xmlCommProfileType" >

                        <xs:sequence>


                                <!-- CM Name as it appears under
'Applications/Application Management/Entities -->

                                <xs:element name="cmName" type="xs:string"
maxOccurs="1" minOccurs="1"/>

                                <xs:element name="prefHandleId" type="xs:string"
maxOccurs="1" minOccurs="0"/>

                                <!-- 'true' if already created extension is to be used.
'false' if available extension is to be used. -->

                                <xs:element name="useExistingExtension"
```

```
type="xs:boolean" maxOccurs="1" minOccurs="0"/>

                                <!-- Extension Range which will be used to create Station
using available extension within given range -->
                                <xs:element name="extensionRange" maxOccurs="1"
minOccurs="0">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="([0-9]+([\.\-][0-
9]+)*)|([0-9]+([\.\-][0-9]+)*:[0-9]+([\.\-][0-9]+)*)"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>

                                <!-- Station extension number that need to be assigned to
the user. -->
                                <xs:element name="extension" maxOccurs="1"
minOccurs="1">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="([0-9]+([\.\-][0-
9]+)*)|[nN][eE][xX][tT]"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>

                                <!-- Template name to be used to create station. Values
defined in Template will be used if not provided. -->
                                <xs:element name="template" type="xs:string"
maxOccurs="1" minOccurs="0"/>

                                <!-- Specifies the set type of the station  -->
                                <xs:element name="setType" type="xs:string"
maxOccurs="1" minOccurs="0"/>

                                <!-- Security code for station. Value can be digit only.  -->
```

```
                                <xs:element name="securityCode" maxOccurs="1"
minOccurs="0">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[0-9]*"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>



                                <!-- Valid values for port   -->
                                <!--01 to 64  First and second numbers are the cabinet
number  -->
                                <!--A to E  Third character is the carrier  -->
                                <!--01 to 20  Fourth and fifth characters are the slot
number  -->
                                <!--01 to 32  Sixth and seventh characters are the circuit
number  -->
                                <!--x or X  Indicates that there is no hardware associated
with the port assignment since the switch was set up, and the administrator expects
that the extension would have a non-IP set. Or, the extension had a non-IP set, and it
dissociated. Use x for Administered WithOut Hardware (AWOH) and Computer
Telephony (CTI) stations, as well as for SBS Extensions.  -->
                                <!--IP  Indicates that there is no hardware associated with
the port assignment since the switch was set up, and the administrator expects that the
extension would have an IP set. This is automatically entered for certain IP station set
types, but you can enter for a DCP set with softphone permissions. This changes to the
s00000 type when the set registers.  -->
                                <xs:element name="port" type="xs:string" maxOccurs="1"
minOccurs="0" />


                                <!-- Whether the station should be deleted if it unassigned
from the user. -->
                                <xs:element name="deleteOnUnassign"
type="xs:boolean" maxOccurs="1" minOccurs="0"/>


                                <!-- Whether the endpoint name on CM should be
overridden with the value in User. -->
                                <xs:element name="overRideEndpointName"
```

```
type="xs:boolean" maxOccurs="1" minOccurs="0"/>

                                <!-- true/false for dualRegistration. -->
                                <xs:element name="dualRegistration" type="xs:boolean"
maxOccurs="1" minOccurs="0"/>

                                <!-- true/false for Enhanced Callr-Info display for 1-line
phones -->
                                <xs:element name="enhCallrInfodisplay"
type="xs:boolean" maxOccurs="1" minOccurs="0"/>

                                <!-- true/false to enable/disable lock messages feature.
-->
                                <xs:element name="lockMessages" type="xs:boolean"
maxOccurs="1" minOccurs="0" />

                                <!-- A coverage path is a prioritized sequence of
extensions to which your voice system will route an unanswered call. -->
                                <!-- Valid values: CM 5.2 - Path Number between 1-2000,
time of day table, t1-t999, or blank. -->
                                <!-- Valid values: CM 6.0 - Path Number between 1-9999,
time of day table, t1-t999, or blank. -->
                                <xs:element name="coveragePath1" maxOccurs="1"
minOccurs="0">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="([1-9]{0})|(t[1-9]
[0-9]{0,2})|([1-9][0-9]{0,3})"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>

                                <!-- A coverage path is a prioritized sequence of
extensions to which your voice system will route an unanswered call. -->
                                <!-- Valid values: CM 5.2 - Path Number between 1-2000,
time of day table, t1-t999, or blank. -->
                                <!-- Valid values: CM 6.0 - Path Number between 1-9999,
```

```
time of day table, t1-t999, or blank. -->
                <xs:element name="coveragePath2" maxOccurs="1"
minOccurs="0">
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                        <xs:pattern value="([1-9]{0})|(t[1-9]
[0-9]{0,2})|([1-9][0-9]{0,3})"/>
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>


                <!-- The extension the system should hunt to for this
telephone when the telephone is busy. A station hunting chain can be created by
assigning a hunt-to station to a series of telephones.  -->
                <xs:element name="huntToStation" type="xs:string"
maxOccurs="1" minOccurs="0" />


                <!-- Provides for partitioning of attendant groups and/or
stations and trunk groups. -->
                <!-- Typically this is used for multiple tenants in a building
or multiple departments within a company or organization. -->
                <!-- Valid values: 1 to 250 when TN is ON in special
application and 1 to 100 o.w. -->
                <xs:element name="tn" maxOccurs="1" minOccurs="0">
                <xs:simpleType>
                                <xs:restriction base="xs:int">
                                        <xs:minInclusive value="1" />
                                        <xs:maxInclusive value="250" />
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>


                <!-- Typically this is used for multiple tenants in a building
or multiple departments within a company or organization. -->
                <!-- Typically this is used for multiple tenants in a building
or multiple departments within a company or organization. -->
```

```
<!-- Valid values: 0 to 995 -->
<xs:element name="cor" maxOccurs="1" minOccurs="0">
    <xs:simpleType>
    <xs:restriction base="xs:int">
        <xs:minInclusive value="0"/>
        <xs:maxInclusive value="995"/>
    </xs:restriction>
    </xs:simpleType>
</xs:element>

<!-- Class of Service lets you define groups of users and
control those groups' access to features -->
<!-- Valid values: 1 to 15 -->
<xs:element name="cos" maxOccurs="1"
minOccurs="0">
    <xs:simpleType>
        <xs:restriction base="xs:int">
            <xs:minInclusive value="0" />
            <xs:maxInclusive value="15" />
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<xs:element name="xmobileType" maxOccurs="1"
minOccurs="0" >
    <xs:simpleType>
        <xs:restriction base="xs:string">
        <xs:enumeration value="EC500"/>
        <xs:enumeration value="DECT"/>
        <xs:enumeration value="IPDECT"/>
        <xs:enumeration value="PHS"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

```
<xs:element name="mappingMode" maxOccurs="1"
minOccurs="0" >
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="termination"/>
<xs:enumeration value="origination"/>
<xs:enumeration value="both"/>
<xs:enumeration value="none"/>
</xs:restriction>
</xs:simpleType>
</xs:element>

<xs:element name="configurationSet" maxOccurs="1"
minOccurs="0" >
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:pattern value="|[1-9]|[0-9][1-
9]"/>
</xs:restriction>
</xs:simpleType>
</xs:element>

<xs:element name="mobilityTrunkGroup" maxOccurs="1"
minOccurs="0" >
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:pattern value="aar|ars|[1-9]|[1-
9][0-9]|[1-9]([0-9]){2}|[1]([0-9]){3}|2000"/>
</xs:restriction>
</xs:simpleType>
</xs:element>

<xs:element name="dialPrefix" maxOccurs="1"
minOccurs="0" >
<xs:simpleType>
```

```
                                        <xs:restriction base="xs:string">
                                                <xs:pattern value="[0-9/*#]{0,4}"/>
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <xs:element name="cellPhoneNumber" maxOccurs="1"
minOccurs="0" >
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:pattern value="[0-9]{0,15}"/>
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <xs:element name="musicSource" maxOccurs="1"
minOccurs="0" >
                                <xs:simpleType>
                                        <xs:restriction base="xs:int">
                                                <xs:minInclusive value="1" />
                                                <xs:maxInclusive value="250" />
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <xs:element name="tests" type="xs:boolean"
maxOccurs="1" minOccurs="0" />

                        <xs:element name="dataModule" type="xs:boolean"
maxOccurs="1" minOccurs="0" />

                        <!-- Controls the behavior of speakerphones. -->
                        <xs:element name="speakerphone" maxOccurs="1"
minOccurs="0">
                                <xs:simpleType>
```

```
                                        <xs:restriction base="xs:string">
                                                <xs:enumeration value="none"/>
                                                <xs:enumeration value="1-way"/>
                                                <xs:enumeration value="2-way"/>
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <!-- The language that displays on stations -->
                        <!-- Time of day is displayed in 24-hour format (00:00 -
23:59) for all languages except English, which is displayed in 12-hour format (12:00
a.m. to 11:59 p.m.). -->
                        <!-- unicode: Displays English messages in a 24-hour
format . If no Unicode file is installed, displays messages in English by default. -->
                        <xs:element name="displayLanguage" maxOccurs="1"
minOccurs="0">
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:enumeration value="english"/>
                                                <xs:enumeration value="french"/>
                                                <xs:enumeration value="italian"/>
                                                <xs:enumeration value="spanish"/>
                                                <xs:enumeration value="unicode"/>
                                                <xs:enumeration
value="unicode2"/>
                                                <xs:enumeration
value="unicode3"/>
                                                <xs:enumeration
value="unicode4"/>
                                                <xs:enumeration value="user-
defined"/>
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <!-- Defines the personalized ringing pattern for the
```

```
station.
                                        Personalized Ringing allows users of some
telephones to have one of 8 ringing patterns for incoming calls.
                                        For virtual stations, this field dictates the ringing
pattern on its mapped-to physical telephone.
                                -->
                                <!-- L = 530 Hz, M = 750 Hz, and H = 1060 Hz -->
                                <!-- Valid Entries  Usage
                                        1  MMM (standard ringing)
                                        2  HHH
                                        3  LLL
                                        4  LHH
                                        5  HHL
                                        6  HLL
                                        7  HLH
                                        8  LHL
                                 -->
                                <xs:element name="personalizedRingingPattern"
maxOccurs="1" minOccurs="0">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:int">
                                                        <xs:minInclusive value="1" />
                                                        <xs:maxInclusive value="8" />
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>



                                <!-- The Message Lamp Extension associated with the
current extension -->
                                <xs:element name="messageLampExt" maxOccurs="1"
minOccurs="0">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[0-9]+([\.\-][0-
```

```
9]+)*"/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>


                                <!-- Enables or disables the mute button on the station.
-->
                                <xs:element name="muteButtonEnabled"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <!--
                                    When used with Multi-media Call Handling,
indicates which extension is
                                    assigned to the data module of the multimedia
complex. Users can dial
                                    this extension to place either a voice or a data call,
and voice
                                    conversion, coverage, and forwarding apply as if
the call were made to
                                    the 1-number.
                                -->
                                <!--
                                    Valid Entry Usage A valid BRI data extension For
MMCH, enter the
                                    extension of the data module that is part of this
multimedia complex.
                                    H.323 station extension For 4600 series IP
Telephones, enter the
                                    corresponding H.323 station. For IP Softphone,
enter the corresponding
                                    H.323 station. If you enter a value in this field, you
can register
                                    this station for either a road-warrior or
telecommuter/Avaya IP Agent
                                    application. blank Leave this field blank for single-
connect IP
                                    applications.
```

```
-->
<xs:element name="mediaComplexExt" maxOccurs="1"
minOccurs="0" >
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="([1-9]{0})|[0-9]+
([\.\-][0-9]+)*"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<!-- Whether this is IP soft phone. -->
<xs:element name="ipSoftphone" type="xs:boolean"
maxOccurs="1" minOccurs="0" />

<!--
    Survivable GK Node Name Identifies the
existence of other H.323
    gatekeepers located within gateway products that
offer survivable call
    features. For example, the MultiTech MVPxxx-AV
H.323 gateway family
    and the SLS function within the H.248 gateways.
When a valid IP node
    name is entered into this field, Communication
Manager adds the IP
    address of this gateway to the bottom of the
Alternate Gatekeeper List
    for this IP network region. As H.323 IP stations
register with
    Communication Manager, this list is sent down in
the registration
    confirm message. This allows the IP station to use
the IP address of
    this Survivable Gatekeeper as the call controller of
last resort to
    register with. Available only if the station type is an
H.323 station
```

(46xxor 96xx models).

Valid Entry Usage

Valid IP node name Any valid previously-administered IP node name.

blank There are no external gatekeeper nodes within a customer's network. This is the default value.

-->

<xs:element name="survivableGkNodeName" type="xs:string" maxOccurs="1" minOccurs="0" />

<!--

Sets a level of restriction for stations to be used with the

survivable dial plan to limit certain users to only to certain types

of calls. You can list the restriction levels in order from the most

restrictive to least restrictive. Each level assumes the calling

ability of the ones above it. This field is used by PIM module of the

Integrated Management to communicate with the Communication Manager

administration tables and obtain the class of service information. PIM

module builds a managed database to send for Standard Local

Survivability (SLS) on the H.248 gateways. Available for all analog

and IP station types.

Valid Entries Usage

emergency This station can only be used to place emergency calls.

internal This station can only make intra-switch calls. This is the default.

local This station can only make calls that are defined as locl, op, svc, or hnpa in the Survivable Gateway Call

Controller's routing tables.

toll This station can place any national toll calls that are defined as fnpa or natl on the Survivable Gateway Call Controller's routing tables.

unrestricted This station can place a call to any number defined in the Survivable Gateway Call Controller's routing tables. Those strings marked as deny are also denied to these users.

```
-->
<xs:element name="survivableCOR" maxOccurs="1" minOccurs="0">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:enumeration value="emergency"/>
            <xs:enumeration value="internal"/>
            <xs:enumeration value="local"/>
            <xs:enumeration value="toll"/>
            <xs:enumeration value="unrestricted"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<!--
    Designates certain telephones as not being allowed to receive incoming
    trunk calls when the Media Gateway is in survivable mode. This field
    is used by the PIM module of the Integrated Management to successfully
    interrogate the Communication Manager administration tables and obtain
    the class of service information. PIM module builds a managed database
    to send for SLS on the H.248 gateways. Available for all analog and IP
    station types.
```

```
                                        Valid Entry              Usage
                                                true                    Allows this station to
be an incoming trunk destination while the Media Gateway is running in survivability
mode. This is the default.
                                                false                   Prevents this station
from receiving incoming trunk calls when in survivable mode.


                                -->
                                <xs:element name="survivableTrunkDest"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <!-- Enter the complete Voice Mail Dial Up number. -->
                                <xs:element name="voiceMailNumber" maxOccurs="1"
minOccurs="0" >


                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[(0-9)(*)(#)
(~mwWps)]{0,24}"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>


                                <!-- Analog telephones only. -->
                                <!--
                                Valid entries           Usage
                                                true            Enter true if this telephone
is not located in the same building with the system. If you enter true, you must complete
R Balance Network.
                                                false           Enter false if the telephone
is located in the same building with the system.
                                 -->
                                <xs:element name="offPremisesStation"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <!-- If a second line on the telephone is administered on
```

```
the I-2 channel, enter analog. Otherwise, enter data module if applicable or none. -->
                                <xs:element name="dataOption" maxOccurs="1"
minOccurs="0">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:enumeration value="analog"/>
                                                        <xs:enumeration value="data-
module"/>
                                                        <xs:enumeration value="none"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>



                                <xs:element name="displayModule" type="xs:boolean"
maxOccurs="1" minOccurs="0" />


                                <!-- if led or neon then messageLampExt should be
enable otherwise its blank -->
                                <xs:element name="messageWaitingIndicator"
maxOccurs="1" minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:enumeration value="led"/>
                                                        <xs:enumeration value="neon"/>
                                                        <xs:enumeration value="none"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>


                                <!-- Enter true to use this station as an endpoint in a
remote office configuration. -->
                                <xs:element name="remoteOfficePhone"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <!-- Defines the source for Leave Word Calling (LWC)
```

```
messages. -->
                                <!--
                                Valid entries                    Usage
                                        audix                            If LWC is attempted,
the messages are stored in AUDIX.
                                        spe                              If LWC is attempted,
the messages are stored in the system processing element (spe).
                                        none                             If LWC is attempted,
the messages are not stored.


                                 -->
                                <xs:element name="lwcReception" maxOccurs="1"
minOccurs="0">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:enumeration value="audix"/>
                                                        <xs:enumeration value="msa"/>
                                                        <xs:enumeration value="spe"/>
                                                        <xs:enumeration value="none"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>


                                <!--
                                        Enter true to allow internal telephone users to
leave short LWC messages
                                        for this extension. If the system has hospitality,
enter true for
                                        guest-room telephones if the extension designated
to receive failed
                                        wakeup messages should receive LWC messages
that indicate the wakeup
                                        calls failed. Enter true if LWC Reception is audix.
                                -->
                                <xs:element name="lwcActivation" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
```

```
                                <xs:element name="lwcLogExternalCalls"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <xs:element name="cdrPrivacy" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
                                <xs:element name="redirectNotification"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <xs:element name="perButtonRingControl"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <xs:element name="bridgedCallAlerting"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <xs:element name="bridgedIdleLinePreference"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <xs:element name="confTransOnPrimaryAppearance"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <xs:element name="customizableLabels"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <xs:element name="expansionModule" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
                                <xs:element name="ipVideoSoftphone"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <xs:element name="activeStationRinging"
maxOccurs="1" minOccurs="0">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                <xs:enumeration value="single"/>
                                                <xs:enumeration value="continuous"/>
                                                <xs:enumeration value="if-busy-single"/>
                                                <xs:enumeration value="silent"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>


                                <!-- Defines how call rings to the telephone when it is on-
hook. -->
                                <!--
```

Valid entries Usage

continuous Enter continuous to cause all calls to this telephone to ring continuously.

if-busy-single Enter if-busy-single to cause calls to this telephone to ring continuously when the telephone is off-hook and idle and calls to this telephone to

receive one ring cycle and then ring silently when the telephone is off-hook and active.

silent-if-busy Enter silent-if-busy to cause calls to ring silently when this station is busy.

single Enter single to cause calls to this telephone to receive one ring cycle and then ring silently.

-->

<xs:element name="idleActiveRinging" type="xs:string" maxOccurs="1" minOccurs="0" /> <!-- not found in xhtml -->

<!-- Must be set to true when the Type field is set to H.323. -->

<xs:element name="switchhookFlash" type="xs:boolean" maxOccurs="1" minOccurs="0" />

<!-- If this field is true, the short switch-hook flash (50 to 150) from a 2500-type set is ignored. -->

<xs:element name="ignoreRotaryDigits" type="xs:boolean" maxOccurs="1" minOccurs="0" />

<!--

H.320 Conversion - Valid entries are true and false (default). This field is

optional for non-multimedia complex voice stations and for Basic

multimedia complex voice stations. It is mandatory for Enhanced

multimedia complex voice stations. Because the system can only handle

a limited number of conversion calls, you might need to limit the

number of telephones with H.320 conversion.

```
Enhanced multimedia
                                        complexes must have this flag set to true.
                                -->
                                                        <xs:element
name="h320Conversion" type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <!--
                                        The service link is the combined hardware and
software multimedia
                                        connection between an Enhanced mode
complex's H.320 DVC system and the
                                        Avaya DEFINITY Server which terminates the
H.320 protocol. A service
                                        link is never used by a Basic mode complex H.320
DVC system.
                                        Connecting a service link will take several
seconds. When the service
                                        link is connected, it uses MMI, VC and system
timeslot resources. When
                                        the service link is disconnected it does not tie up
any resources.
                                        The
                                        Service Link Mode can be administered as either
'as-needed' or
                                        'permanent' as described below: - As-Needed -
Most non-call center
                                        multimedia users will be administered with this
service link mode. The
                                        as-needed mode provides the Enhanced
multimedia complex with a
                                        connected service link whenever a multimedia call
is answered by the
                                        station and for a period of 10 seconds after the
last multimedia call
                                        on the station has been disconnected. Having the
service link stay
                                        connected for 10 seconds allows a user to
disconnect a multimedia call
                                        and then make another multimedia call without
```

```
having to wait for the
                                        service link to disconnect and re-establish. -
Permanent - Multimedia
                                        call center agents and other users who are
constantly making or
                                        receiving multimedia calls might want to be
administered with this
                                        service link mode. The permanent mode service
link will be connected
                                        during the station's first multimedia call and will
remain in a
                                        connected state until the user disconnects from
their PC's multimedia
                                        application or the Avaya DEFINITY Server
restarts. This provides a
                                        multimedia user with a much quicker video cut-
through when answering a
                                        multimedia call from another permanent mode
station or a multimedia
call that has been early answered. ● Multimedia Mode - There are two
                                        multimedia modes, Basic and Enhanced, as
                                -->
                                <xs:element name="serviceLinkMode" maxOccurs="1"
minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                <xs:enumeration value="as-needed"/>
                                                <xs:enumeration value="permanent"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>

                                <!--
                                        There are two multimedia modes, Basic and
Enhanced, as described
                                        below:
                                        Basic - A Basic multimedia complex consists of a
```

BRI-connected multimedia-equipped PC and a non-BRI-connected multifunction telephone set. When in Basic mode, users place voice calls at the multifunction telephone and multimedia calls from the multimedia equipped PC. Voice calls will be answered at the multifunction telephone and multimedia calls will alert first at the PC and if unanswered will next alert at the voice station if it is administered with H.320 enabled. A Basic mode complex has limited multimedia feature capability.

Enhanced - An Enhanced multimedia complex consists of a BRI-connected multimedia-equipped PC and a non-BRI-connected multifunction telephone. The Enhanced mode station acts as though the PC were directly connected to the multifunction telephone; the service link provides the actual connection between the Avaya DEFINITY Server and the PC. Thus, voice and multimedia calls are originated and received at the telephone set. Voice and multimedia call status are also displayed at the telephone set. An Enhanced mode station allows multimedia calls to take full advantage of most call control features

```
-->
<xs:element name="multimediaMode" maxOccurs="1" minOccurs="0" >
        <xs:simpleType>
                <xs:restriction base="xs:string">
                <xs:enumeration value="basic"/>
```

```
                                        <xs:enumeration value="enhanced"/>
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <!-- Controls the auditing or interrogation of a served
user's message waiting indicator (MWI).
                        Valid entries                   Usage
                                fp-mwi                          Use if the station is a
served user of an fp-mwi message center.
                                qsig-mwi                        Use if the station is a
served user of a qsig-mwi message center.
                                blank                           Leave blank if you
do not want to audit the served user's MWI or
                                                                                if the user is
not a served user of either an fp-mwi or qsig-mwi message center.
                         -->
                        <xs:element name="mwiServedUserType"
maxOccurs="1" minOccurs="0" >
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                        <xs:enumeration value="fp-mwi"/>
                                        <xs:enumeration value="qsig-mwi"/>
                                        <xs:enumeration value="sip-adjunct"/>
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <!-- The AUDIX associated with the station.
                                Must contain a user-defined adjunct name that
was previously administered.
                        -->
                                                <xs:element name="audixName"
type="xs:string" maxOccurs="1" minOccurs="0" />

                        <!--
```

	Automatic Moves allows a DCP telephone to be unplugged from one location and moved to a new location without additional Communication Manager administration. Communication Manager automatically associates the extension to the new port. **********CAUTION********** When a DCP telephone is unplugged and moved to another physical location, the Emergency Location Extension field must be changed for that extension or the USA Automatic Location Identification data base must be manually updated. If the Emergency Location Extension field is not changed or if the USA Automatic Location Identification data base is not updated, the DID number sent to the Public Safety Network could send emergency response personnel to the wrong location.
Valid entries	Usage
always	Enter always and the DCP telephone can be moved anytime without additional administration by unplugging from one location and plugging into a new location.
once	Enter once and the DCP telephone can be unplugged and plugged into a new location once. After a move, the field is set to done the next time that routine maintenance runs on the DCP telephone. Use once when moving a large number of DCP telephones so each extension is

```
removed from the move list. Use once to prevent automatic
                                                                    maintenance
replacement.
                                        no                          Enter no to require
administration in order to move the DCP telephone.
                                        done                    Done is a display-only
value. Communication Manager sets the field to
                                                                    done after the
telephone is moved and routine maintenance runs on the
                                                                    DCP telephone.
                                        error                   Error is a display-only value.
Communication Manager sets the field to
                                                                    error, after routine
maintenance runs on the DCP telephone, when a
                                                                    non-serialized
telephone is set as a movable telephone.
                                -->
                                <xs:element name="automaticMoves" maxOccurs="1"
minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                <xs:enumeration value="always"/>
                                                <xs:enumeration value="no"/>
                                                <xs:enumeration value="once"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>



                                <!--
                                        Tells Communication Manager how to handle
emergency calls from the IP
                                        telephone.

        **********CAUTION**********

                                An Avaya IP endpoint can dial
```

U.S.). It only reaches Answering Point area Please be advised that an local emergency not have local emergency numbers for locations. Avaya Inc. is from misplaced Your use of this product agree to use an from remote representative if you have telephones. Available only if station.	emergency calls (for example, 911 calls in the the local emergency service in the Public Safety where the telephone system has local trunks. Avaya IP endpoint cannot dial to and connect with service when dialing from remote locations that do trunks. Do not use an Avaya IP endpoint to dial emergency services when dialing from remote not responsible or liable for any damages resulting emergency calls made from an Avaya endpoint. indicates that you have read this advisory and alternative telephone to dial all emergency calls locations. Please contact your Avaya questions about emergency calls from IP the station is an IP Softphone or a remote office

Valid entries	Usage
as-on-local	Type as-on-local to achieve the following results: If the administrator chooses to leave the Emergency Location Extension fields (that correspond to this station's IP address) on the IP Address Mapping screen blank, the value as-on-local sends

the extension entered in the Emergency Location Extension field in the Station screen to the Public Safety Answering Point (PSAP). If the administrator populates the IP Address Mapping screen with emergency numbers, the value as-on-local functions as follows:

- If the Emergency Location Extension field in the Station screen is the same as the Emergency Location Extension field in the IP Address Mapping screen, the value as-on-local sends the extension to the Public Safety Answering Point (PSAP).
- If the Emergency Location Extension field in the Station screen is different from the Emergency Location Extension field in the IP Address Mapping screen, the value as-on-local sends the extension in the IP Address Mapping screen to the Public Safety Answering Point (PSAP).

block Enter block to prevent the completion of emergency calls. Use this entry for users who move around but always have a circuit-switched telephone nearby, and for users who are farther away from the Avaya S8XXX Server than an adjacent area code served by the same 911 Tandem office. When users attempt to dial an emergency call from an IP Telephone and the

call is blocked, they can dial 911 from a nearby circuit-switched telephone instead.

cesid Enter cesid to allow Communication Manager to send the CESID information supplied by the IP Softphone to the PSAP. The end user enters the emergency information into the IP Softphone. Use this entry for IP Softphones with road warrior service that are near enough to the Avaya S8XXX Server that an emergency call routed over the it's trunk reaches the PSAP that covers the server or switch. If the server uses ISDN trunks for emergency calls, the digit string is the telephone number, provided that the number is a local direct-dial number with the local area code, at the physical location of the IP Softphone. If the server uses CAMA trunks for emergency calls, the end user enters a specific digit string for each IP Softphone location, based on advice from the local emergency response personnel.

option Enter option to allow the user to select the option (extension, block, or cesid) that the user selected during registration and the IP Softphone reported. Use this entry for extensions that can be swapped back and forth between IP Softphones and a telephone with a fixed location. The

```
user chooses between block and cesid on the softphone. A DCP or
                                                                                                IP
telephone in the office automatically selects extension.

                                -->
                                <xs:element name="remoteSoftphoneEmergencyCalls"
maxOccurs="1" minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                <xs:enumeration value="as-on-local"/>
                                                <xs:enumeration value="block"/>
                                                <xs:enumeration value="cesid"/>
                                                <xs:enumeration value="option"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>



                                <!--
                                        This field allows the system to properly identify the
location of a
                                        caller who dials a 911 emergency call from this
station. An entry in
                                        this field must be of an extension type included in
the dial plan, but
                                        does not have to be an extension on the local
system. It can be a UDP
                                        extension. The entry defaults to blank. A blank
entry typically would
                                        be used for an IP softphone dialing in through
PPP from somewhere
                                        outside your network. If you populate the IP
Address Mapping screen
                                        with emergency numbers, the feature functions as
follows: If the
                                        Emergency Location Extension field in the Station
screen is the same
```

```
                                        as the Emergency Location Extension field in the
IP Address Mapping
                                        screen, the feature sends the extension to the
Public Safety Answering
                                        Point (PSAP). If the Emergency Location
Extension field in the Station
                                        screen is different from the Emergency Location
Extension field in the
                                        IP Address Mapping screen, the feature sends the
extension in the IP
                                        Address Mapping screen to the Public Safety
Answering Point (PSAP).
                                -->
                                <xs:element name="emergencyLocationExt"
maxOccurs="1" minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[0-9]+([\.\-][0-
9]+)*"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>

                                <!--
                                        A softphone can register no matter what
emergency call handling settings
                                        the user has entered into the softphone. If a
softphone dials 911, the
                                        administered Emergency Location Extension is
used. The softphone's
                                        user-entered settings are ignored. If an IP
telephone dials 911, the
                                        administered Emergency Location Extension is
used. If a call center
                                        agent dials 911, the physical station extension is
displayed,
                                        overriding the administered LoginID for ISDN
Display . Does not apply
```

```
                                        to SCCAN wireless telephones, or to extensions
administered as type
                                        h.323.
                                -->
                                <xs:element name="alwaysUse" type="xs:boolean"
maxOccurs="1" minOccurs="0" />


                                <!-- Activates or deactivates Precedence Call Waiting for
this station -->
                                <xs:element name="precedenceCallWaiting"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <!--
                                        Enables or disables automatic selection of any idle
appearance for
                                        transferred or conferenced calls. Communication
Manager first attempts
                                        to find an idle appearance that has the same
extension number as the
                                        call being transferred or conferenced has. If that
attempt fails,
                                        Communication Manager selects the first idle
appearance.
                                -->
                                                        <xs:element
name="autoSelectAnyIdleAppearance" type="xs:boolean" maxOccurs="1"
minOccurs="0" />


                                <!--
                                        Allows or denies users in the telephone's
Coverage Path to retrieve
                                        Leave Word Calling (LWC) messages for this
telephone. Applies only if
                                        the telephone is enabled for LWC Reception.
                                -->


                                <xs:element name="coverageMsgRetrieval"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
```

```
<!--
In EAS environments, the auto answer setting for the Agent LoginID can
override a station's setting when an agent logs in.
Valid Entry Usage
all All ACD and non-ACD calls terminated to an idle station cut through immediately.
Does not allow automatic hands-free answer for intercom calls. With non-ACD calls,
the set is also rung while the call is cut through. The ring can be prevented by activating
the ringer-off feature button when the Allow Ringer-off with Auto-Answer is enabled for the system.
acd Only ACD split /skill calls and direct agent calls to auto answer. Non-ACD calls terminated to a station ring audibly.
For analog stations, the station is off-hook and idle, only the ACD split/skill calls and direct agent calls
auto answer; non-ACD calls receive busy treatment. If the station is active on an ACD call and
a non-ACD call arrives, the Agent receives call-waiting tone.
none All calls terminated to this station receive an audible ringing treatment.
icom Allows a telephone user to answer an intercom call from the same intercom group without pressing the intercom button.
-->
<xs:element name="autoAnswer" maxOccurs="1" minOccurs="0" >
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="acd"/>
<xs:enumeration value="all"/>
<xs:enumeration value="icom"/>
<xs:enumeration value="none"/>
</xs:restriction>
```

```
                    </xs:simpleType>
                </xs:element>

                <!--
                    Enables or disables data restriction that is used to
prevent tones, such as call-waiting tones, from interrupting data calls.
                    Data restriction provides permanent protection
and cannot be changed by the telephone user. Cannot be assigned if Auto Answer
                    is administered as all or acd. If enabled, whisper
page to this station is denied.
                 -->
                <xs:element name="dataRestriction" type="xs:boolean"
maxOccurs="1" minOccurs="0" />

                <!--
                    Indicates which call appearance is selected when
the user lifts the handset and there is an incoming call.
                    Valid Entry                    Usage
                    true                                The
user connects to an idle call appearance instead of the ringing call.
                    false                               The
Alerting Appearance Preference is set and the user connects to the ringing call
appearance.
                 -->
                <xs:element name="idleAppearancePreference"
type="xs:boolean" maxOccurs="1" minOccurs="0" />

                <!--
                    enable/disable call waiting for this station
                -->
                <xs:element name="callWaitingIndication"
type="xs:boolean" maxOccurs="1" minOccurs="0" />

                <!--
                     Attendant call waiting allows attendant-originated
or attendant-extended calls to a busy
                    single-line telephone to wait and sends distinctive
```

call-waiting tone to the single-line user.

Enable/disable attendant call waiting

-->

<xs:element name="attCallWaitingIndication" type="xs:boolean" maxOccurs="1" minOccurs="0" />

<!--

Enter true so the telephone can receive the 3 different types of ringing patterns which identify the type of incoming calls.

Distinctive ringing might not work properly for off-premises telephones. -->

<xs:element name="distinctiveAudibleAlert" type="xs:boolean" maxOccurs="1" minOccurs="0" />

<!--

Valid Entries Usage

true Restricts the last idle call appearance used for incoming priority calls and outgoing call originations only.

false Last idle call appearance is used for incoming priority calls and outgoing call originations.

-->

<xs:element name="restrictLastAppearance" type="xs:boolean" maxOccurs="1" minOccurs="0" />

<!--

Valid entries Usage

true Analog disconnect signal is sent automatically to the port after a call terminates. Analog devices (such as answering machines and speakerphones) use this signal to turn the devices off after a call terminates.

false Hunt group agents are alerted to incoming calls. In a hunt group environment, the disconnect signal blocks the reception of zip tone and incoming call notification by an auto-answer station when a call

```
                                                                    is
queued for the station.
                         -->
                        <xs:element name="adjunctSupervision"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                        <!--
                                    Send Calling Number.
                                    Valid Entries           Usage
                                    y                                   All
outgoing calls from the station will deliver the Calling Party Number

      (CPN) information as "Presentation Allowed."
                                    n                                   No
CPN information is sent for the call
                                    r
      Outgoing non-DCS network calls from the station will deliver the Calling
                                                                        Party
Number information as "Presentation Restricted."
                         -->
                        <xs:element name="perStationCpnSendCallingNumber"
maxOccurs="1" minOccurs="0" >
                              <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                    <xs:enumeration value="r"/>
                                    <xs:enumeration value="n"/>
                                    <xs:enumeration value="y"/>
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>


                        <!--
                              Appears on the Station screen for analog
telephones, only if the Without Flash field in the
                              ANALOG BUSY AUTO CALLBACK section of the
Feature-Related System Parameters
                              screen is set to true. The Busy Auto Callback
```

```
without Flash field then defaults to true for all analog
                                        telephones that allow Analog Automatic Callback.
                                        Set true to provide automatic callback for a calling
analog station without flashing the hook.
                                 -->
                                <xs:element name="busyAutoCallbackWithoutFlash"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <!-- Provides audible message waiting. -->
                                <xs:element name="audibleMessageWaiting"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <!-- Provides extended local calls /
imsFeatureSequencing
                                Extended Local Calls (ELC) /imsFeatureSequencing
allows DCP and H.323 stations to use SIP sequenced applications.  The feature works
by routing calls
                                involving those stations over SIP IMS trunks.  In other
words, CM is applying the half-call model to those stations.
                                That also has the side effect that features which work
differently under the half-call model than under the usual (full-call) model
                                also work differently for ELC stations.
                                The Extended Local Calls feature is administrable per
station.  We're allowing stations that always use SIP IMS trunks to coexist on
                                the same server with stations that don’t always use SIP
IMS trunks.  In other words, ELC is changing a previous marketing rule that
                                the full-call model (CM-ES) and the half-call model (CM-
FS) functions can't co-exist on the same server.  As noted above, that also
                                has the side effect that features which work differently
under the half-call model than under the full-call model now also can work
                                differently for two different SIP stations on the same CM
server.
                                 -->
                                <xs:element name="extendedLocalCalls"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <xs:element name="imsFeatureSequencing"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
```

```
<!--
Only administrable if Hospitality is enabled on the System Parameters
Customer-Options (Optional Features) screen. This field affects the
telephone display on calls that originated from a station with Client
Room Class of Service. Note: For stations with an audix station
type, AUDIX Voice Power ports, or ports for any other type of
messaging that needs display information, Display Client Redirection
must be enabled.
Set true to redirect information for a call originating from a Client Room and terminating to this station displays.
-->
<xs:element name="displayClientRedirection" type="xs:boolean" maxOccurs="1" minOccurs="0" />


<!--
Valid Entries Usage
true Indicates that a station's line selection is not to be moved from the currently selected line button
to a different, non-alerting line button. If you enter true, the line selection on an on-hook station only moves from the last
used line button to a line button with an audibly alerting call. If there are no alerting calls, the line selection
remains on the button last used for a call.
false The line selection on an on-hook station with no alerting calls can be moved to a different line button, which might be serving a different
extension.
-->
<xs:element name="selectLastUsedAppearance" type="xs:boolean" maxOccurs="1" minOccurs="0" />
```

```
                                <!-- Whether an unanswered forwarded call is provided
coverage treatment. -->
                                <xs:element name="coverageAfterForwarding"
type="xs:string" maxOccurs="1" minOccurs="0" />


                                <!-- Allow/disallow direct audio connections between IP
endpoints. -->
                                <xs:element name="directIpIpAudioConnections"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <!-- Allows IP endpoints to be connected through the
server's IP circuit pack. -->
                                <xs:element name="ipAudioHairpinning"
type="xs:boolean" maxOccurs="1" minOccurs="0" />



                                <xs:element name="primeAppearancePreference"
type="xs:string" maxOccurs="1" minOccurs="0" />


                                <!-- Elements with complex data type. Please refer the
appropriate elements for more details. -->
                                <xs:element name="stationSiteData"
type="csm:xmlStationSiteData" maxOccurs="1" minOccurs="0" />
                                <xs:element name="abbrList"
type="csm:xmlStationAbbreviatedDialingData" maxOccurs="unbounded"
minOccurs="0" />
                                <xs:element name="buttons" type="csm:xmlButtonData"
maxOccurs="24" minOccurs="0" />
                                <xs:element name="featureButtons"
type="csm:xmlButtonData" maxOccurs="24" minOccurs="0" />
                                <xs:element name="expansionModuleButtons"
type="csm:xmlButtonData" maxOccurs="72" minOccurs="0" />
                                <xs:element name="softKeys" type="csm:xmlButtonData"
maxOccurs="15" minOccurs="0" />
                                <xs:element name="displayButtons"
type="csm:xmlButtonData" maxOccurs="unbounded" minOccurs="0" />
                                <xs:element name="stationDataModule"
type="csm:xmlStationDataModule" maxOccurs="1" minOccurs="0" />
```

```
                                <xs:element name="hotLineData"
type="csm:xmlStationHotLineData" maxOccurs="1" minOccurs="0" />
                                <xs:element  name="nativeName"
type="csm:xmlNativeNameData" maxOccurs="1" minOccurs="0"/>


                                <!-- Number of button modules 0-3-->
                                <xs:element name="buttonModules" maxOccurs="1"
minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:int">
                                                        <xs:minInclusive value="0" />
                                                        <xs:maxInclusive value="3" />
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>



                                <xs:element name="unconditionalInternalDest"
maxOccurs="1" minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[*][0-9]{1,16}[#]|
[0123456789]{1,17}[#]|[*][0-9]{1,17}|[0-9]{1,18}|[*][#]|"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>



                                <xs:element name="unconditionalInternalActive"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <xs:element name="unconditionalExternalDest"
maxOccurs="1" minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[*][0-9]{1,16}[#]|
```

```
[0123456789]{1,17}[#]|[*][0-9]{1,17}|[0-9]{1,18}|[*][#]|"/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>



                                <xs:element name="unconditionalExternalActive"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <xs:element name="busyInternalDest" maxOccurs="1"
minOccurs="0" >
                                    <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                            <xs:pattern value="[*][0-9]{1,16}[#]|
[0123456789]{1,17}[#]|[*][0-9]{1,17}|[0-9]{1,18}|[*][#]|"/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>


                                <xs:element name="busyInternalActive"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <xs:element name="busyExternalDest" maxOccurs="1"
minOccurs="0" >
                                    <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                            <xs:pattern value="[*][0-9]{1,16}[#]|
[0123456789]{1,17}[#]|[*][0-9]{1,17}|[0-9]{1,18}|[*][#]|"/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>


                                <xs:element name="busyExternalActive"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <xs:element name="noReplyInternalDest"
```

```
maxOccurs="1" minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[*][0-9]{1,16}[#]|
[0123456789]{1,17}[#]|[*][0-9]{1,17}|[0-9]{1,18}|[*][#]|"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>


                                <xs:element name="noReplyInternalActive"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <xs:element name="noReplyExternalDest"
maxOccurs="1" minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[*][0-9]{1,16}[#]|
[0123456789]{1,17}[#]|[*][0-9]{1,17}|[0-9]{1,18}|[*][#]|"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>


                                <xs:element name="noReplyExternalActive"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <xs:element name="sacCfOverride" maxOccurs="1"
minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                <xs:enumeration value="a"/>
                                                <xs:enumeration value="n"/>
                                                <xs:enumeration value="y"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>
```

```
<xs:element name="lossGroup" maxOccurs="1"
minOccurs="0" >
    <xs:simpleType>
        <xs:restriction base="xs:int">
            <xs:minInclusive value="1" />
            <xs:maxInclusive value="19" />
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<xs:element name="timeOfDayLockTable"
maxOccurs="1" minOccurs="0">

    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="[1-5]|[0-9]{0}"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<xs:element name="emuLoginAllowed"
type="xs:boolean" maxOccurs="1" minOccurs="0" />

<xs:element name="ec500State" maxOccurs="1"
minOccurs="0" >
    <xs:simpleType>
        <xs:restriction base="xs:string">
        <xs:enumeration value="enabled"/>
        <xs:enumeration value="disabled"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<!-- true/false to enable/disable Mute on Off Hook in
```

```
Shared Control Mode feature. -->
                                <xs:element name="muteOnOffHookInSCMode"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                                <xs:element name="type3pccEnabled" maxOccurs="1"
minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                <xs:enumeration value="None"/>
                                                <xs:enumeration value="Avaya"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>


                                <!-- true/false for calculate route pattern. -->
                                <xs:element name="calculateRoutePattern"
type="xs:boolean" maxOccurs="1" minOccurs="0"/>


                                <xs:element name="sipTrunk" maxOccurs="1"
minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="aar|ars|tg[1-9]|
tg[1-9][0-9]|tg[1-9]([0-9]){2}|tg[1]([0-9]){3}|tg2000|rp[1-9]|rp[1-9][0-9]|rp[1-9]([0-9]){2}|
rp[1]([0-9]){3}|rp2000"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>


                                <xs:element name="enableReachStaDomainControl"
maxOccurs="1" minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                <xs:enumeration value="system"/>
                                                <xs:enumeration value="s"/>
                                                <xs:enumeration value="y"/>
```

```
                                        <xs:enumeration value="n"/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>

                                <xs:element name="multimediaEarlyAnswer"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <xs:element name="bridgedApprOrigRestr"
type="xs:boolean" maxOccurs="1" minOccurs="0" />

                                <xs:element name="callApprDispFormat" maxOccurs="1"
minOccurs="0" >
                                    <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                        <xs:enumeration value="inter-location"/>
                                        <xs:enumeration value="intra-location"/>
                                        <xs:enumeration value="disp-param-
default"/>
                                    </xs:restriction>

                                    </xs:simpleType>
                                </xs:element>

                                <!-- Enter a Group ID between 0-999, or blank -->
                                <xs:element name="ipPhoneGroupId" maxOccurs="1"
minOccurs="0">
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                            <xs:pattern value="[0-9]|[0-9][0-9]|
[0-9][0-9][0-9]|[0-9]{0}"/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>

                                <xs:element name="xoipEndPointType" maxOccurs="1"
minOccurs="0" >
```

```
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                <xs:enumeration value="auto"/>
                                                <xs:enumeration value="fax"/>
                                                <xs:enumeration value="modem"/>
                                                <xs:enumeration value="tty"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>

                                <xs:element name="xid" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
                                <xs:element name="stepClearing" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
                                <xs:element name="fixedTei" type="xs:boolean"
maxOccurs="1" minOccurs="0" />

                                <xs:element name="tei" maxOccurs="1" minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[0-6][0-3]"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>

                                <xs:element name="countryProtocol" maxOccurs="1"
minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                <xs:enumeration value="1"/>
                                                <xs:enumeration value="2"/>
                                                <xs:enumeration value="3"/>
                                                <xs:enumeration value="6"/>
                                                <xs:enumeration value="etsi"/>
                                                </xs:restriction>
```

```
                                        </xs:simpleType>
                                    </xs:element>


                                    <xs:element name="endptInit" type="xs:boolean"
maxOccurs="1" minOccurs="0" />


                                    <xs:element name="spid" maxOccurs="1" minOccurs="0"
>
                                        <xs:simpleType>
                                            <xs:restriction base="xs:string">
                                                <xs:pattern value="[0-9]{1,10}"/>
                                            </xs:restriction>
                                        </xs:simpleType>
                                    </xs:element>



                                    <xs:element name="endptId" maxOccurs="1"
minOccurs="0" > <!-- 00 to 62 -->
                                        <xs:simpleType>
                                            <xs:restriction base="xs:string">
                                                <xs:pattern value="[0-6][0-2]"/>
                                            </xs:restriction>
                                        </xs:simpleType>
                                    </xs:element>


                                    <xs:element name="isMCTSignalling" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
                                    <xs:element name="isShortCallingPartyDisplay"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                    <xs:element name="passageWay" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
                                    <xs:element name="dtmfOverIp" maxOccurs="1"
minOccurs="0" >
                                        <xs:simpleType>
                                            <xs:restriction base="xs:string">
                                            <xs:enumeration value="in-band"/>
```

```
                                        <xs:enumeration value="in-band-g711"/>
                                        <xs:enumeration value="out-of-band"/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>
                                <xs:element name="location" maxOccurs="1"
minOccurs="0">
                                    <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                            <xs:pattern value="[1-9]{0}|[1-9]|[1-
9][0-9]|[1-9]([0-9]){2}|[1]([0-9]){3}|2000"/>
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>
                                <!-- true/false for display Caller Id. -->
                                <xs:element name="displayCallerId" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
                                <!-- true/false for caller id message waiting indication. -->
                                <xs:element name="callerIdMsgWaitingIndication"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                                <!-- true/false for recall rotary digit. -->
                                <xs:element name="recallRotaryDigit" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
                            </xs:sequence>
                        </xs:extension>
    </xs:complexContent>
</xs:complexType>


<xs:complexType name="xmlStationSiteData">
        <xs:sequence>
                <xs:element name="room" maxOccurs="1" minOccurs="0" >
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                        <xs:maxLength value="10"/>
```

```
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>

            <xs:element name="jack" maxOccurs="1" minOccurs="0" >
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:maxLength value="5"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="cable" maxOccurs="1" minOccurs="0" >
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:maxLength value="5"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>

            <xs:element name="floor" type="xs:string" maxOccurs="1"
minOccurs="0" />
            <xs:element name="building" type="xs:string" maxOccurs="1"
minOccurs="0" />
            <xs:element name="headset" type="xs:boolean" maxOccurs="1"
minOccurs="0" />
            <xs:element name="speaker" type="xs:boolean" maxOccurs="1"
minOccurs="0" />

            <xs:element name="mounting" maxOccurs="1" minOccurs="0" >
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                    <xs:enumeration value="d"/>
                    <xs:enumeration value="w"/>
                    </xs:restriction>
                </xs:simpleType>
```

```
                </xs:element>

                <!-- Enter numeric cord length (0-99) -->
                <xs:element name="cordLength" maxOccurs="1" minOccurs="0" >
                        <xs:simpleType>
                                <xs:restriction base="xs:int">
                                        <xs:minInclusive value="0" />
                                        <xs:maxInclusive value="99" />
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>

                <xs:element name="setColor" type="xs:string" maxOccurs="1"
minOccurs="0" />
        </xs:sequence>
</xs:complexType>


<xs:complexType name="xmlStationAbbreviatedDialingData">
        <xs:sequence>
                <xs:element name="listType" maxOccurs="1" minOccurs="1" >
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                <xs:enumeration value="enhanced"/>
                                <xs:enumeration value="group"/>
                                <xs:enumeration value="personal"/>
                                <xs:enumeration value="system"/>
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>

                <xs:element name="number" type="xs:int" maxOccurs="1"
minOccurs="1" />
                <xs:element name="listId" type="xs:int" maxOccurs="1"
```

```
minOccurs="0" />
        </xs:sequence>
</xs:complexType>


<xs:complexType name="xmlButtonData">
        <xs:sequence>
                <xs:element name="number" type="xs:int" maxOccurs="1"
minOccurs="1" /><!--  *******Must present******  -->
                <xs:element name="type" type="xs:string" maxOccurs="1"
minOccurs="1" /><!--  *******Must present******  -->
                <xs:element name="data1" type="xs:string" maxOccurs="1"
minOccurs="0" />
                <xs:element name="data2" type="xs:string" maxOccurs="1"
minOccurs="0" />
                <xs:element name="data3" type="xs:string" maxOccurs="1"
minOccurs="0" />
                <xs:element name="data4" type="xs:string" maxOccurs="1"
minOccurs="0" />
                <xs:element name="data5" type="xs:string" maxOccurs="1"
minOccurs="0" />
                <xs:element name="data6" type="xs:string" maxOccurs="1"
minOccurs="0" />
        </xs:sequence>
</xs:complexType>
<xs:complexType name="xmlStationDataModule">
        <xs:sequence>
                <xs:element name="dataExtension" maxOccurs="1" minOccurs="1"
><!--  *******Must present******  -->
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                        <xs:pattern value="[0-9]+([.-][0-9]+)*"/>
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>


                <xs:element name="name" maxOccurs="1" minOccurs="0" >
```

```
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                                <xs:maxLength value="29"/>
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="cor" maxOccurs="1" minOccurs="1" ><!--
*******Must present******  -->
                <xs:simpleType>
                        <xs:restriction base="xs:int">
                                <xs:minInclusive value="0" />
                                <xs:maxInclusive value="995" />
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
        <xs:element name="cos" maxOccurs="1" minOccurs="1" ><!--
*******Must present******  -->
                <xs:simpleType>
                        <xs:restriction base="xs:int">
                                <xs:minInclusive value="0" />
                                <xs:maxInclusive value="15" />
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>


        <xs:element name="itc" maxOccurs="1" minOccurs="1" ><!--
*******Must present******  -->
                <xs:simpleType>
                        <xs:restriction base="xs:string">
                        <xs:enumeration value="restricted"/>
                        <xs:enumeration value="unrestricted"/>
                        </xs:restriction>
                </xs:simpleType>
        </xs:element>
```

```
<!-- CM dependant field - 100 or 250 depends on system params -->
<xs:element name="tn" maxOccurs="1" minOccurs="1" ><!--
*******Must present******  -->
<xs:simpleType>
<xs:restriction base="xs:int">
<xs:minInclusive value="1" />
<xs:maxInclusive value="250" />
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:element name="listType" maxOccurs="1" minOccurs="0" >
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="enhanced"/>
<xs:enumeration value="group"/>
<xs:enumeration value="personal"/>
<xs:enumeration value="system"/>
</xs:restriction>
</xs:simpleType>
</xs:element>

<xs:element name="listId" type="xs:int" maxOccurs="1"
minOccurs="0" />

<xs:element name="specialDialingOption" maxOccurs="1"
minOccurs="0" >
<xs:simpleType>
<xs:restriction base="xs:string">
<xs:enumeration value="default"/>
<xs:enumeration value="hot-line"/>
</xs:restriction>
</xs:simpleType>
</xs:element>
<xs:element name="specialDialingAbbrDialCode" maxOccurs="1"
minOccurs="0" >
```

```
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                        <xs:maxLength value="4"/>
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>
        </xs:sequence>
</xs:complexType>
<xs:complexType name="xmlStationHotLineData">
        <xs:sequence>
                <xs:element name="hotLineDestAbbrevList" maxOccurs="1"
minOccurs="0" >
                        <xs:simpleType>
                                <xs:restriction base="xs:int">
                                        <xs:minInclusive value="1" />
                                        <xs:maxInclusive value="3" />
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>
                <xs:element name="hotLineAbbrevDialCode" maxOccurs="1"
minOccurs="0" >
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[0-9]*"/>
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>
        </xs:sequence>
</xs:complexType>


<!-- If displayName,givenName or surname contains characters of multiple scripts then
locale tag should be present.
        If displayName tag is present then it overwrites native name.
        If displayname is not present then combination of givenName and surname
gets copied in native name.
```

```
        Please find below locale for multiscript language
        Language            Locale
        Japanese                                ja, ja-jp
        Simplified Chinese              zh-cn
        Traditional Chinese    zh-tw -->
<xs:complexType name="xmlNativeNameData">
        <xs:sequence>
                <xs:element name="locale" maxOccurs="1" minOccurs="0">
                  <xs:simpleType>
                                <xs:restriction base="xs:string">
                                <xs:enumeration value="ja-jp"/>
                                <xs:enumeration value="ja"/>
                                <xs:enumeration value="zh-cn"/>
                                <xs:enumeration value="zh-tw"/>

                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>
                <xs:element name="name" maxOccurs="1" minOccurs="0" >
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                        <xs:maxLength value="27"/>
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>
        </xs:sequence>
</xs:complexType>

</xs:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Messaging

Note: System Manager 7.0 introduces Updated Messaging XSD.

Note: System Manager 7.0.1 introduces new fields in Messaging XSD: 'TimeZone', 'frwdDestAddress' and 'deleteAfterForward'

Note: System Manager 7.1 introduces new fields in Messaging XSD: 'tuiMessageOrderUnread', 'playUnreadUrgentFirst', 'arrangeBySenderUnread', 'tuiMessageOrderRead', 'playReadUrgentFirst', 'arrangeBySenderRead', 'tuiMessageOrderSaved', 'playSavedUrgentFirst', 'arrangeBySenderSaved'

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:one="http://xml.avaya.com/schema/import"
elementFormDefault="qualified"
        targetNamespace="http://xml.avaya.com/schema/import_csm_mm"
xmlns:csm="http://xml.avaya.com/schema/import_csm_mm">


        <xs:import namespace="http://xml.avaya.com/schema/import"
                schemaLocation="userimport.xsd" />
        <!--Changes in xsd file need to generate jaxb src using this xsd-->
        <xs:complexType name="xmlMessagingProfile">
                <xs:complexContent>
                        <xs:extension base="one:xmlCommProfileType">
                                <xs:sequence>
                                        <!--
                                                Specifies the messaging system of the
subscriber you want to add.
                                                Name as it appears under
'Applications/Application
                                                Management/Entities
                                        -->
                                        <xs:element name="messagingName"
type="xs:string"
                                                maxOccurs="1" minOccurs="1" />


                                        <xs:element name="msgprefHandleId"
type="xs:string"
```

```
                                        maxOccurs="1" minOccurs="0" />

                                <xs:element name="useExisting"
type="xs:boolean"
                                        maxOccurs="1" minOccurs="0" />

                                <!-- Specifies the messaging template of a
subscriber.  -->
                                <xs:element name="messagingTemplate"
type="xs:string"
                                        maxOccurs="1" minOccurs="0" />

                                <xs:element name="mailboxNumber"
maxOccurs="1"
                                        minOccurs="1">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[0-9]
{1,50}" />
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>

                                <!--
                                        Specifies the default password the
subscriber must use to log in
                                        to his or her mailbox. The password can be
from one digit in
                                        length to a maximum of 15 digits.
                                -->
                                <xs:element name="password" maxOccurs="1"
minOccurs="1">
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[0-9]
{0,15}" />
```

```
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>

                                <xs:element name="deleteOnUnassign"
type="xs:boolean"
                                    maxOccurs="1" minOccurs="0" />

                                <!-- follows overrriding  subscriber  data  -->

                                <!--
                                    The class of service for this subscriber.
The COS controls
                                    subscriber access to many features and
provides general settings,
                                    such as mailbox size.
                                -->
                                <xs:element name="cos" maxOccurs="1"
minOccurs="0">
                                    <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                            <xs:pattern
                                                value="[0-9]|[0-9]{2}|
[0-4][0-9]{2}|[5][0-4][0-9]|[5][5][0-1]" />
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>

                                <!--
                                    Specifies the name that appears before the
machine name and domain
                                    in the subscriber's e-mail address. The
machine name and domain
                                    are automatically added to the handle you
enter when the
                                    subscriber sends or receives an e-mail.
```

```
                                        -->
                                        <xs:element name="emailHandle" maxOccurs="1"
minOccurs="0">
                                                <xs:simpleType>
                                                        <xs:restriction base="xs:string">
                                                                <xs:pattern value="[a-zA-
Z0-9_@\w\.\-]*" />
                                                        </xs:restriction>
                                                </xs:simpleType>
                                        </xs:element>

                                        <!--
                                                Specifies the display name of the
subscriber in address book
                                                listings, such as those for e-mail client
applications. The name
                                                you enter can be 1 to 64 characters in
length.
                                        -->
                                        <xs:element name="commonName"
type="xs:string"
                                                maxOccurs="1" minOccurs="0" />

                                        <xs:element name="mmSpecific"
type="csm:xmlMMSpecific"
                                                maxOccurs="1" minOccurs="0" />
                                        <xs:element name="cmmSpecific"
type="csm:xmlCMMSpecific"
                                                maxOccurs="1" minOccurs="0" />
                                        <xs:element name="aamSpecific"
type="csm:xmlAAMSpecific"
                                                maxOccurs="1" minOccurs="0" />
                                </xs:sequence>
                        </xs:extension>
                </xs:complexContent>
        </xs:complexType>
```

```
<xs:complexType name="xmlMMSpecific">
        <xs:sequence>

                <!-- Specifies the default community ID for the subscriber.
Community IDs
                        are used to control message sending and receiving
among groups of subscribers.
                        The default value is 1. -->
                <xs:element name="communityID" maxOccurs="1"
minOccurs="0">
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                        <xs:pattern value="[0-9]|[0-1][0-5]" />
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>

                <!--
                        Specifies a unique address in the voice mail network. The
numeric
                        address can be from 1 to 50 digits and can contain the
Mailbox
                        Number.
                -->
                <xs:element name="numericAddress" maxOccurs="1"
minOccurs="0">
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                        <xs:pattern value="([0-9])*" />
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>
```

```
<!-- The primary telephone extension of the subscriber. -->
<xs:element name="pbxExtension" maxOccurs="1"
minOccurs="0">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="([+0-9])*" />
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<!--
    The telephone number of the subscriber as displayed in
address book
    listings and client applications. The entry can be a
maximum of 50
    characters in length and can contain any combination of
digits
    (0-9), period (.), hyphen (-), plus sign (+), and left and
right
    parentheses ([) and (]).
-->
<xs:element name="telephoneNumber" maxOccurs="1"
    minOccurs="0"> <!-- MM field -->
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="([-+\.()0-9])*" />
        </xs:restriction>
    </xs:simpleType>
</xs:element>

<!--
    If the subscriber name is entered in multi-byte character
format,
    then this field specifies the ASCII translation of the
subscriber
```

```
                    name.
                -->
                <xs:element name="asciiVersionOfName" type="xs:string"
                    maxOccurs="1" minOccurs="0" />

                <xs:element name="expirePassword" type="csm:xmlyesNoType"
                    maxOccurs="1" minOccurs="0" />

                <!--
                    Specifies whether you want your mailbox to be locked. A
subscriber
                    mailbox can become locked after two unsuccessful login
attempts. You
                    can choose one of the following: - no: to unlock your
mailbox - yes:
                    to lock your mailbox and prevent access to it
                -->
                <xs:element name="mailBoxLocked" type="csm:xmlyesNoType"
                    maxOccurs="1" minOccurs="0" />

                <!--
                    Specifies the mailbox number or transfer dial string of the
                    subscriber's personal operator or assistant. This field also
                    indicates the transfer target when a caller to this
subscriber
                    presses 0 while listening to the subscriber's greeting.
                -->
                <xs:element name="personalOperatorMailbox" maxOccurs="1"
                    minOccurs="0">
                    <xs:simpleType>
                        <xs:restriction base="xs:string">
                            <xs:pattern value="[0-9]+([*#,][0-9]+)*" />
                        </xs:restriction>
                    </xs:simpleType>
                </xs:element>
```

```
<!--
        Specifies when to route calls to the backup operator
mailbox. The
        default value for this field is Always Active.
-->
<xs:element name="personalOperatorSchedule" type="xs:string"
        maxOccurs="1" minOccurs="0" />

<xs:element name="tuiMessageOrderAdmin" maxOccurs="1"
        minOccurs="0">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs:enumeration value="urgent first then
newest" />
                        <xs:enumeration value="oldest messages
first" />
                        <xs:enumeration value="newest messages
first" />
                        <xs:enumeration value="urgent first then
oldest" />
                </xs:restriction>
        </xs:simpleType>
</xs:element>

<xs:element name="tuiMessageOrderDeleted" maxOccurs="1"
        minOccurs="0">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs:enumeration value="urgent first then
newest" />
                        <xs:enumeration value="oldest messages
first" />
                        <xs:enumeration value="newest messages
first" />
                        <xs:enumeration value="urgent first then
```

```
oldest" />
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>

                                <xs:element name="tuiMessageOrderNew" maxOccurs="1"
                                    minOccurs="0">
                                    <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                            <xs:enumeration value="urgent first then
newest" />
                                            <xs:enumeration value="oldest messages
first" />
                                            <xs:enumeration value="newest messages
first" />
                                            <xs:enumeration value="urgent first then
oldest" />
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>

                                <xs:element name="tuiMessageOrderSaved" maxOccurs="1"
                                    minOccurs="0"> <!-- MM field -->
                                    <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                            <xs:enumeration value="urgent first then
newest" />
                                            <xs:enumeration value="oldest messages
first" />
                                            <xs:enumeration value="newest messages
first" />
                                            <xs:enumeration value="urgent first then
oldest" />
                                        </xs:restriction>
                                    </xs:simpleType>
                                </xs:element>
```

```
<!--
        Specifies the intercom paging settings for a subscriber. You can
        choose one of the following: - paging is off: to disable intercom
        paging for this subscriber. - paging is manual: if the subscriber
        can modify, with Subscriber Options or the TUI, the setting that
        allows callers to page the subscriber. - paging is automatic: if the
        TUI automatically allows callers to page the subscriber.
-->
<xs:element name="intercomPaging" maxOccurs="1" minOccurs="0"> <!-- MM field -->
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs:enumeration value="paging is off" />
                        <xs:enumeration value="paging is manual" />
                        <xs:enumeration value="paging is automatic" />
                </xs:restriction>
        </xs:simpleType>
</xs:element>


<!--
        Specifies whether a subscriber can receive messages, e-mail messages
        and call-answer messages from other subscribers. You can choose one
        of the following: - yes: to allow the subscriber to create, forward,
        and receive messages. - no: to prevent the subscriber from receiving
        call-answer messages and to hide the subscriber from the
```

```
telephone
                    user interface (TUI). The subscriber cannot use the TUI to access
                    the mailbox, and other TUI users cannot address messages to the
                    subscriber.
            -->
            <xs:element name="voiceMailEnabled" type="csm:xmlTrueFalseType"
                    maxOccurs="1" minOccurs="0" />


            <!--
                    Specifies additional, useful information about a subscriber. Entries
                    in this field are for convenience and are not used by the messaging
                    system.
            -->
            <xs:element name="miscellaneous1" type="csm:xmlLength51Type"
                    maxOccurs="1" minOccurs="0" />


            <!--
                    Specifies additional, useful information about a subscriber. Entries
                    in this field are for convenience and are not used by the messaging
                    system.
            -->
            <xs:element name="miscellaneous2" type="csm:xmlLength51Type"
                    maxOccurs="1" minOccurs="0" />


            <!--
                    Specifies additional, useful information about a subscriber. Entries
                    in this field are for convenience and are not used by the
```

```
messaging
                                        system.
                                -->
                                <xs:element name="miscellaneous3"
type="csm:xmlLength51Type"
                                        maxOccurs="1" minOccurs="0" />


                                <!--
                                        Specifies additional, useful information about a
subscriber. Entries
                                        in this field are for convenience and are not used by the
messaging
                                        system.
                                -->
                                <xs:element name="miscellaneous4"
type="csm:xmlLength51Type"
                                        maxOccurs="1" minOccurs="0" />


                                <!-- Specifies one or more alternate number to reach a
subscriber. You can
                                        use secondary extensions to specify a telephone number
for direct reception
                                        of faxes, to allow callers to use an existing Caller
Application, or to identify
                                        each line appearance on the subscriber's telephone set if
they have different
                                        telephone numbers. -->
                                <xs:element name="secondaryExtension" maxOccurs="1"
                                        minOccurs="0"> <!-- MM/CMM field -->
                                        <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                        <xs:pattern value="[0-9,]{0,50}" />
                                                </xs:restriction>
                                        </xs:simpleType>
                                </xs:element>
                        </xs:sequence>
```

```
        </xs:complexType>

        <xs:complexType name="xmlCMMSpecific">
                <xs:sequence>

                        <!--
                                Specifies the number of the switch on which this
subscriber's
                                extension is administered. You can enter "0" through "99",
or leave
                                this field blank. - Leave this field blank if the host switch
number
                                should be used. - Enter a "0" if no message waiting
indicators
                                should be sent for this subscriber. You should enter 0
when the
                                subscriber does not have a phone on any switch in the
network.
                        -->
                        <xs:element name="switchNumber" maxOccurs="1"
minOccurs="0">
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:pattern value="[0-9]|[0-9][0-9]" />
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <!-- Specifies the default community ID for the subscriber.
Community IDs
                                are used to control message sending and receiving
among groups of subscribers.
                                The default value is 1. -->
                        <xs:element name="communityID" maxOccurs="1"
minOccurs="0"> <!-- MM/CMM field -->
                                <xs:simpleType>
```

```
                                        <xs:restriction base="xs:string">
                                                <xs:pattern value="[0-9]|[0-1][0-5]" />
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <!--
                                Specifies the Subscriber Account Code. The Subscriber
Account Code
                                is used to create Call Detail Records on the switch for
calls placed
                                by the voice ports. The value you enter in this field can
contain
                                any combination of digits from 0 to 9. If an account code
is not
                                specified, the system will use the subscriber's mailbox
extension as
                                the account code.
                        -->
                        <xs:element name="accountCode" maxOccurs="1"
minOccurs="0"> <!-- CMM field -->
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:pattern value="([0-9])*" />
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <xs:element name="mwiEnabled" type="csm:xmlyesNoType"
                                maxOccurs="1" minOccurs="0" />

                        <!--
                                Specifies the number to be used as the default
destination for the
                                Transfer Out of Messaging feature. You can enter 3 to 50
digits in
```

```
                    this field depending on the length of the system's
extension, or
                    leave this field blank.
            -->
            <xs:element name="coveringExtension" maxOccurs="1"
                    minOccurs="0"> <!-- CMM field -->
                    <xs:simpleType>
                            <xs:restriction base="xs:string">
                                    <xs:pattern value="[0-9]{0}|[0-9]{3,50}" />
                            </xs:restriction>
                    </xs:simpleType>
            </xs:element>

            <!--
                    Specifies additional, useful information about a
subscriber. Entries
                    in this field are for convenience and are not used by the
messaging
                    system.
            -->
            <xs:element name="miscellaneous1"
type="csm:xmlLength11Type"
                    maxOccurs="1" minOccurs="0" />

            <!--
                    Specifies additional, useful information about a
subscriber. Entries
                    in this field are for convenience and are not used by the
messaging
                    system.
            -->
            <xs:element name="miscellaneous2"
type="csm:xmlLength11Type"
                    maxOccurs="1" minOccurs="0" />
```

```
<!--
        Specifies additional, useful information about a
subscriber. Entries
        in this field are for convenience and are not used by the
messaging
        system.
-->
<xs:element name="miscellaneous3"
type="csm:xmlLength11Type"
        maxOccurs="1" minOccurs="0" />


<!--
        Specifies additional, useful information about a
subscriber. Entries
        in this field are for convenience and are not used by the
messaging
        system.
-->
<xs:element name="miscellaneous4"
type="csm:xmlLength11Type"
        maxOccurs="1" minOccurs="0" />


<!-- Specifies alternate number to reach a subscriber. You can
        use secondary extension to specify a telephone number
for direct reception
        of faxes, to allow callers to use an existing Caller
Application, or to identify
        each line appearance on the subscriber's telephone set if
they have different
        telephone numbers. Valid Entries are blank or 3-50 digits
(0-9), depending on the length of the system's extension. -->
<xs:element name="secondaryExtension" maxOccurs="1"
        minOccurs="0"> <!-- MM/CMM field -->
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs:pattern value="[0-9]{0}|[0-9]{3,50}" />
```

```
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>


                        <!--
                                If the subscriber name is entered in multi-byte character
format,
                                then this field specifies the ASCII translation of the
subscriber
                                name. wi01032730
                        -->
                        <xs:element name="asciiVersionOfName" type="xs:string"
                                maxOccurs="1" minOccurs="0" />
                        <!--
                                Specifies an email address to be used as the Forwarding
Destination Address
                                where messages should be forwarded to when an
incoming message is received.
                                This field can include alphanumeric characters, periods
(.), hyphens (-),
                                underscores (_), pluses (+), colons (:) and should also
include the commercial at (@).
                                Note: we allow maxLength 76 for frwdDestAddress in
order to avoid issues with System Manager data base limitation. -->
                        <xs:element name="frwdDestAddress" maxOccurs="1"
                                minOccurs="0"> <!-- CMM field -->
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:pattern value="[a-zA-Z0-9_\.\+-:@a-
zA-Z0-9_\.\+-:]*" />
                                                <xs:maxLength value="76" />
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>
                        <!--
                                Specifies that an incoming message should be deleted
```

```
once it is forwarded on to the Forwarding Destination Address.
                        -->
                        <xs:element name="deleteAfterForward" type="xs:boolean"
                                maxOccurs="1" minOccurs="0" />
                </xs:sequence>
        </xs:complexType>

        <xs:complexType name="xmlAAMSpecific">
                <xs:sequence>

                        <!-- The Id of Site -->
                        <xs:element name="siteId" maxOccurs="1" minOccurs="0">
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:pattern value="([0-9])*" />
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <!--
                                Specifies a unique address in the voice mail network. The
numeric
                                address can be from 1 to 52 digits and can contain the
Mailbox Number.
                        -->
                        <xs:element name="numericAddress" maxOccurs="1"
minOccurs="0">
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:pattern value="([0-9])*" />
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <!-- The primary telephone extension of the subscriber. It can be
```

```
from 3 to 50 digits. -->
                        <xs:element name="pbxExtension" maxOccurs="1"
minOccurs="0">
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:pattern value="([+0-9])*" />
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <xs:element name="storageDestination"
type="csm:xmlMssExchangeType"
                                maxOccurs="1" minOccurs="0" />

                        <xs:element name="storageDestinationAccountId"
type="xs:string"
                                maxOccurs="1" minOccurs="0" />

                        <xs:element name="storageDestinationServer" type="xs:string"
                                maxOccurs="1" minOccurs="0" />

                        <!--
                                The telephone number of the subscriber as displayed in
address book
                                listings and client applications. The entry can be a
maximum of 32
                                characters in length and can contain any combination of
digits
                                (0-9), period (.), hyphen (-), plus sign (+), and left and
right
                                parentheses ([) and (]).
                        -->
                        <xs:element name="telephoneNumber" maxOccurs="1"
                                minOccurs="0"> <!-- MM field -->
                                <xs:simpleType>
```

```
                    <xs:restriction base="xs:string">
                        <xs:pattern value="([-+\.()0-9])*" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>

            <xs:element name="asciiVersionOfName" type="xs:string"
                maxOccurs="1" minOccurs="0" />

            <xs:element name="pronounceableName" type="xs:string"
                maxOccurs="1" minOccurs="0" />

            <xs:element name="includedInAADirectory" type="xs:boolean"
                        maxOccurs="1" minOccurs="0" />

            <xs:element name="expirePassword" type="csm:xmlyesNoType"
                maxOccurs="1" minOccurs="0" />

            <!--
                Specifies whether you want your mailbox to be locked. A
subscriber
                mailbox can become locked after two unsuccessful login
attempts. You
                can choose one of the following: - no: to unlock your
mailbox - yes:
                to lock your mailbox and prevent access to it
            -->
            <xs:element name="mailBoxLocked" type="csm:xmlyesNoType"
                maxOccurs="1" minOccurs="0" />

            <xs:element name="mustChangePassword" type="xs:boolean"
                        maxOccurs="1" minOccurs="0" />

            <!--
                Specifies subscriber's personal attendant.
```

```
-->
<xs:element name="personalAttendant" maxOccurs="1"
        minOccurs="0"> <!-- MM field -->
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs:pattern value="([-+()0-9])*" />
                </xs:restriction>
        </xs:simpleType>
</xs:element>


<!--
        Specifies the intercom paging settings for a subscriber.
You can
        choose one of the following: - paging is off: to disable
intercom
        paging for this subscriber. - paging is manual: if the
subscriber
        can modify, with Subscriber Options or the TUI, the
setting that
        allows callers to page the subscriber. - paging is
automatic: if the
        TUI automatically allows callers to page the subscriber.
        Note: intercomPaging is no longer used since
SMGR_10.1.0.0. However, we don't remove the related code from
messagingCommProfile.xsd in order to support backward compatibility.
-->
<xs:element name="intercomPaging" maxOccurs="1"
minOccurs="0"> <!-- MM field -->
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs:enumeration value="paging is off" />
                        <xs:enumeration value="paging is manual"
/>
                        <xs:enumeration value="paging is
automatic" />
                </xs:restriction>
```

```
                    </xs:simpleType>
            </xs:element>

            <!--
                    Specifies the order in which the subscriber hears the voice
                    messages. You can choose one of the following: - urgent first then
                    newest: to direct the system to play any messages marked as urgent
                    prior to playing non-urgent messages. Both the urgent and non-urgent
                    messages are played in the reverse order of how they were received.
                    - oldest messages first: to direct the system to play messages in
                    the order they were received. - urgent first then oldest: to direct
                    the system to play any messages marked as urgent prior to playing
                    non-urgent messages. Both the urgent and non-urgent messages are
                    played in the order of how they were received. - newest messages
                    first: to direct the system to play messages in the reverse order of
                    how they were received.
                    Note: tuiMessageOrder field is no longer used since
SMGR 7.1.2. However, we don't remove the related code from
messagingCommProfile.xsd in order to support backward compatibility.
You should use playback order related fields for Unread, Read and
Saved messages instead of tuiMessageOrder:
tuiMessageOrderUnread, playUnreadUrgentFirst, arrangeBySenderUnread,
tuiMessageOrderRead, playReadUrgentFirst, arrangeBySenderRead,
tuiMessageOrderSaved, playSavedUrgentFirst, arrangeBySenderSaved.
            -->
            <xs:element name="tuiMessageOrder" maxOccurs="1"
```

```
                                minOccurs="0"> <!-- MM field -->
                                <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                                <xs:enumeration value="urgent first then
newest" />
                                                <xs:enumeration value="oldest messages
first" />
                                                <xs:enumeration value="newest messages
first" />
                                                <xs:enumeration value="urgent first then
oldest" />
                                        </xs:restriction>
                                </xs:simpleType>
                        </xs:element>

                        <xs:element name="mwiEnabled" type="csm:xmlMwiType"
                                maxOccurs="1" minOccurs="0" />

                        <!--
                                Specifies whether a subscriber can receive messages, e-
mail messages
                                and call-answer messages from other subscribers. You
can choose one
                                of the following: - yes: to allow the subscriber to create,
forward,
                                and receive messages. - no: to prevent the subscriber
from receiving
                                call-answer messages and to hide the subscriber from the
telephone
                                user interface (TUI). The subscriber cannot use the TUI to
access
                                the mailbox, and other TUI users cannot address
messages to the
                                subscriber.
                                Note: voiceMailEnabled is no longer used since
SMGR_10.1.0.0. However, we don't remove the related code from
messagingCommProfile.xsd in order to support backward compatibility.
```

```
-->
<xs:element name="voiceMailEnabled"
type="csm:xmlTrueFalseType"
maxOccurs="1" minOccurs="0" />

<!--
Specifies additional, useful information about a
subscriber. Entries
in this field are for convenience and are not used by the
messaging
system.
-->
<xs:element name="miscellaneous1"
type="csm:xmlLength51Type"
maxOccurs="1" minOccurs="0" />

<!--
Specifies additional, useful information about a
subscriber. Entries
in this field are for convenience and are not used by the
messaging
system.
-->
<xs:element name="miscellaneous2"
type="csm:xmlLength51Type"
maxOccurs="1" minOccurs="0" />

<!--
Specifies additional, useful information about a
subscriber. Entries
in this field are for convenience and are not used by the
messaging
system.
Note: miscellaneous3 is no longer used since SMGR_10.1.0.0. However, we don't
remove the related code from messagingCommProfile.xsd in order to support backward
compatibility.
-->
```

```
                <xs:element name="miscellaneous3"
type="csm:xmlLength51Type"
                        maxOccurs="1" minOccurs="0" />


                <!--
                        Specifies additional, useful information about a
subscriber. Entries
                        in this field are for convenience and are not used by the
messaging
                        system.
                        Note: miscellaneous4 is no longer used since
SMGR_10.1.0.0. However, we don't remove the related code from
messagingCommProfile.xsd in order to support backward compatibility.
                -->
                <xs:element name="miscellaneous4"
type="csm:xmlLength51Type"
                        maxOccurs="1" minOccurs="0" />


                <!-- Specifies one or more alternate number to reach a
subscriber. You can
                        use secondary extensions to specify a telephone number
for direct reception
                        of faxes, to allow callers to use an existing Caller
Application, or to identify
                        each line appearance on the subscriber's telephone set if
they have different
                        telephone numbers. -->
                <xs:element name="secondaryExtension" maxOccurs="1"
                        minOccurs="0"> <!-- MM/CMM field -->
                        <xs:simpleType>
                                <xs:restriction base="xs:string">
                                        <xs:pattern value="([0-9]{3,50},){0,6}([0-9]
{3,50})?" />
                                </xs:restriction>
                        </xs:simpleType>
                </xs:element>
                <xs:element name="timezone" type="xs:string"
```

```
maxOccurs="1" minOccurs="0" />
<!--
The options to configure your own message playback order. The message playback order options depend on the following message types:
Unread messages: fields below define the order for unread messages in Aria or AUDIX TUI case and for all messages in CallPilot TUI case.
TUI Message Order:
Play newest first: The TUI plays the unread messages starting with the newest message first.
Play oldest first: The TUI plays the unread messages starting with the oldest message first.
Default value: The TUI plays the unread messages in order defined by selected Class Of Service (i.e. Play newest first or Play oldest first). This value is only for AAM 7.0 and above.
Play important messages before others: The TUI plays the unread urgent messages first. For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are:  Yes, No, Default value.
Arrange by sender: The TUI plays the unread messages that Messaging sorts by sender name. For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are:  Yes, No, Default value.
Read messages: fields below define the order for read messages in Aria or AUDIX TUI case.
TUI Message Order:
Play newest first: The TUI plays the read messages starting with the newest message first.
Play oldest first: The TUI plays the read messages starting with the oldest message first.
Default value: The TUI plays the read messages in order defined by selected Class Of Service (i.e. Play newest first or Play oldest first). This value is only for AAM 7.0 and above.
Play important messages before others: The TUI plays the read urgent messages first. For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are:  Yes, No, Default value.
Arrange by sender: The TUI plays the read messages that Messaging sorts by sender name. For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are:  Yes, No, Default value.
Saved messages: fields below define the order for saved messages in Aria TUI case.
TUI Message Order:
```

Play newest first: The TUI plays the saved messages starting with the newest message first.

Play oldest first: The TUI plays the saved messages starting with the oldest message first.

Default value: The TUI plays the saved messages in order defined by selected Class Of Service (i.e. Play newest first or Play oldest first). This value is only for AAM 7.0 and above.

Play important messages before others: The TUI plays the saved urgent messages first. For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are: Yes, No, Default value.

Arrange by sender: The TUI plays the saved messages that Messaging sorts by sender name. For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are: Yes, No, Default value.

```
-->
<!-- For unread messages -->
<xs:element name="tuiMessageOrderUnread"
type="csm:xmlAAMTuiOrderType"
        maxOccurs="1" minOccurs="0" />


<xs:element name="playUnreadUrgentFirst"
type="csm:xmlyesNoDefaultType"
        maxOccurs="1" minOccurs="0" />


<xs:element name="arrangeBySenderUnread"
type="csm:xmlyesNoDefaultType"
        maxOccurs="1" minOccurs="0" />


<!-- For read messages -->
<xs:element name="tuiMessageOrderRead"
type="csm:xmlAAMTuiOrderType"
        maxOccurs="1" minOccurs="0" />


<xs:element name="playReadUrgentFirst"
type="csm:xmlyesNoDefaultType"
        maxOccurs="1" minOccurs="0" />


<xs:element name="arrangeBySenderRead"
type="csm:xmlyesNoDefaultType"
```

```
                                maxOccurs="1" minOccurs="0" />


                        <!-- For saved messages -->
                        <xs:element name="tuiMessageOrderSaved"
type="csm:xmlAAMTuiOrderType"
                                maxOccurs="1" minOccurs="0" />


                        <xs:element name="playSavedUrgentFirst"
type="csm:xmlyesNoDefaultType"
                                maxOccurs="1" minOccurs="0" />


                        <xs:element name="arrangeBySenderSaved"
type="csm:xmlyesNoDefaultType"
                                maxOccurs="1" minOccurs="0" />
                </xs:sequence>
        </xs:complexType>

        <xs:simpleType name="xmlyesNoType">
                <xs:restriction base="xs:string">
                        <xs:enumeration value="Yes" />
                        <xs:enumeration value="No" />
                </xs:restriction>
        </xs:simpleType>

        <xs:simpleType name="xmlTrueFalseType">
                <xs:restriction base="xs:string">
                        <xs:enumeration value="TRUE" />
                        <xs:enumeration value="FALSE" />
                </xs:restriction>
        </xs:simpleType>

        <xs:simpleType name="xmlLength11Type">
                <xs:restriction base="xs:string">
                        <xs:maxLength value="11" />
                </xs:restriction>
```

```
        </xs:simpleType>

        <xs:simpleType name="xmlLength51Type">
                <xs:restriction base="xs:string">
                        <xs:maxLength value="51" />
                </xs:restriction>
        </xs:simpleType>

        <xs:simpleType name="xmlMssExchangeType">
                <xs:restriction base="xs:string">
                        <xs:enumeration value="Avaya Message Store" />
                        <xs:enumeration value="Microsoft Exchange" />
                </xs:restriction>
        </xs:simpleType>

        <xs:simpleType name="xmlMwiType">
                <xs:restriction base="xs:string">
                        <xs:enumeration value="No" />
                        <xs:enumeration value="ByCOS" />
                </xs:restriction>
        </xs:simpleType>
</xs:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Agent

Note: System Manager 7.0.1 introduces a new field in Agent XSD: auxAgentConsideredIdleMIA

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:one="http://xml.avaya.com/schema/import"
elementFormDefault="qualified"
targetNamespace="http://xml.avaya.com/schema/import_csm_agent"
```

```
xmlns:csm="http://xml.avaya.com/schema/import_csm_agent">
<xs:import namespace="http://xml.avaya.com/schema/import"
schemaLocation="userimport.xsd"/>


<!--Changes in xsd file need to generate jaxb src using this
xsd-->
<xs:complexType name="xmlAgentProfile">
     <xs:complexContent>
          <xs:extension base="one:xmlCommProfileType" >
               <xs:sequence>
                    <!-- CM Name as it appears under
'Applications/Application Management/Entities -->
                    <xs:element name="cmName" type="xs:string"
maxOccurs="1" minOccurs="1"/>


                    <!-- 'true' if already created extension
is to be used.  'false' if available extension is to be used.
-->
                    <xs:element name="useExistingAgent"
type="xs:boolean" maxOccurs="1" minOccurs="0"/>


                    <!-- Extension Range which will be used to
create Agent using available extension within given range -->
                    <xs:element name="extensionRange"
maxOccurs="1" minOccurs="0">
                         <xs:simpleType>
                              <xs:restriction
base="xs:string">
                                   <xs:pattern value="([0-9]+
([\.\-][0-9]+)*)|([0-9]+([\.\-][0-9]+)*:[0-9]+([\.\-][0-
9]+)*)"/>
                              </xs:restriction>
                         </xs:simpleType>
                    </xs:element>


                    <!-- Agent Login ID extension number that
need to be assigned to the user. -->
                    <xs:element name="loginIdExtension"
```

```
maxOccurs="1" minOccurs="1">
                          <xs:simpleType>
                               <xs:restriction
base="xs:string">
                                    <xs:pattern value="([0-9]+
([\.\-][0-9]+)*)|[nN][eE][xX][tT]"/>
                               </xs:restriction>
                          </xs:simpleType>
                     </xs:element>


                     <!-- Template name to be used to create
agent. Values defined in Template will be used if not provided.
-->
                     <xs:element name="template"
type="xs:string" maxOccurs="1" minOccurs="0"/>


                     <!-- Security code for station. Value can
be digit only.  -->
                     <xs:element name="securityCode"
maxOccurs="1" minOccurs="0">
                          <xs:simpleType>
                               <xs:restriction
base="xs:string">
                                    <xs:pattern value="[0-9]
{0,4}"/>
                               </xs:restriction>
                          </xs:simpleType>
                     </xs:element>


                     <xs:element name="aas" type="xs:boolean"
maxOccurs="1" minOccurs="0"/>
                     <xs:element name="audix" type="xs:boolean"
maxOccurs="1" minOccurs="0"/>


                     <xs:element name="password" maxOccurs="1"
minOccurs="0">
                          <xs:simpleType>
```

```
                                    <xs:restriction
base="xs:string">
                                         <xs:pattern value="[0-9]
{0,9}" />
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>

                        <xs:element name="portExtension"
maxOccurs="1" minOccurs="0">
                              <xs:simpleType>
                                    <xs:restriction
base="xs:string">
                                         <xs:pattern value="[0-9]+
([\.\-][0-9]+)*"/>
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>

                        <!-- Whether the agent should be deleted
if it unassigned from the user. -->
                        <xs:element name="deleteOnUnassign"
type="xs:boolean" maxOccurs="1" minOccurs="0"/>

                        <!-- CM dependent field for max value -->
                        <xs:element name="tn" maxOccurs="1"
minOccurs="0">
                        <xs:simpleType>
                                    <xs:restriction base="xs:int">
                                         <xs:minInclusive value="1"
/>
                                         <xs:maxInclusive
value="250" />
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>
```

```
                        <xs:element
name="checkSkillTnToMatchAgentTn" type="xs:boolean"
maxOccurs="1" minOccurs="0" />
                        <xs:element
name="includeTenantCallingPermissions" type="xs:boolean"
maxOccurs="1" minOccurs="0" />


                        <xs:element name="cor" maxOccurs="1"
minOccurs="0">
                              <xs:simpleType>
                              <xs:restriction base="xs:int">
                                         <xs:minInclusive
value="0"/>
                                         <xs:maxInclusive
value="995"/>
                              </xs:restriction>
                              </xs:simpleType>
                        </xs:element>


                       <!--Coverage path = Enter path number
between 1-9999, time of day table t1-t999, or blank - CM
Dependent-->
                        <xs:element name="coveragePath"
maxOccurs="1" minOccurs="0">
                              <xs:simpleType>
                                    <xs:restriction
base="xs:string">
                                         <xs:pattern value="(t[1-9]
[0-9]{0,2})|([1-9]{0})|([1-9][0-9]{0,3})"/>
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>


                        <xs:element name="lwcReception"
maxOccurs="1" minOccurs="0">
                              <xs:simpleType>
                                    <xs:restriction
```

```
base="xs:string">
                                        <xs:enumeration
value="audix"/>
                                        <xs:enumeration
value="msa"/>
                                        <xs:enumeration
value="spe"/>
                                        <xs:enumeration
value="none"/>
                                  </xs:restriction>
                            </xs:simpleType>
                       </xs:element>


                       <xs:element name="lwcLogExternalCalls"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                       <xs:element name="audixNameforMessaging"
type="xs:string" maxOccurs="1" minOccurs="0" />
                       <xs:element
name="hearsServiceObservingTone" type="xs:boolean" maxOccurs="1"
minOccurs="0" />
                       <xs:element
name="loginIDforISDNSIPDisplay" type="xs:boolean" maxOccurs="1"
minOccurs="0" />


                       <xs:element name="autoAnswer"
maxOccurs="1" minOccurs="0">
                            <xs:simpleType>
                                  <xs:restriction
base="xs:string">
                                        <xs:enumeration
value="acd"/>
                                        <xs:enumeration
value="all"/>
                                        <xs:enumeration
value="none"/>
                                        <xs:enumeration
value="station"/>
                                  </xs:restriction>
```

```
                        </xs:simpleType>
                    </xs:element>

                    <xs:element name="miaAcrossSkills"
maxOccurs="1" minOccurs="0">
                        <xs:simpleType>
                            <xs:restriction
base="xs:string">
                                <xs:enumeration
value="n"/>
                                <xs:enumeration
value="y"/>
                                <xs:enumeration
value="system"/>
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>

                    <xs:element name="acwAgentConsideredIdle"
maxOccurs="1" minOccurs="0">
                        <xs:simpleType>
                            <xs:restriction
base="xs:string">
                                <xs:enumeration
value="n"/>
                                <xs:enumeration
value="y"/>
                                <xs:enumeration
value="system"/>
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>

                    <xs:element name="auxWorkReasonCodeType"
maxOccurs="1" minOccurs="0">
                        <xs:simpleType>
                            <xs:restriction
```

```
base="xs:string">
                                        <xs:enumeration
value="forced"/>
                                        <xs:enumeration
value="requested"/>
                                        <xs:enumeration
value="system"/>
                                        <xs:enumeration
value="none"/>
                                  </xs:restriction>
                            </xs:simpleType>
                      </xs:element>


                      <xs:element name="logoutReasonCodeType"
maxOccurs="1" minOccurs="0">
                            <xs:simpleType>
                                  <xs:restriction
base="xs:string">
                                        <xs:enumeration
value="forced"/>
                                        <xs:enumeration
value="requested"/>
                                        <xs:enumeration
value="system"/>
                                        <xs:enumeration
value="none"/>
                                  </xs:restriction>
                            </xs:simpleType>
                      </xs:element>


                      <xs:element
name="maximumTimeAgentInAcwBeforeLogoutSec" maxOccurs="1"
minOccurs="0">
                            <xs:simpleType>
                                  <xs:restriction
base="xs:string">
                                        <xs:pattern value="|[3-9]
[0-9]{1}|[1-9][0-9]{1,3}|(none)|(system)"/>
```

```
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>


                        <xs:element name="forcedAgentLogoutTimeHr"
maxOccurs="1" minOccurs="0">
                              <xs:simpleType>
                                    <xs:restriction
base="xs:string">
                                          <xs:pattern value="|[0-9]|
[1][0-9]{1}|[2][0-3]{1}"/>
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>


                        <xs:element
name="forcedAgentLogoutTimeSec" maxOccurs="1" minOccurs="0">
                              <xs:simpleType>
                                    <xs:restriction
base="xs:string">
                                          <xs:pattern value="|(00)|
(15)|(30)|(45)"/>
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>


                        <xs:element name="directAgentSkill"
maxOccurs="1" minOccurs="0">
                              <xs:simpleType>
                                    <xs:restriction
base="xs:string">
                                          <xs:pattern value="|[1-9]|
[1-9][0-9]{0,2}|[1-7][0-9]{3}|8000"/>
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>
```

```
                        <xs:element name="callHandlingPreference"
maxOccurs="1" minOccurs="0">
                              <xs:simpleType>
                                    <xs:restriction
base="xs:string">
                                          <xs:enumeration
value="greatest-need"/>
                                          <xs:enumeration
value="percent-allocation"/>
                                          <xs:enumeration
value="skill-level"/>
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>


                        <xs:element name="serviceObjective"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                        <xs:element name="directAgentCallsFirst"
type="xs:boolean" maxOccurs="1" minOccurs="0" />
                        <xs:element name="localCallPreference"
type="xs:boolean" maxOccurs="1" minOccurs="0" />


                        <xs:element
name="auxAgentConsideredIdleMIA" maxOccurs="1" minOccurs="0">
                              <xs:simpleType>
                                    <xs:restriction
base="xs:string">
                                          <xs:enumeration
value="n"/>
                                          <xs:enumeration
value="y"/>
                                          <xs:enumeration
value="system"/>
                                    </xs:restriction>
                              </xs:simpleType>
                        </xs:element>
```

```
                        <xs:element name="skills"
type="csm:xmlAgentLoginIdSkillsData" maxOccurs="unbounded"
minOccurs="0" />


                        <xs:element  name="nativeName"
type="csm:xmlNativeNameData" maxOccurs="1" minOccurs="0"/>


                        <!--


                        private String NativeNameScripts;


                         -->
                  </xs:sequence>
            </xs:extension>
   </xs:complexContent>
</xs:complexType>


<xs:complexType name="xmlAgentLoginIdSkillsData">
      <xs:sequence>
      <!--
            private AgentLoginIdData agentLoginId;


       -->
            <xs:element name="number" type="xs:string"
maxOccurs="1" minOccurs="1" />
            <xs:element name="skillNumber" maxOccurs="1"
minOccurs="1">
                  <xs:simpleType>
                        <xs:restriction base="xs:string">
                              <xs:pattern value="|[1-9][0-9]{0,2}|
[1-7][0-9]{3}|8000"/>
                        </xs:restriction>
                  </xs:simpleType>
            </xs:element>
```

```
            <xs:element name="reserveLevel" maxOccurs="1"
minOccurs="0" >
                  <xs:simpleType>
                        <xs:restriction base="xs:string">
                              <xs:pattern value="|a|m|n|[1-2]"/>
                        </xs:restriction>
                  </xs:simpleType>
            </xs:element>

            <xs:element name="skillLevel" maxOccurs="1"
minOccurs="0" >
                  <xs:simpleType>
                        <xs:restriction base="xs:string">
                              <xs:pattern value="|[1-9]|[1-9][0-6]
{1}"/>
                        </xs:restriction>
                  </xs:simpleType>
            </xs:element>

            <xs:element name="percentAllocation" maxOccurs="1"
minOccurs="0" >
                  <xs:simpleType>
                        <xs:restriction base="xs:string">
                              <xs:pattern value="|[1-9]|[1-9][0-9]
{1}|100"/>
                        </xs:restriction>
                  </xs:simpleType>
            </xs:element>



      </xs:sequence>
</xs:complexType>


<!-- If displayName,givenName or surname contains characters of
```

```
multiple scripts then locale tag should be present.
       If displayName tag is present then it overwrites native
name.
       If displayname is not present then combination of
givenName and surname gets copied in native name.
       Please find below locale for multiscript language
       Language               Locale
       Japanese                   ja, ja-jp
       Simplified Chinese         zh-cn
       Traditional Chinese   zh-tw-->
<xs:complexType name="xmlNativeNameData">
      <xs:sequence>
           <xs:element name="locale" maxOccurs="1"
minOccurs="0">
              <xs:simpleType>
                      <xs:restriction base="xs:string">
                      <xs:enumeration value="ja-jp"/>
                      <xs:enumeration value="ja"/>
                      <xs:enumeration value="zh-cn"/>
                      <xs:enumeration value="zh-tw"/>

                      </xs:restriction>
                 </xs:simpleType>
           </xs:element>
           <xs:element name="name" maxOccurs="1" minOccurs="0" >
                 <xs:simpleType>
                      <xs:restriction base="xs:string">
                           <xs:maxLength value="27"/>
                      </xs:restriction>
                 </xs:simpleType>
           </xs:element>
      </xs:sequence>
</xs:complexType>

</xs:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): CS1000 and Callpilot

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     xmlns:one="http://xml.avaya.com/schema/import"
     targetNamespace="http://xml.avaya.com/schema/import1"
     elementFormDefault="qualified"
     xmlns:abc="http://xml.avaya.com/schema/import1">
<xsd:import namespace="http://xml.avaya.com/schema/import"
            schemaLocation="userimport.xsd"/>
<xsd:complexType name="AccountCommProfileType">
    <xsd:complexContent>
        <xsd:extension base="one:xmlCommProfileType" >
            <xsd:sequence>
               <xsd:element name="serviceDetails" type="xsd:string"
minOccurs="0"/>
               <xsd:element name="element" type="xsd:string"
minOccurs="0"/>
               <xsd:element name="target" type="xsd:string"
minOccurs="0"/>
               <xsd:element name="template" type="xsd:string"
minOccurs="0"/>
               <xsd:element name="serviceType" type="xsd:string"
minOccurs="0"/>
               <xsd:element name="accountDetails" type="xsd:string"
minOccurs="0"/>
               <xsd:element name="accountProperties"
type="abc:AccountPropertyType" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
```

```
<xsd:complexType name="AccountPropertyType">
   <xsd:sequence>
      <xsd:element name="propertyName" type="xsd:string"/>
      <xsd:element name="propertyValue" type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Avaya Branch Gateway

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:one="http://xml.avaya.com/schema/import"
elementFormDefault="qualified"
      targetNamespace="http://xml.avaya.com/schema/import_csm_b5800"
xmlns:csm="http://xml.avaya.com/schema/import_csm_b5800">


      <xs:import namespace="http://xml.avaya.com/schema/import"
            schemaLocation="userimport.xsd" />


      <!--Changes in xsd file need to generate jaxb src using this xsd-->
      <xs:complexType name="xmlB5800UserProfile">
            <xs:complexContent>
                  <xs:extension base="one:xmlCommProfileType">
                        <xs:sequence>
                              <!--
                                    B5800/B5800L Device Name as it appears
under 'Applications/Application
                                    Management/Entities
                              -->
                              <xs:element name="deviceName" type="xs:string"
```

```
maxOccurs="1"
                                minOccurs="1" />

                            <!--
                                Template name to be used to create
station. Values defined in
                                Template will be used if not provided.
                            -->
                            <xs:element name="userTemplate"
type="xs:string"
                                maxOccurs="1" minOccurs="0" />

                            <xs:element name="useExistingExt"
type="xs:boolean"
                                maxOccurs="1" minOccurs="0" />

                            <!-- extension number that need to be assigned
to the user. -->
                            <xs:element name="extension" maxOccurs="1"
minOccurs="1">
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                        <xs:pattern value="[0-9]+
([\.\-][0-9]+)*" />
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:element>

                            <!-- Specifies the type of the extn  -->
                            <xs:element name="extensionType" maxOccurs="1"
                                minOccurs="1">
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                        <xs:enumeration value="Analog"
/>
                                        <xs:enumeration value="IPDECT"
```

```
/>
                                                <xs:enumeration value="Sip" />
                                                <xs:enumeration
value="Digital" />
                                                <xs:enumeration
value="H323" />
                                          </xs:restriction>
                                     </xs:simpleType>
                               </xs:element>


                               <xs:element name="deleteExtOnUserDelete"
type="xs:boolean"
                                     maxOccurs="1" minOccurs="0" />


                               <xs:element name="data"
type="csm:xmlB5800UserProfileData"
                                     maxOccurs="1" minOccurs="0" />
                          </xs:sequence>
                     </xs:extension>
                </xs:complexContent>
           </xs:complexType>


           <xs:complexType name="xmlB5800UserProfileData">
                <xs:sequence>
                     <xs:element name="ws_object"
type="csm:xmlB5800UserConfig">
                     </xs:element>
                </xs:sequence>
           </xs:complexType>


           <xs:complexType name="xmlB5800UserConfig">
                <xs:sequence>
                     <xs:element name="Extension"
type="csm:xmlB5800ExtensionInfo">
                     </xs:element>
```

```
                <xs:element name="User" type="csm:xmlB5800UserInfo">
                </xs:element>
          </xs:sequence>
     </xs:complexType>

     <xs:complexType name="xmlB5800ExtensionInfo">
          <xs:sequence>
                <xs:element name="Id" type="xs:int" minOccurs="0" />
                <xs:element name="Extension" type="xs:string"
minOccurs="0" />
                <xs:element name="TypeInfo" type="xs:int"
minOccurs="0" />
                <xs:element name="CallerDisplayType" type="xs:int"
minOccurs="0" />
                <xs:element name="MessageLampType" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnClassification" type="xs:int"
minOccurs="0" />
                <xs:element name="LineType" type="xs:int"
minOccurs="0" />
                <xs:element name="MinFlashPulseWidth" type="xs:int"
minOccurs="0" />
                <xs:element name="MaxFlashPulseWidth" type="xs:int"
minOccurs="0" />
                <xs:element name="UseSystemFlashHook" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ResetVolumeAfterCalls"
type="xs:boolean" minOccurs="0" />
                <xs:element name="DisconnectPulseWidth" type="xs:int"
minOccurs="0" />
                <xs:element name="HookPersistency" type="xs:int"
minOccurs="0" />
                <xs:element name="Mac" type="xs:string" minOccurs="0" />
                <xs:element name="SilenceSuppression" type="xs:boolean"
minOccurs="0" />
                <xs:element name="VoicePktSize" type="xs:int"
minOccurs="0" />
                <xs:element name="VoiceCompression" type="xs:int"
```

```
minOccurs="0" />
                <xs:element name="voip" type="csm:xmlVoip"
minOccurs="0" />
                <xs:element name="RenegotiationSupported"
type="xs:boolean" minOccurs="0" />
                <xs:element name="RenegotiateBeforeConnect"
type="xs:boolean" minOccurs="0" />
                <xs:element name="UseVocoder" type="xs:boolean"
minOccurs="0" />
                <xs:element name="EarlyH245Supported" type="xs:boolean"
minOccurs="0" />
                <xs:element name="RFC2833" type="xs:boolean"
minOccurs="0" />
                <xs:element name="MediaWait" type="xs:boolean"
minOccurs="0" />
                <xs:element name="MediaOnOverlap" type="xs:boolean"
minOccurs="0" />
                <xs:element name="PauseRequired" type="xs:boolean"
minOccurs="0" />
                <xs:element name="PauseOnEndRequired" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ParallelH245" type="xs:boolean"
minOccurs="0" />
                <xs:element name="AnnexFSupported" type="xs:boolean"
minOccurs="0" />
                <xs:element name="PhoneType" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPIAudio_setting" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPIHeadset_setting" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPIContrast" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPIRedial_time" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPISpeaker_volume" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPIHandsfree_settings"
type="xs:int" minOccurs="0" />
                <xs:element name="ExtnAPIRingtone_volume" type="xs:int"
```

```
minOccurs="0" />
                <xs:element name="ExtnAPIDoor_phone" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ExtnAPIHandset_volume" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPIRingtone_speed" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPIHeadset_volume" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPIHeadset_config" type="xs:int"
minOccurs="0" />
                <xs:element name="ExtnAPIAlpha_keypad_layout"
type="xs:int" minOccurs="0" />
                <xs:element name="ExtnAPIDirect_dial_enabled"
type="xs:boolean" minOccurs="0" />
                <xs:element name="ExtnAPIHandsfree_enabled"
type="xs:boolean" minOccurs="0" />
                <xs:element name="T38Fax" type="csm:xmlT38Fax"
minOccurs="0" />
                <xs:element name="SipExtn" type="csm:xmlSipExtn"
minOccurs="0" />
                <xs:element name="DisableSpeaker" type="xs:boolean"
minOccurs="0" />
                <xs:element name="VPNExtn" type="xs:boolean"
minOccurs="0" />
                <xs:element name="IPAvayaLicenseReserved"
type="xs:boolean" minOccurs="0" />
                <xs:element name="IPEndpointsLicenseReserved"
type="xs:boolean" minOccurs="0" />
                <xs:element name="IsExtnCentralized" type="xs:boolean"
minOccurs="0" />
                <xs:element name="CentralizedDDINumber" type="xs:string"
minOccurs="0" />
                <xs:element name="ExtnDS" type="csm:xmlExtnDS"
minOccurs="0" />
                <xs:element name="SpecificBstType" type="xs:int"
minOccurs="0" />
           </xs:sequence>
           <xs:attribute name="GUID" type="xs:string" />
```

```
      </xs:complexType>


      <xs:complexType name="xmlB5800UserInfo">
            <xs:sequence>
                  <xs:element name="UserRightsView" type="xs:string"
minOccurs="0" />
                  <xs:element name="UsingView" type="xs:boolean"
minOccurs="0" />
                  <xs:element name="UserRightsTimeProfile" type="xs:string"
minOccurs="0" />
                  <xs:element name="OutOfHoursUserRights" type="xs:string"
minOccurs="0" />
                  <xs:element name="Name" type="xs:string" minOccurs="0" />
                  <xs:element name="KName" type="xs:string"
minOccurs="0" />
                  <xs:element name="Password" type="xs:string"
minOccurs="0" />
                  <xs:element name="FullName" type="xs:string"
minOccurs="0" />
                  <xs:element name="Extension" type="xs:string"
minOccurs="0" />
                  <xs:element name="Priority" type="xs:int"
minOccurs="0" />
                  <xs:element name="OutsideCallSeq" type="xs:int"
minOccurs="0" />
                  <xs:element name="InsideCallSeq" type="xs:int"
minOccurs="0" />
                  <xs:element name="RingbackCallSeq" type="xs:int"
minOccurs="0" />
                  <xs:element name="NoAnswerTime" type="xs:int"
minOccurs="0" />
                  <xs:element name="ForwardOnBusy" type="xs:boolean"
minOccurs="0" />
                  <xs:element name="BookConferenceWithPM" type="xs:boolean"
minOccurs="0" />
                  <xs:element name="DisableForwardOnInt" type="xs:boolean"
minOccurs="0" />
                  <xs:element name="DisableForwardUncondOnInt"
type="xs:boolean" minOccurs="0" />
```

```
                <xs:element name="DisableForwardBusyNoAnsOnInt"
type="xs:boolean" minOccurs="0" />
                <xs:element name="VoicemailReception2" type="xs:string"
minOccurs="0" />
                <xs:element name="VoicemailReception3" type="xs:string"
minOccurs="0" />
                <xs:element name="DSSKeys" type="csm:xmlDSSKeys"
minOccurs="0" />
                <xs:element name="InhibitOffSwitchForwarding"
type="xs:boolean" minOccurs="0" />
                <xs:element name="IsNoUser" type="xs:boolean"
minOccurs="0" />
                <xs:element name="IsRealUser" type="xs:boolean"
minOccurs="0" />
                <xs:element name="IsRemoteManager" type="xs:boolean"
minOccurs="0" />
                <xs:element name="IsVoiceEmailModeAlert"
type="xs:boolean" minOccurs="0" />
                <xs:element name="IsVoiceEmailModeCopy" type="xs:boolean"
minOccurs="0" />
                <xs:element name="IsVoiceEmailModeForward"
type="xs:boolean" minOccurs="0" />
                <xs:element name="IsVoiceEmailModeOff" type="xs:boolean"
minOccurs="0" />
                <xs:element name="MaxTwinnedCalls" type="xs:int"
minOccurs="0" />
                <xs:element name="PhoneManagerCallStatusOptions"
type="xs:long" minOccurs="0" />
                <xs:element name="PhoneManagerCloseOptions" type="xs:int"
minOccurs="0" />
                <xs:element name="PhoneManagerCanChange"
type="xs:boolean" minOccurs="0" />
                <xs:element name="PhoneManagerConfigureOptions"
type="xs:int" minOccurs="0" />
                <xs:element name="PhoneManagerOptions" type="xs:int"
minOccurs="0" />
                <xs:element name="PhoneManagerOptionsOriginal"
type="xs:int" minOccurs="0" />
                <xs:element name="PhoneType" type="xs:int"
minOccurs="0" />
```

```
                <xs:element name="PhoneTypeIndex" type="xs:int"
minOccurs="0" />
                <xs:element name="PopupAnswering" type="xs:boolean"
minOccurs="0" />
                <xs:element name="PopupExternal" type="xs:boolean"
minOccurs="0" />
                <xs:element name="PopupInternal" type="xs:boolean"
minOccurs="0" />
                <xs:element name="PopupOutlook" type="xs:boolean"
minOccurs="0" />
                <xs:element name="PopupRinging" type="xs:boolean"
minOccurs="0" />
                <xs:element name="PopupOptions" type="xs:int"
minOccurs="0" />
                <xs:element maxOccurs="unbounded" name="RingDelay"
type="xs:int" minOccurs="0" />
                <xs:element name="ShowAccountCodes" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ShowAllCalls" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ShowCallStatus" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ShowCostOfCall" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ShowIncoming" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ShowMessages" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ShowMissed" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ShowOutgoing" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ShowSpeedDials" type="xs:boolean"
minOccurs="0" />
                <xs:element name="StartInCompactMode" type="xs:boolean"
minOccurs="0" />
                <xs:element name="StayInCompactModeOnIncommingCall"
type="xs:boolean" minOccurs="0" />
                <xs:element name="StayInCompaceModeOnOutgoingCall"
type="xs:boolean" minOccurs="0" />
```

```
                <xs:element name="T3AllowThirdPartyFwd" type="xs:boolean"
minOccurs="0" />
                <xs:element name="T3ProtectFromThirdPartyFwd"
type="xs:boolean" minOccurs="0" />
                <xs:element name="TwinnedDialDelay" type="xs:int"
minOccurs="0" />
                <xs:element name="TwinnedEligibleForForwarded"
type="xs:boolean" minOccurs="0" />
                <xs:element name="TwinnedEligibleForGroup"
type="xs:boolean" minOccurs="0" />
                <xs:element name="TwinnedMobileNumber" type="xs:string"
minOccurs="0" />
                <xs:element name="TwinnedTimeProfile" type="xs:string"
minOccurs="0" />
                <xs:element name="TwinningNumber" type="xs:string"
minOccurs="0" />
                <xs:element name="TwinningType" type="xs:int"
minOccurs="0" />
                <xs:element name="ForwardOnNoAnswer" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ForwardUnconditional" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ForwardHuntGroupCalls"
type="xs:boolean" minOccurs="0" />
                <xs:element name="ForwardNumber" type="xs:string"
minOccurs="0" />
                <xs:element name="ForwardBusyNumber" type="xs:string"
minOccurs="0" />
                <xs:element name="DoNotDisturb" type="xs:boolean"
minOccurs="0" />
                <xs:element name="DNDExceptions" type="xs:string"
minOccurs="0" />
                <xs:element name="OutgoingCallBar" type="xs:boolean"
minOccurs="0" />
                <xs:element name="OffHookStation" type="xs:boolean"
minOccurs="0" />
                <xs:element name="BusyOnHeld" type="xs:boolean"
minOccurs="0" />
                <xs:element name="FollowMeNumber" type="xs:string"
minOccurs="0" />
```

```
                <xs:element name="CallWaitingOn" type="xs:boolean"
minOccurs="0" />
                <xs:element name="VoicemailOn" type="xs:boolean"
minOccurs="0" />
                <xs:element name="VoicemailHelp" type="xs:boolean"
minOccurs="0" />
                <xs:element name="VoicemailCode" type="xs:string"
minOccurs="0" />
                <xs:element name="VoicemailEmail" type="xs:string"
minOccurs="0" />
                <xs:element name="VoicemailEmailReading"
type="xs:boolean" minOccurs="0" />
                <xs:element name="VoicemailReception" type="xs:string"
minOccurs="0" />
                <xs:element name="VoicemailEmailMode" type="xs:int"
minOccurs="0" />
                <xs:element name="VoicemailRingback" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ShortCodes" type="csm:xmlShortCodes"
minOccurs="0" />
                <xs:element name="DialInOn" type="xs:boolean"
minOccurs="0" />
                <xs:element name="DialInTimeProfile" type="xs:string"
minOccurs="0" />
                <xs:element name="DialInFirewallProfile" type="xs:string"
minOccurs="0" />
                <xs:element name="SourceNumbers" type="xs:string"
minOccurs="0" />
                <xs:element name="DialInQuotaTime" type="xs:int"
minOccurs="0" />
                <xs:element name="LoginCode" type="xs:string"
minOccurs="0" />
                <xs:element name="LoginIdleTime" type="xs:string"
minOccurs="0" />
                <xs:element name="WrapUpTime" type="xs:int" minOccurs="0"
/>
                <xs:element name="TwinMaster" type="xs:string"
minOccurs="0" />
                <xs:element name="SecTwinCallEnabled" type="xs:boolean"
minOccurs="0" />
```

```
                <xs:element name="CanIntrude" type="xs:boolean"
minOccurs="0" />
                <xs:element name="CannotBeIntruded" type="xs:boolean"
minOccurs="0" />
                <xs:element name="XDirectory" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ForceLogin" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ForceAuthCode" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ForceAccountCode" type="xs:boolean"
minOccurs="0" />
                <xs:element name="SystemPhone" type="xs:int"
minOccurs="0" />
                <xs:element name="AbsentMsg" type="xs:int"
minOccurs="0" />
                <xs:element name="AbsentSet" type="xs:int"
minOccurs="0" />
                <xs:element name="AbsentText" type="xs:string"
minOccurs="0" />
                <xs:element name="T3HuntGroupMembershipStatus"
type="xs:string" minOccurs="0" />
                <xs:element name="T3HuntGroupServiceStatus"
type="xs:string" minOccurs="0" />
                <xs:element name="T3DirectoryEntries" type="xs:string"
minOccurs="0" />
                <xs:element name="MonitorGroup" type="xs:string"
minOccurs="0" />
                <xs:element name="DisplayLocale" type="xs:string"
minOccurs="0" />
                <xs:element name="Locale" type="xs:string"
minOccurs="0" />
                <xs:element name="PMType" type="xs:int" minOccurs="0" />
                <xs:element name="InboundAutoRecord" type="xs:int"
minOccurs="0" />
                <xs:element name="OutboundAutoRecord" type="xs:int"
minOccurs="0" />
                <xs:element name="AutoRecordTimeProfile" type="xs:string"
minOccurs="0" />
                <xs:element name="RemoteWorker" type="xs:boolean"
```

```
minOccurs="0" />
                <xs:element name="CanAcceptCollectCalls"
type="xs:boolean" minOccurs="0" />
                <xs:element name="UserRights" type="xs:string"
minOccurs="0" />
                <xs:element name="Secretaries" type="xs:string"
minOccurs="0" />
                <xs:element name="TransferReturnTime" type="xs:string"
minOccurs="0" />
                <xs:element name="AnswerCallWaiting" type="xs:boolean"
minOccurs="0" />
                <xs:element name="RingingLinePreference"
type="xs:boolean" minOccurs="0" />
                <xs:element name="IdleLinePreference" type="xs:boolean"
minOccurs="0" />
                <xs:element name="CoverageTime" type="xs:int"
minOccurs="0" />
                <xs:element name="AutoVRL" type="xs:int" minOccurs="0" />
                <xs:element name="ManualVRL" type="xs:int"
minOccurs="0" />
                <xs:element name="DelayedRingPreference"
type="xs:boolean" minOccurs="0" />
                <xs:element name="AnswerPreSelect" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ReserveLastCA" type="xs:boolean"
minOccurs="0" />
                <xs:element name="CallTracingOn" type="xs:boolean"
minOccurs="0" />
                <xs:element name="DisplayCharges" type="xs:boolean"
minOccurs="0" />
                <xs:element name="MarkUpFactor" type="xs:int"
minOccurs="0" />
                <xs:element name="reset_longest_idle_info" type="xs:int"
minOccurs="0" />
                <xs:element name="NoAnswerStatus" type="xs:int"
minOccurs="0" />
                <xs:element name="PBXAddress" type="xs:string"
minOccurs="0" />
                <xs:element name="SIPName" type="xs:string" minOccurs="0"
/>
```

```
                <xs:element name="SIPDisplayName" type="xs:string"
minOccurs="0" />
                <xs:element name="SIPContact" type="xs:string"
minOccurs="0" />
                <xs:element name="SIPAnonymous" type="xs:boolean"
minOccurs="0" />
                <xs:element name="AbbreviatedRing" type="xs:boolean"
minOccurs="0" />
                <xs:element name="CustomerServiceRep" type="xs:boolean"
minOccurs="0" />
                <xs:element name="ACWTime" type="xs:int" minOccurs="0" />
                <xs:element name="AutoACW" type="xs:boolean"
minOccurs="0" />
                <xs:element name="UMSWebServices" type="xs:boolean"
minOccurs="0" />
                <xs:element name="DisableVMOnFU" type="xs:boolean"
minOccurs="0" />
                <xs:element name="DTMFCallCtrl" type="xs:boolean"
minOccurs="0" />
                <xs:element name="LoggedOutTwinning" type="xs:int"
minOccurs="0" />
                <xs:element name="OneXClient" type="xs:boolean"
minOccurs="0" />
                <xs:element name="MobilityFeatures" type="xs:boolean"
minOccurs="0" />
                <xs:element name="TwinnedBridgeAppearances"
type="xs:boolean" minOccurs="0" />
                <xs:element name="TwinnedCoverageAppearances"
type="xs:boolean" minOccurs="0" />
                <xs:element name="TwinnedLineAppearances"
type="xs:boolean" minOccurs="0" />
                <xs:element name="PersonalDirectory" type="xs:string"
minOccurs="0" />
                <xs:element name="ForwardToVoicemail" type="xs:boolean"
minOccurs="0" />
                <xs:element name="CoverageGroup" type="xs:string"
minOccurs="0" />
                <xs:element name="CanChangeHGOOSGroup" type="xs:string"
minOccurs="0" />
                <xs:element name="CanChangeHGONGroup" type="xs:string"
```

```
minOccurs="0" />
                <xs:element name="IncludeForwardInMenu" type="xs:boolean"
minOccurs="0" />
                <xs:element name="CallLoggingCentralised"
type="xs:string" minOccurs="0" />
                <xs:element name="AttentionRing" type="xs:string"
minOccurs="0" />
                <xs:element name="CoverageRing" type="xs:string"
minOccurs="0" />
                <xs:element name="LogMissedCallsForHG" type="xs:string"
minOccurs="0" />
                <xs:element name="DisableForwardToVoicemail"
type="xs:int" minOccurs="0" />
                <xs:element name="AnnouncementsOn" type="xs:boolean"
minOccurs="0" />
                <xs:element name="FollowAnnouncementsOn"
type="xs:boolean" minOccurs="0" />
                <xs:element name="LoopAnnouncementsOn" type="xs:boolean"
minOccurs="0" />
                <xs:element name="SyncAnnouncementsOn" type="xs:boolean"
minOccurs="0" />
                <xs:element name="FirstAnnTime" type="xs:int"
minOccurs="0" />
                <xs:element name="SecondAnnTime" type="xs:int"
minOccurs="0" />
                <xs:element name="BetweenAnnTime" type="xs:int"
minOccurs="0" />
                <xs:element name="PostAnnTone" type="xs:int"
minOccurs="0" />
                <xs:element name="PortalServices" type="xs:int"
minOccurs="0" />
                <xs:element name="WorkingHoursUserRightsGroup"
type="xs:string" minOccurs="0" />
                <xs:element name="T3SelfAdmin" type="xs:string"
minOccurs="0" />
                <xs:element name="MobileCallback" type="xs:boolean"
minOccurs="0" />
                <xs:element name="Receptionist" type="xs:boolean"
minOccurs="0" />
                <xs:element name="SoftPhone" type="xs:boolean"
```

```
minOccurs="0" />
                <xs:element name="OneXTelecommuter" type="xs:boolean"
minOccurs="0" />
                <xs:element name="AssignedPackage" type="xs:int"
minOccurs="0" />
                <xs:element name="AutoRecMode" type="xs:int"
minOccurs="0" />
                <xs:element name="CallLogTimeout" type="xs:string"
minOccurs="0" />
                <xs:element name="UserCLI" type="xs:string" minOccurs="0"
/>
           </xs:sequence>
           <xs:attribute name="GUID" type="xs:string" />
      </xs:complexType>

      <xs:complexType name="xmlDSSKeys">
           <xs:sequence>
                <xs:element minOccurs="0" maxOccurs="unbounded"
name="DSSKey"
                     type="csm:xmlDSSKey"/>

           </xs:sequence>
      </xs:complexType>

      <xs:complexType name="xmlDSSKey">
           <xs:sequence>
                <xs:element name="KeyType" type="xs:int" minOccurs="0"/>
                <xs:element name="Label" type="xs:string"
minOccurs="0" />
                <xs:element name="ActionObject" type="xs:string"
minOccurs="0" />
                <xs:element name="Data" type="xs:string" minOccurs="0" />
                <xs:element name="RingDelay" type="xs:int"
minOccurs="0" />
                <xs:element name="IdlePos" type="xs:string"
minOccurs="0"/>
```

```
            </xs:sequence>
            <xs:attribute name="Key" type="xs:int" />
      </xs:complexType>



      <xs:complexType name="xmlShortCodes">
            <xs:sequence>
                  <xs:element minOccurs="0" maxOccurs="unbounded"
name="ShortCode"
                        type="csm:xmlShortCode"  />


            </xs:sequence>
      </xs:complexType>


      <xs:complexType name="xmlShortCode">
            <xs:sequence>
                  <xs:element name="Code" type="xs:string" minOccurs="0" />
                  <xs:element name="TelephoneNumber" type="xs:string"
minOccurs="0" />
                  <xs:element name="LineGroupId" type="xs:int"
minOccurs="0" />
                  <xs:element name="Feature" type="xs:string" minOccurs="0"
/>
                  <xs:element name="Locale" type="xs:string"
minOccurs="0" />
                  <xs:element name="ForceAccountCode" type="xs:boolean"
minOccurs="0" />
                  <xs:element name="ForceAuthCode" type="xs:boolean"
minOccurs="0" />
            </xs:sequence>
            <xs:attribute name="GUID" type="xs:string" />
      </xs:complexType>


      <xs:complexType name="xmlVoip">
            <xs:sequence>
                  <xs:element name="GatekeeperPrimaryIPAddress"
```

```
type="xs:string" minOccurs="0" />
                <xs:element name="GatekeeperSecondaryIPAddress"
type="xs:string" minOccurs="0" />
                <xs:element name="IPAddress" type="xs:string"
minOccurs="0" />
                <xs:element name="EnableFaststart" type="xs:boolean"
minOccurs="0" />
                <xs:element name="FaxTransportSupport" type="xs:boolean"
minOccurs="0" />
                <xs:element name="LocalHoldMusic" type="xs:boolean"
minOccurs="0" />
                <xs:element name="LocalTones" type="xs:boolean"
minOccurs="0" />
                <xs:element name="RSVPEnabled" type="xs:boolean"
minOccurs="0" />
                <xs:element name="OOB_DTMF" type="xs:boolean"
minOccurs="0" />
                <xs:element name="AllowDirectMedia" type="xs:boolean"
minOccurs="0" />
                <xs:element name="H450Support" type="xs:int"
minOccurs="0" />
                <xs:element name="AnnexlSupport" type="xs:boolean"
minOccurs="0" />
                <xs:element name="InputGain" type="xs:int"
minOccurs="0" />
                <xs:element name="OutputGain" type="xs:int" minOccurs="0"
/>
                <xs:element name="MediaSecurity" type="xs:int"
minOccurs="0" />
                <xs:element name="RTP_Authentication" type="xs:boolean"
minOccurs="0" />
                <xs:element name="RTP_Encryption" type="xs:boolean"
minOccurs="0" />
                <xs:element name="RTCP_Authentication" type="xs:boolean"
minOccurs="0" />
                <xs:element name="RTCP_Encryption" type="xs:boolean"
minOccurs="0" />
                <xs:element name="SRTP_Window_Size" type="xs:string"
minOccurs="0" />
                <xs:element name="Crypto_Suite_SHA_80" type="xs:boolean"
```

```
minOccurs="0" />
                <xs:element name="Crypto_Suite_SHA_32" type="xs:boolean"
minOccurs="0" />


          </xs:sequence>
     </xs:complexType>


     <xs:complexType name="xmlSipExtn">
          <xs:sequence>
                <xs:element name="ForceAuthentication" type="xs:boolean"
minOccurs="0" />
                <xs:element name="Rel100Supported" type="xs:string"
minOccurs="0" />


          </xs:sequence>
     </xs:complexType>


     <xs:complexType name="xmlExtnDS">
          <xs:sequence>
                <xs:element name="AdmmUseHandsetConfig" type="xs:boolean"
minOccurs="0" />
                <xs:element name="AdmmType" type="xs:int"
minOccurs="0" />
                <xs:element name="AdmmIpei" type="xs:int"
minOccurs="0" />
                <xs:element name="AdmmAnonymous" type="xs:boolean"
minOccurs="0" />


          </xs:sequence>
     </xs:complexType>


     <xs:complexType name="xmlT38Fax">
          <xs:sequence>
                <xs:element name="Defaulted" type="xs:string"
minOccurs="0" />
                <xs:element name="T38FaxVersion" type="xs:string"
```

```
minOccurs="0" />
                <xs:element name="RedundancyLowSpeed" type="xs:string"
minOccurs="0" />
                <xs:element name="RedundancyHighSpeed" type="xs:string"
minOccurs="0" />
                <xs:element name="NSFOveride" type="xs:string"
minOccurs="0" />
                <xs:element name="NSFCountryCode" type="xs:string"
minOccurs="0" />
                <xs:element name="NSFVendorCode" type="xs:string"
minOccurs="0" />
                <xs:element name="TxNetworkTimeout" type="xs:string"
minOccurs="0" />
                <xs:element name="ScanLineFixup" type="xs:string"
minOccurs="0" />
                <xs:element name="TopEnhancement" type="xs:string"
minOccurs="0" />
                <xs:element name="DisableT30ECM" type="xs:string"
minOccurs="0" />
                <xs:element name="DisableT30MR" type="xs:string"
minOccurs="0" />
                <xs:element name="DisableEFlagsForFirstDis"
type="xs:string" minOccurs="0" />
                <xs:element name="EflagStartTimer" type="xs:string"
minOccurs="0" />
                <xs:element name="EflagStopTimer" type="xs:string"
minOccurs="0" />
                <xs:element name="FaxTransport" type="xs:string"
minOccurs="0" />
                <xs:element name="TCFMethod" type="xs:int"
minOccurs="0" />
                <xs:element name="MaxFaxRate" type="xs:int" minOccurs="0"
/>


           </xs:sequence>
     </xs:complexType>
</xs:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Presence

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

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:one="http://xml.avaya.com/schema/import"
targetNamespace="http://xml.avaya.com/schema/presence"
elementFormDefault="qualified"
xmlns:abc="http://xml.avaya.com/schema/import1">

    <xsd:import namespace="http://xml.avaya.com/schema/import"
schemaLocation="userimport.xsd"/>

    <xsd:complexType name="XmlPsCommProfile">

        <xsd:complexContent>

            <xsd:extension base="one:xmlCommProfileType" >

                <xsd:sequence>

                    <xsd:element name="system" type="xsd:string"
maxOccurs="1" />

                    <xsd:element name="imGatewaySipEntity"
type="xsd:string" maxOccurs="1" minOccurs="0"/>

                    <xsd:element name="publishViaAESCollector"
type="xsd:string" maxOccurs="1" />

                </xsd:sequence>

            </xsd:extension>

        </xsd:complexContent>

    </xsd:complexType>

</xsd:schema>
```

XML Schema Definition for deleting User(s)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<xs:schema xmlns:tns="http://xml.avaya.com/schema/bulkdelete"
targetNamespace="http://xml.avaya.com/schema/bulkdelete"

                 elementFormDefault="qualified" version="1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema" >
```

```
   <xs:element name="user" type="tns:xmlUserDelete" />
   <xs:element name="deleteType" type="tns:xmlDeleteType" />

   <xs:element name="deleteUsers">
      <xs:complexType>
            <xs:sequence>
                  <xs:element name="deleteType" type="tns:xmlDeleteType"
maxOccurs="1" minOccurs="1"/>
                  <xs:element minOccurs="1" maxOccurs="unbounded"
name="user" type="tns:xmlUserDelete" />
            </xs:sequence>
      </xs:complexType>
   </xs:element>



   <xs:complexType name="xmlUserDelete">
      <xs:sequence>
            <xs:element name="loginName" minOccurs="1" maxOccurs="1">
                  <xs:simpleType>
                        <xs:restriction base="xs:string">
                              <xs:maxLength value="128"></xs:maxLength>
                        </xs:restriction>
                  </xs:simpleType>
            </xs:element>
            <xs:element name="id" type="xs:string" maxOccurs="1"
minOccurs="0"></xs:element>
      </xs:sequence>
   </xs:complexType>

  <xs:simpleType name="xmlDeleteType">
      <xs:restriction base="xs:string"></xs:restriction>
  </xs:simpleType>
</xs:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Work Assignment

Note: The XSD is introduced in System Manager 7.0.

```
<?xml version="1.0" encoding="UTF-8" ?>
     <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                     xmlns:smgr="http://xml.avaya.com/schema/import"

     targetNamespace="http://xml.avaya.com/schema/import_workassignment"
                     elementFormDefault="qualified">

     <!-- This is the XML schema for the "Work Assignment Profile".  It
          defines this profile inside of an XML document that defines
          a user record (see userimport.xsd) -->

     <xsd:import namespace="http://xml.avaya.com/schema/import"
                 schemaLocation="userimport.xsd" />

     <xsd:complexType name="WorkAssignmentCommProfXML">
       <xsd:complexContent>
         <xsd:extension base="smgr:xmlCommProfileType" >
            <xsd:sequence>
             <xsd:element name="workAssignmentResourceDetails"
minOccurs="0" maxOccurs="unbounded">
               <xsd:complexType>
                 <xsd:sequence>
                   <xsd:element name="associatedHandleName"
type="xsd:string" minOccurs="1" maxOccurs="1" />
                       <xsd:element name="accountName" type="xsd:string"
minOccurs="1" maxOccurs="1" />
                       <xsd:element name="accountAddress"
type="xsd:string" minOccurs="1" maxOccurs="1" />
                       <xsd:element name="sourceName" type="xsd:string"
minOccurs="1" maxOccurs="1" />
                       <xsd:element name="sourceAddress" type="xsd:string"
minOccurs="0" maxOccurs="1" />
                 </xsd:sequence>
               </xsd:complexType>
             </xsd:element>
           </xsd:sequence>
         </xsd:extension>
       </xsd:complexContent>
     </xsd:complexType>
```

```
      </xsd:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Officelinx

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:smgr="http://xml.avaya.com/schema/import"
elementFormDefault="qualified"
      targetNamespace="http://xml.avaya.com/schema/import_mem_officelinx">


      <xs:import namespace="http://xml.avaya.com/schema/import"
            schemaLocation="userimport.xsd" />


      <xs:complexType name="xmlOfficelinxProfile">
            <xs:complexContent>
                  <xs:extension base="smgr:xmlCommProfileType">
                        <xs:sequence>
                              <xs:element type="xs:string"
name="officelinxName" maxOccurs="1" />
                              <xs:element type="xs:long"
name="mailBoxNumber" maxOccurs="1" />
                              <xs:element type="xs:string"
name="numericPassword" maxOccurs="1" minOccurs="0" />
                              <xs:element type="xs:string"
name="applicationUserPassword" maxOccurs="1" minOccurs="0" />
                              <xs:element type="xs:string" name="company"
maxOccurs="1" minOccurs="0" />
                              <xs:element type="xs:string" name="department"
maxOccurs="1" minOccurs="0" />
                              <xs:element type="xs:string"
name="featureGroup" maxOccurs="1" minOccurs="0" />
                              <xs:element type="xs:string" name="capability"
maxOccurs="1" minOccurs="0" />
```

```
                            <xs:element type="xs:string"
name="domainAccountName" maxOccurs="1" minOccurs="0" />
                            <xs:element type="xs:string"
name="synchronizationUserName" maxOccurs="1" minOccurs="0" />
                        </xs:sequence>
                 </xs:extension>
           </xs:complexContent>
     </xs:complexType>
</xs:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Equinox / Scopia

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:one="http://xml.avaya.com/schema/import"
targetNamespace="http://xml.avaya.com/schema/import_scopia"
elementFormDefault="qualified"
xmlns:abc="http://xml.avaya.com/schema/import_scopia">
<xsd:import namespace="http://xml.avaya.com/schema/import"
            schemaLocation="userimport.xsd"/>
<xsd:complexType name="ScopiaCommProfileType">
    <xsd:complexContent>
        <xsd:extension base="one:xmlCommProfileType" >
            <xsd:sequence>
                <xsd:element name="scopiaUserId" type="xsd:string"
minOccurs="0"/>
                <xsd:element name="password" type="xsd:string"/>
                <xsd:element name="vrNumber" type="xsd:string"
minOccurs="0"/>
                <xsd:element name="needVR" type="xsd:boolean"
minOccurs="0"/>
                <xsd:element name="virtualRoomId" type="xsd:string"
minOccurs="0"/>
            </xsd:sequence>
```

```
        </xsd:extension>
    </xsd:complexContent>
 </xsd:complexType>
</xsd:schema>
```

XML Schema Definition for retrieving, creating and updating and User(s): Messaging

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:one="http://xml.avaya.com/schema/import"
elementFormDefault="qualified"
      targetNamespace="http://xml.avaya.com/schema/import_csm_mm"
xmlns:csm="http://xml.avaya.com/schema/import_csm_mm">


      <xs:import namespace="http://xml.avaya.com/schema/import"
            schemaLocation="userimport.xsd" />
      <!--Changes in xsd file need to generate jaxb src using this xsd-->
      <xs:complexType name="xmlMessagingProfile">
            <xs:complexContent>
                  <xs:extension base="one:xmlCommProfileType">
                        <xs:sequence>
                              <!--
                                    Specifies the messaging system of the
subscriber you want to add.
                                    Name as it appears under
'Applications/Application
                                    Management/Entities
                              -->
                              <xs:element name="messagingName"
type="xs:string"
                                    maxOccurs="1" minOccurs="1" />
```

```
                              <xs:element name="msgprefHandleId"
type="xs:string"
                                    maxOccurs="1" minOccurs="0" />

                              <xs:element name="useExisting"
type="xs:boolean"
                                    maxOccurs="1" minOccurs="0" />

                              <!-- Specifies the messaging template of a
subscriber.  -->
                              <xs:element name="messagingTemplate"
type="xs:string"
                                    maxOccurs="1" minOccurs="0" />

                              <xs:element name="mailboxNumber" maxOccurs="1"
                                    minOccurs="1">
                                    <xs:simpleType>
                                          <xs:restriction base="xs:string">
                                                <xs:pattern value="[0-9]
{1,50}" />
                                          </xs:restriction>
                                    </xs:simpleType>
                              </xs:element>

                              <!--
                                    Specifies the default password the
subscriber must use to log in
                                    to his or her mailbox. The password can
be from one digit in
                                    length to a maximum of 15 digits.
                              -->
                              <xs:element name="password" maxOccurs="1"
minOccurs="1">
                                    <xs:simpleType>
                                          <xs:restriction base="xs:string">
                                                <xs:pattern value="[0-9]
```

```
{0,15}" />
                                        </xs:restriction>
                                  </xs:simpleType>
                            </xs:element>


                            <xs:element name="deleteOnUnassign"
type="xs:boolean"
                                  maxOccurs="1" minOccurs="0" />


                            <!-- follows overrriding  subscriber  data
-->


                            <!--
                                  The class of service for this
subscriber. The COS controls
                                  subscriber access to many features and
provides general settings,
                                  such as mailbox size.
                            -->
                            <xs:element name="cos" maxOccurs="1"
minOccurs="0">
                                  <xs:simpleType>
                                        <xs:restriction base="xs:string">
                                              <xs:pattern
                                                    value="[0-9]|[0-9]{2}|
[0-4][0-9]{2}|[5][0-4][0-9]|[5][5][0-1]" />
                                        </xs:restriction>
                                  </xs:simpleType>
                            </xs:element>


                            <!--
                                  Specifies the name that appears before
the machine name and domain
                                  in the subscriber's e-mail address. The
machine name and domain
                                  are automatically added to the handle
```

```
you enter when the
                                    subscriber sends or receives an e-mail.
                              -->
                              <xs:element name="emailHandle" maxOccurs="1"
minOccurs="0">
                                    <xs:simpleType>
                                          <xs:restriction base="xs:string">
                                                <xs:pattern value="[a-zA-Z0-
9_@\w\.\-]*" />
                                          </xs:restriction>
                                    </xs:simpleType>
                              </xs:element>

                              <!--
                                    Specifies the display name of the
subscriber in address book
                                    listings, such as those for e-mail
client applications. The name
                                    you enter can be 1 to 64 characters in
length.
                              -->
                              <xs:element name="commonName" type="xs:string"
                                    maxOccurs="1" minOccurs="0" />

                              <xs:element name="mmSpecific"
type="csm:xmlMMSpecific"
                                    maxOccurs="1" minOccurs="0" />
                              <xs:element name="cmmSpecific"
type="csm:xmlCMMSpecific"
                                    maxOccurs="1" minOccurs="0" />
                              <xs:element name="aamSpecific"
type="csm:xmlAAMSpecific"
                                    maxOccurs="1" minOccurs="0" />
                        </xs:sequence>
                  </xs:extension>
            </xs:complexContent>
```

```
      </xs:complexType>



      <xs:complexType name="xmlMMSpecific">
            <xs:sequence>


                  <!-- Specifies the default community ID for the
subscriber. Community IDs
                        are used to control message sending and receiving
among groups of subscribers.
                        The default value is 1. -->
                  <xs:element name="communityID" maxOccurs="1"
minOccurs="0">
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                                    <xs:pattern value="[0-9]|[0-1][0-5]" />
                              </xs:restriction>
                        </xs:simpleType>
                  </xs:element>


                  <!--
                        Specifies a unique address in the voice mail
network. The numeric
                        address can be from 1 to 50 digits and can contain
the Mailbox
                        Number.
                  -->
                  <xs:element name="numericAddress" maxOccurs="1"
minOccurs="0">
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                                    <xs:pattern value="([0-9])*" />
                              </xs:restriction>
                        </xs:simpleType>
                  </xs:element>
```

```
                <!-- The primary telephone extension of the subscriber.
-->
                <xs:element name="pbxExtension" maxOccurs="1"
minOccurs="0">
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
                                  <xs:pattern value="([+0-9])*" />
                            </xs:restriction>
                      </xs:simpleType>
                </xs:element>

                <!--
                      The telephone number of the subscriber as displayed
in address book
                      listings and client applications. The entry can be
a maximum of 50
                      characters in length and can contain any
combination of digits
                      (0-9), period (.), hyphen (-), plus sign (+), and
left and right
                      parentheses ([) and (]).
                -->
                <xs:element name="telephoneNumber" maxOccurs="1"
                      minOccurs="0"> <!-- MM field -->
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
                                  <xs:pattern value="([-+\.()0-9])*" />
                            </xs:restriction>
                      </xs:simpleType>
                </xs:element>

                <!--
                      If the subscriber name is entered in multi-byte
character format,
                      then this field specifies the ASCII translation of
```

```
the subscriber
                      name.
                 -->
                 <xs:element name="asciiVersionOfName" type="xs:string"
                      maxOccurs="1" minOccurs="0" />

                 <xs:element name="expirePassword" type="csm:xmlyesNoType"
                      maxOccurs="1" minOccurs="0" />

                 <!--
                      Specifies whether you want your mailbox to be
locked. A subscriber
                      mailbox can become locked after two unsuccessful
login attempts. You
                      can choose one of the following: - no: to unlock
your mailbox - yes:
                      to lock your mailbox and prevent access to it
                 -->
                 <xs:element name="mailBoxLocked" type="csm:xmlyesNoType"
                      maxOccurs="1" minOccurs="0" />

                 <!--
                      Specifies the mailbox number or transfer dial
string of the
                      subscriber's personal operator or assistant. This
field also
                      indicates the transfer target when a caller to this
subscriber
                      presses 0 while listening to the subscriber's
greeting.
                 -->
                 <xs:element name="personalOperatorMailbox" maxOccurs="1"
                      minOccurs="0">
                      <xs:simpleType>
                           <xs:restriction base="xs:string">
                                <xs:pattern value="[0-9]+([*#,][0-9]+)*"
```

```
/>
                              </xs:restriction>
                        </xs:simpleType>
                  </xs:element>

                  <!--
                        Specifies when to route calls to the backup
operator mailbox. The
                        default value for this field is Always Active.
                  -->
                  <xs:element name="personalOperatorSchedule"
type="xs:string"
                        maxOccurs="1" minOccurs="0" />

                  <xs:element name="tuiMessageOrderAdmin" maxOccurs="1"
                        minOccurs="0">
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                                    <xs:enumeration value="urgent first then
newest" />
                                    <xs:enumeration value="oldest messages
first" />
                                    <xs:enumeration value="newest messages
first" />
                                    <xs:enumeration value="urgent first then
oldest" />
                              </xs:restriction>
                        </xs:simpleType>
                  </xs:element>

                  <xs:element name="tuiMessageOrderDeleted" maxOccurs="1"
                        minOccurs="0">
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                                    <xs:enumeration value="urgent first then
newest" />
```

```
                                <xs:enumeration value="oldest messages
first" />
                                <xs:enumeration value="newest messages
first" />
                                <xs:enumeration value="urgent first then
oldest" />
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>

                    <xs:element name="tuiMessageOrderNew" maxOccurs="1"
                        minOccurs="0">
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                                <xs:enumeration value="urgent first then
newest" />
                                <xs:enumeration value="oldest messages
first" />
                                <xs:enumeration value="newest messages
first" />
                                <xs:enumeration value="urgent first then
oldest" />
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>

                    <xs:element name="tuiMessageOrderSaved" maxOccurs="1"
                        minOccurs="0"> <!-- MM field -->
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                                <xs:enumeration value="urgent first then
newest" />
                                <xs:enumeration value="oldest messages
first" />
                                <xs:enumeration value="newest messages
first" />
                                <xs:enumeration value="urgent first then
```

```
oldest" />
                            </xs:restriction>
                      </xs:simpleType>
                </xs:element>


                <!--
                      Specifies the intercom paging settings for a
subscriber. You can
                      choose one of the following: - paging is off: to
disable intercom
                      paging for this subscriber. - paging is manual: if
the subscriber
                      can modify, with Subscriber Options or the TUI, the
setting that
                      allows callers to page the subscriber. - paging is
automatic: if the
                      TUI automatically allows callers to page the
subscriber.
                -->
                <xs:element name="intercomPaging" maxOccurs="1"
minOccurs="0"> <!-- MM field -->
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
                                  <xs:enumeration value="paging is off" />
                                  <xs:enumeration value="paging is manual"
/>
                                  <xs:enumeration value="paging is
automatic" />
                            </xs:restriction>
                      </xs:simpleType>
                </xs:element>


                <!--
                      Specifies whether a subscriber can receive
messages, e-mail messages
                      and call-answer messages from other subscribers.
You can choose one
```

```
                      of the following: - yes: to allow the subscriber to
create, forward,
                      and receive messages. - no: to prevent the
subscriber from receiving
                      call-answer messages and to hide the subscriber
from the telephone
                      user interface (TUI). The subscriber cannot use the
TUI to access
                      the mailbox, and other TUI users cannot address
messages to the
                      subscriber.
                -->
                <xs:element name="voiceMailEnabled"
type="csm:xmlTrueFalseType"
                      maxOccurs="1" minOccurs="0" />


                <!--
                      Specifies additional, useful information about a
subscriber. Entries
                      in this field are for convenience and are not used
by the messaging
                      system.
                -->
                <xs:element name="miscellaneous1"
type="csm:xmlLength51Type"
                      maxOccurs="1" minOccurs="0" />


                <!--
                      Specifies additional, useful information about a
subscriber. Entries
                      in this field are for convenience and are not used
by the messaging
                      system.
                -->
                <xs:element name="miscellaneous2"
type="csm:xmlLength51Type"
                      maxOccurs="1" minOccurs="0" />
```

```
                <!--
                      Specifies additional, useful information about a
subscriber. Entries
                      in this field are for convenience and are not used
by the messaging
                      system.
                -->
                <xs:element name="miscellaneous3"
type="csm:xmlLength51Type"
                      maxOccurs="1" minOccurs="0" />


                <!--
                      Specifies additional, useful information about a
subscriber. Entries
                      in this field are for convenience and are not used
by the messaging
                      system.
                -->
                <xs:element name="miscellaneous4"
type="csm:xmlLength51Type"
                      maxOccurs="1" minOccurs="0" />


                <!-- Specifies one or more alternate number to reach a
subscriber. You can
                      use secondary extensions to specify a telephone
number for direct reception
                      of faxes, to allow callers to use an existing
Caller Application, or to identify
                      each line appearance on the subscriber's telephone
set if they have different
                      telephone numbers. -->
                <xs:element name="secondaryExtension" maxOccurs="1"
                      minOccurs="0"> <!-- MM/CMM field -->
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
```

```
                                    <xs:pattern value="[0-9,]{0,50}" />
                              </xs:restriction>
                        </xs:simpleType>
                  </xs:element>
            </xs:sequence>
      </xs:complexType>

      <xs:complexType name="xmlCMMSpecific">
            <xs:sequence>

                  <!--
                        Specifies the number of the switch on which this
subscriber's
                        extension is administered. You can enter "0"
through "99", or leave
                        this field blank. - Leave this field blank if the
host switch number
                        should be used. - Enter a "0" if no message waiting
indicators
                        should be sent for this subscriber. You should
enter 0 when the
                        subscriber does not have a phone on any switch in
the network.
                  -->
                  <xs:element name="switchNumber" maxOccurs="1"
minOccurs="0">
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                                    <xs:pattern value="[0-9]|[0-9][0-9]" />
                              </xs:restriction>
                        </xs:simpleType>
                  </xs:element>

                  <!-- Specifies the default community ID for the
subscriber. Community IDs
                        are used to control message sending and receiving
```

```
among groups of subscribers.
                      The default value is 1. -->
                 <xs:element name="communityID" maxOccurs="1"
minOccurs="0"> <!-- MM/CMM field -->
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
                                  <xs:pattern value="[0-9]|[0-1][0-5]" />
                            </xs:restriction>
                      </xs:simpleType>
                 </xs:element>


                 <!--
                      Specifies the Subscriber Account Code. The
Subscriber Account Code
                      is used to create Call Detail Records on the switch
for calls placed
                      by the voice ports. The value you enter in this
field can contain
                      any combination of digits from 0 to 9. If an
account code is not
                      specified, the system will use the subscriber's
mailbox extension as
                      the account code.
                 -->
                 <xs:element name="accountCode" maxOccurs="1"
minOccurs="0"> <!-- CMM field -->
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
                                  <xs:pattern value="([0-9])*" />
                            </xs:restriction>
                      </xs:simpleType>
                 </xs:element>

                 <xs:element name="mwiEnabled" type="csm:xmlyesNoType"
                      maxOccurs="1" minOccurs="0" />
```

```
                <!--
                      Specifies the number to be used as the default
destination for the
                      Transfer Out of Messaging feature. You can enter 3
to 50 digits in
                      this field depending on the length of the system's
extension, or
                      leave this field blank.
                -->
                <xs:element name="coveringExtension" maxOccurs="1"
                      minOccurs="0"> <!-- CMM field -->
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
                                  <xs:pattern value="[0-9]{0}|[0-9]{3,50}"
/>
                            </xs:restriction>
                      </xs:simpleType>
                </xs:element>

                <!--
                      Specifies additional, useful information about a
subscriber. Entries
                      in this field are for convenience and are not used
by the messaging
                      system.
                -->
                <xs:element name="miscellaneous1"
type="csm:xmlLength11Type"
                      maxOccurs="1" minOccurs="0" />

                <!--
                      Specifies additional, useful information about a
subscriber. Entries
                      in this field are for convenience and are not used
by the messaging
```

```
                    system.
                -->
                <xs:element name="miscellaneous2"
type="csm:xmlLength11Type"
                    maxOccurs="1" minOccurs="0" />


                <!--
                    Specifies additional, useful information about a
subscriber. Entries
                    in this field are for convenience and are not used
by the messaging
                    system.
                -->
                <xs:element name="miscellaneous3"
type="csm:xmlLength11Type"
                    maxOccurs="1" minOccurs="0" />


                <!--
                    Specifies additional, useful information about a
subscriber. Entries
                    in this field are for convenience and are not used
by the messaging
                    system.
                -->
                <xs:element name="miscellaneous4"
type="csm:xmlLength11Type"
                    maxOccurs="1" minOccurs="0" />


                <!-- Specifies alternate number to reach a subscriber.
You can
                    use secondary extension to specify a telephone
number for direct reception
                    of faxes, to allow callers to use an existing
Caller Application, or to identify
                    each line appearance on the subscriber's telephone
set if they have different
                    telephone numbers. Valid Entries are blank or 3-50
```

```
digits (0-9), depending on
                      the length of the system's extension.
                 -->
                 <xs:element name="secondaryExtension" maxOccurs="1"
                      minOccurs="0"> <!-- MM/CMM field -->
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
                                  <xs:pattern value="[0-9]{0}|[0-9]{3,50}"
/>
                            </xs:restriction>
                      </xs:simpleType>
                 </xs:element>


                 <!--
                      If the subscriber name is entered in multi-byte
character format,
                      then this field specifies the ASCII translation of
the subscriber
                      name. wi01032730
                 -->
                 <xs:element name="asciiVersionOfName" type="xs:string"
                      maxOccurs="1" minOccurs="0" />


                 <!--
                      Specifies an email address to be used as the
Forwarding Destination Address
                      where messages should be forwarded to when an
incoming message is received.
                      This field can include alphanumeric characters,
periods (.), hyphens (-),
                      underscores (_), pluses (+), colons (:) and should
also include the commercial at (@).
                 -->
                 <xs:element name="frwdDestAddress" maxOccurs="1"
                      minOccurs="0"> <!-- CMM field -->
```

```
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                                    <xs:pattern value="[a-zA-Z0-9_\.\+-:@a-
zA-Z0-9_\.\+-:]*" />
                              </xs:restriction>
                        </xs:simpleType>
                  </xs:element>

                  <!--
                        Specifies that an incoming message should be
deleted once it is forwarded on to the Forwarding Destination Address.
                  -->
                  <xs:element name="deleteAfterForward" type="xs:boolean"
                        maxOccurs="1" minOccurs="0" />
            </xs:sequence>
      </xs:complexType>

      <xs:complexType name="xmlAAMSpecific">
            <xs:sequence>

                  <!-- The Id of Site -->
                  <xs:element name="siteId" maxOccurs="1" minOccurs="0">
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                                    <xs:pattern value="([0-9])*" />
                              </xs:restriction>
                        </xs:simpleType>
                  </xs:element>

                  <!--
                        Specifies a unique address in the voice mail
network. The numeric
                        address can be from 1 to 50 digits and can contain
the Mailbox
                        Number.
```

```
                -->
                <xs:element name="numericAddress" maxOccurs="1"
minOccurs="0">
                     <xs:simpleType>
                          <xs:restriction base="xs:string">
                               <xs:pattern value="([0-9])*" />
                          </xs:restriction>
                     </xs:simpleType>
                </xs:element>

                <!-- The primary telephone extension of the subscriber.
-->
                <xs:element name="pbxExtension" maxOccurs="1"
minOccurs="0">
                     <xs:simpleType>
                          <xs:restriction base="xs:string">
                               <xs:pattern value="([+0-9])*" />
                          </xs:restriction>
                     </xs:simpleType>
                </xs:element>

                <xs:element name="storageDestination"
type="csm:xmlMssExchangeType"
                     maxOccurs="1" minOccurs="0" />

                <xs:element name="storageDestinationAccountId"
type="xs:string"
                     maxOccurs="1" minOccurs="0" />

                <xs:element name="storageDestinationServer"
type="xs:string"
                     maxOccurs="1" minOccurs="0" />

                <!--
                     The telephone number of the subscriber as displayed
```

```
in address book
                      listings and client applications. The entry can be
a maximum of 50
                      characters in length and can contain any
combination of digits
                      (0-9), period (.), hyphen (-), plus sign (+), and
left and right
                      parentheses ([) and (]).
                -->
                <xs:element name="telephoneNumber" maxOccurs="1"
                      minOccurs="0"> <!-- MM field -->
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
                                  <xs:pattern value="([-+\.()0-9])*" />
                            </xs:restriction>
                      </xs:simpleType>
                </xs:element>

                <xs:element name="asciiVersionOfName" type="xs:string"
                      maxOccurs="1" minOccurs="0" />

                <xs:element name="pronounceableName" type="xs:string"
                      maxOccurs="1" minOccurs="0" />

                <xs:element name="includedInAADirectory"
type="xs:boolean"
                                  maxOccurs="1" minOccurs="0" />

                <xs:element name="expirePassword" type="csm:xmlyesNoType"
                      maxOccurs="1" minOccurs="0" />

                <!--
                      Specifies whether you want your mailbox to be
locked. A subscriber
                      mailbox can become locked after two unsuccessful
```

```
login attempts. You
                      can choose one of the following: - no: to unlock
your mailbox - yes:
                      to lock your mailbox and prevent access to it
                 -->
                 <xs:element name="mailBoxLocked" type="csm:xmlyesNoType"
                      maxOccurs="1" minOccurs="0" />

                 <xs:element name="mustChangePassword" type="xs:boolean"
                                 maxOccurs="1" minOccurs="0" />

                 <!--
                      Specifies subscriber's personal attendant.
                 -->
                 <xs:element name="personalAttendant" maxOccurs="1"
                      minOccurs="0"> <!-- MM field -->
                      <xs:simpleType>
                           <xs:restriction base="xs:string">
                                <xs:pattern value="([-+()0-9])*" />
                           </xs:restriction>
                      </xs:simpleType>
                 </xs:element>

                 <!--
                      Specifies the intercom paging settings for a
subscriber. You can
                      choose one of the following: - paging is off: to
disable intercom
                      paging for this subscriber. - paging is manual: if
the subscriber
                      can modify, with Subscriber Options or the TUI, the
setting that
                      allows callers to page the subscriber. - paging is
automatic: if the
                      TUI automatically allows callers to page the
```

```
subscriber.
                -->
                <xs:element name="intercomPaging" maxOccurs="1"
minOccurs="0"> <!-- MM field -->
                      <xs:simpleType>
                            <xs:restriction base="xs:string">
                                  <xs:enumeration value="paging is off" />
                                  <xs:enumeration value="paging is manual"
/>
                                  <xs:enumeration value="paging is
automatic" />
                            </xs:restriction>
                      </xs:simpleType>
                </xs:element>


                <!--
                      Specifies the order in which the subscriber hears
the voice
                      messages. You can choose one of the following: -
urgent first then
                      newest: to direct the system to play any messages
marked as urgent
                      prior to playing non-urgent messages. Both the
urgent and non-urgent
                      messages are played in the reverse order of how
they were received.
                      - oldest messages first: to direct the system to
play messages in
                      the order they were received. - urgent first then
oldest: to direct
                      the system to play any messages marked as urgent
prior to playing
                      non-urgent messages. Both the urgent and non-urgent
messages are
                      played in the order of how they were received. -
newest messages
                      first: to direct the system to play messages in the
reverse order of
```

```
                        how they were received.


                        Note: tuiMessageOrder field is no longer used since
SMGR_7.1.2_S4.
                        You should use playback order related fields for
Unread, Read and
                        Saved messages instead of tuiMessageOrder:
                        tuiMessageOrderUnread, playUnreadUrgentFirst,
arrangeBySenderUnread,
                        tuiMessageOrderRead, playReadUrgentFirst,
arrangeBySenderRead,
                        tuiMessageOrderSaved, playSavedUrgentFirst,
arrangeBySenderSaved.
                  -->
                  <xs:element name="tuiMessageOrder" maxOccurs="1"
                        minOccurs="0"> <!-- MM field -->
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                                    <xs:enumeration value="urgent first then
newest" />
                                    <xs:enumeration value="oldest messages
first" />
                                    <xs:enumeration value="newest messages
first" />
                                    <xs:enumeration value="urgent first then
oldest" />
                              </xs:restriction>
                        </xs:simpleType>
                  </xs:element>

                  <xs:element name="mwiEnabled" type="csm:xmlMwiType"
                        maxOccurs="1" minOccurs="0" />


                  <!--
                        Specifies whether a subscriber can receive
messages, e-mail messages
                        and call-answer messages from other subscribers.
```

```
You can choose one
                      of the following: - yes: to allow the subscriber to
create, forward,
                      and receive messages. - no: to prevent the
subscriber from receiving
                      call-answer messages and to hide the subscriber
from the telephone
                      user interface (TUI). The subscriber cannot use the
TUI to access
                      the mailbox, and other TUI users cannot address
messages to the
                      subscriber.
                -->
                <xs:element name="voiceMailEnabled"
type="csm:xmlTrueFalseType"
                      maxOccurs="1" minOccurs="0" />


                <!--
                      Specifies additional, useful information about a
subscriber. Entries
                      in this field are for convenience and are not used
by the messaging
                      system.
                -->
                <xs:element name="miscellaneous1"
type="csm:xmlLength51Type"
                      maxOccurs="1" minOccurs="0" />


                <!--
                      Specifies additional, useful information about a
subscriber. Entries
                      in this field are for convenience and are not used
by the messaging
                      system.
                -->
                <xs:element name="miscellaneous2"
type="csm:xmlLength51Type"
```

```
                        maxOccurs="1" minOccurs="0" />


                  <!--
                        Specifies additional, useful information about a
subscriber. Entries
                        in this field are for convenience and are not used
by the messaging
                        system.
                  -->
                  <xs:element name="miscellaneous3"
type="csm:xmlLength51Type"
                        maxOccurs="1" minOccurs="0" />


                  <!--
                        Specifies additional, useful information about a
subscriber. Entries
                        in this field are for convenience and are not used
by the messaging
                        system.
                  -->
                  <xs:element name="miscellaneous4"
type="csm:xmlLength51Type"
                        maxOccurs="1" minOccurs="0" />


                  <!-- Specifies one or more alternate number to reach a
subscriber. You can
                        use secondary extensions to specify a telephone
number for direct reception
                        of faxes, to allow callers to use an existing
Caller Application, or to identify
                        each line appearance on the subscriber's telephone
set if they have different
                        telephone numbers. -->
                  <xs:element name="secondaryExtension" maxOccurs="1"
                        minOccurs="0"> <!-- MM/CMM field -->
                        <xs:simpleType>
```

```
                        <xs:restriction base="xs:string">
                              <xs:pattern value="([0-9]{3,50},){0,6}
([0-9]{3,50})?" />
                        </xs:restriction>
                  </xs:simpleType>
            </xs:element>
            <xs:element name="timezone" type="xs:string"
                  maxOccurs="1" minOccurs="0" />


            <!--
                  The options to configure your own message playback
order. The message playback order options depend on the following message
types:
                  Unread messages: fields below define the order for
unread messages in Aria or AUDIX TUI case and for all messages in
CallPilot TUI case.
                        TUI Message Order:
                              Play newest first : The TUI plays the
unread messages starting with the newest message first.
                              Play oldest first: The TUI plays the
unread messages starting with the oldest message first.
                              Default value: The TUI plays the unread
messages in order defined by selected Class Of Service (i.e. Play newest
first or Play oldest first). This value is only for AAM 7.0 and above.
                        Play important messages before others: The TUI
plays the unread urgent messages first. For AAM 6.3 and below the options
are: Yes, No. For AAM 7.0 and above the options are:  Yes, No, Default
value.
                        Arrange by sender: The TUI plays the unread
messages that Messaging sorts by sender name. For AAM 6.3 and below the
options are: Yes, No. For AAM 7.0 and above the options are:  Yes, No,
Default value.
                  Read messages: fields below define the order for
read messages in Aria or AUDIX TUI case.
                        TUI Message Order:
                              Play newest first : The TUI plays the
read messages starting with the newest message first.
                              Play oldest first: The TUI plays the
read messages starting with the oldest message first.
```

```
                                    Default value: The TUI plays the read
messages in order defined by selected Class Of Service (i.e. Play newest
first or Play oldest first). This value is only for AAM 7.0 and above.

                              Play important messages before others: The TUI
plays the read urgent messages first. For AAM 6.3 and below the options
are: Yes, No. For AAM 7.0 and above the options are:  Yes, No, Default
value.

                              Arrange by sender: The TUI plays the read
messages that Messaging sorts by sender name. For AAM 6.3 and below the
options are: Yes, No. For AAM 7.0 and above the options are:  Yes, No,
Default value.

                        Saved messages: fields below define the order for
saved messages in Aria TUI case.

                              TUI Message Order:

                                    Play newest first : The TUI plays the
saved messages starting with the newest message first.

                                    Play oldest first: The TUI plays the
saved messages starting with the oldest message first.

                                    Default value: The TUI plays the saved
messages in order defined by selected Class Of Service (i.e. Play newest
first or Play oldest first). This value is only for AAM 7.0 and above.

                              Play important messages before others: The TUI
plays the saved urgent messages first. For AAM 6.3 and below the options
are: Yes, No. For AAM 7.0 and above the options are:  Yes, No, Default
value.

                              Arrange by sender: The TUI plays the saved
messages that Messaging sorts by sender name. For AAM 6.3 and below the
options are: Yes, No. For AAM 7.0 and above the options are:  Yes, No,
Default value.

                 -->

                 <!-- For unread messages -->

                 <xs:element name="tuiMessageOrderUnread"
type="csm:xmlAAMTuiOrderType"

                       maxOccurs="1" minOccurs="0" />


                 <xs:element name="playUnreadUrgentFirst"
type="csm:xmlyesNoDefaultType"

                       maxOccurs="1" minOccurs="0" />


                 <xs:element name="arrangeBySenderUnread"
type="csm:xmlyesNoDefaultType"
```

```
                    maxOccurs="1" minOccurs="0" />


                <!-- For read messages -->
                <xs:element name="tuiMessageOrderRead"
type="csm:xmlAAMTuiOrderType"
                    maxOccurs="1" minOccurs="0" />


                <xs:element name="playReadUrgentFirst"
type="csm:xmlyesNoDefaultType"
                    maxOccurs="1" minOccurs="0" />


                <xs:element name="arrangeBySenderRead"
type="csm:xmlyesNoDefaultType"
                    maxOccurs="1" minOccurs="0" />


                <!-- For saved messages -->
                <xs:element name="tuiMessageOrderSaved"
type="csm:xmlAAMTuiOrderType"
                    maxOccurs="1" minOccurs="0" />


                <xs:element name="playSavedUrgentFirst"
type="csm:xmlyesNoDefaultType"
                    maxOccurs="1" minOccurs="0" />


                <xs:element name="arrangeBySenderSaved"
type="csm:xmlyesNoDefaultType"
                    maxOccurs="1" minOccurs="0" />


            </xs:sequence>
        </xs:complexType>


        <xs:simpleType name="xmlyesNoType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="Yes" />
                <xs:enumeration value="No" />
```

```
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="xmlyesNoDefaultType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Yes" />
            <xs:enumeration value="No" />
            <xs:enumeration value="Default value" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="xmlTrueFalseType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="TRUE" />
            <xs:enumeration value="FALSE" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="xmlLength11Type">
        <xs:restriction base="xs:string">
            <xs:maxLength value="11" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="xmlLength51Type">
        <xs:restriction base="xs:string">
            <xs:maxLength value="51" />
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="xmlMssExchangeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Avaya Message Store" />
```

```
                <xs:enumeration value="Microsoft Exchange" />
            </xs:restriction>
        </xs:simpleType>

        <xs:simpleType name="xmlMwiType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="No" />
                <xs:enumeration value="ByCOS" />
            </xs:restriction>
        </xs:simpleType>

        <xs:simpleType name="xmlAAMTuiOrderType">
            <xs:restriction base="xs:string">
                <xs:enumeration value="Play newest first" />
                <xs:enumeration value="Play oldest first" />
                <xs:enumeration value="Default value" />
            </xs:restriction>
        </xs:simpleType>
</xs:schema>
```

Appendix G – XSD attributes description

XSD attribute Definition for retrieving, creating and updating and User(s): Basic Attributes

Attribute	Attribute Description	Mandatory / Optional	Validation Constraints
authenticationType	This defines the type of authentication the user undergoes at runtime to gain access to the system. Possible values are basic or enterprise. Although this attribute is mandatory, the value entered is ignored by System Manager. It sets the value	Mandatory	Possible values: BASIC;ENTERPRISE

description	depending on how the user is provisioned in the system. On user addition via web service, it will always be set as "basic". On user update via web service, the current value in System Manager will not be updated. This is a text description of the user; a human readable description of this user instance.	Optional	
displayName	This is the localized name of the user to be used when displaying. Typically, the value is the localized full name. This value might be provisioned from the enterprise directory entry of the user. If the value does not exist, you can use synchronization rules to populate the value for other fields. For example: Surname, GivenName, or LoginName.	Optional	
displayNameAscii	This corresponds to the console attribute Endpoint Display Name. This is the full text name of the user represented in ASCII. It is used to support display (e.g. endpoints) that cannot handle localized text.	Optional	
Dn	This is the distinguished name (DN) of the user. DN is a sequence of relative distinguished names (RDN) connected by commas. RDN is an attribute with an associated value in the form of attribute=value, typically expressed in a UTF-8 string format. DN can be used to identify the user and can be used for authentication subject mapping. Note that DN is changeable.	Optional	
isDuplicatedLoginAllowed	This is a boolean indicator showing whether this user is allowed a duplicate concurrent logins. A true stipulates that the user is allow to have duplicate logins.	Optional	Default value is true.

isEnabled	This is a boolean indicator showing whether or not the user is active. Users with AuthenticationType=Basic fails if the value is false. This attribute can be used to disable access between login attempts. You cannot revoke login for a running session. Alternatively, the administrator can always modify the password to disable the user from logging in. A true stipulates this is an active user, a false used for a disabled user.	Optional	Default value is false.
isVirtualUser	A boolean indicator showing whether or not the record is being used for a non-human entity such as an application, service, software agent, etc. You use this attribute where the entity behaves as a user and needs to have subset of the user profile populated. If the entity does not behave as a user and has a different trust relationship, for example, a trust certificate must not be treated as a virtual user. A virtual user can represent an Avaya or an external non-human entity. This attribute is provided as a convenience to track such accounts. A true stipulates this is a virtual users, a false is used for human users.	Optional	Default value is false.
givenName	This is the first name of the user.	Mandatory	
honorific	This is the personal title used to address a user. This is typically a social title and not the work title which is contained in the title attribute. This attribute can map to "PersonalTitle".	Optional	
loginName	This is the unique system login name given to the user. It can take the form of username@domain or just username. This might vary across customers. It can be used to help provision default user handles in the CSHandle table. The username is an alphanumeric value that must comply with the userinfo related portion of a URI as described in rfc2396. However, it is further restricted as ASCII characters with only the "_" and "." special characters supported. This is the rfc2798 "uid" attribute.	Mandatory	
middleName	This is the middle name of the user.	Optional	

managerName	This is the text name of the user's manager. This is a free formed field and does not require the user's manager to also be a user of the solution. This attribute was requested to support reporting needs.	Optional	
preferredGivenName	This is the preferred first name of the user.	Optional	
preferredLanguage	This is the individual's preferred written or spoken language. Values conforms to rfc4646. Refer to rfc4646 for syntax. This format uses the ISO standard Language (ISO-639) and region (ISO-3166) codes In the absence of a value the locale of the client must be used, if no value is set, en_US must be used as default.	Optional	Possible values: English (United States) - en_US Chinese (Simplified) - zh_CN Japanese (Japan) - ja_JP Korean (Korea) - ko_KR French (France) - fr_FR German (Germany) - de_DE Italian (Italy) - it_IT Russian (Russia) - ru_RU English (United Kingdom) - en_GB Spanish (Mexico) - es_MX Portugese (Brazil) - pt_BR French (Canada) - fr_CA English (Canada) - en_CA

source	Free format text field that identifies the entity that created this user record. The format of this field must be a IP Address/Port or a name representing an enterprise LDAP or Avaya.	Optional	User Management populates the source field with the name of the file.
sourceUserKey	This is the key of the user from the source system. If the source is an Enterprise Active Directory server, this value with be the objectGUID.	Optional	By default, the value is none.
Status	This information is to help manage provisioning activities such as correcting or completing the provisioning of a user instance. It can also signify that approval is needed (PENDINGAUTHZ) before a user account is sufficiently configured to be a valid user (PROVISIONED).	Optional	Possible values: AUTHPENDING; PENDINGAUTHZ ; PROVISIONED
Suffix	This is the text appended to a name e.g. Jr., III.	Optional	
Surname	This is the user's last name, also called the family name.	Mandatory	

timeZone	This is the preferred time zone of the user. For example: America/New_York, Europe/Dublin. The application consuming this information must know how to translate e.g. in Java it is TimeZone.getTimeZone("Europe/Moscow"); In the absence of a value, the system uses the local services timezone. Note: You must consider daylight saving time (DST) and summer time adjustments while using the suggested values for timeZone. Typically, you add 1 hour to the offset. Note: You cannot use the following characters as is in the xml. Make the following modifications while using them in the import xml files: less-than character (<) as < < ampersand character (&) as & greater-than character (>) as > double-quote character (") as " apostrophe or single-quote character (') as '	Optional	Possible values: (-12:0)International Date Line West (-11:0)Midway Island, Samoa (-10:0)Hawaii (-9:0)Alaska (-8:0)Pacific Time (US & Canada); Tijuana (-7:0)Mountain Time (US & Canada); Chihuahua, La Paz (-7:0)Arizona (-6:0)Central Time (US & Canada); Guadalajara, Mexico City (-6:0)Central America; Saskatchewan (-5:0)Indiana (East); Bogota, Lima, Quito (-5:0)Eastern Time (US & Canada) (-4:0)Caracas, La Paz (-4:0)Atlantic Time (Canada); Santiago, Manaus (-3:30)Newfoundland (-3:0)Georgetown (-3:0)Brasilia, Greenland, Buenos Aires, Montevideo (-2:0)Mid-Atlantic (-1:0)Azores (-1:0)Cape Verde Is. (0:0)Monrovia,

			Reykjavik (0:0)GMT : Dublin, Edinburgh, Lisbon, London, Casablanca (+1:0)West Central Africa (+1:0)Amsterdam , Berlin, Rome, Belgrade, Prague, Brussels, Sarajevo (+2:0)Harare, Pretoria (+2:0)Amman, Athens, Minsk, Beirut, Cairo, Jerusalem, Helsinki, Windhoek (+3:0)Baghdad, Kuwait, Riyadh, Nairobi, Tbilisi (+3:0)Moscow, St. Petersburg, Volgograd (+3:30)Tehran (+4:0)Abu Dhabi, Muscat, Caucasus Standard Time (+4:0)Baku, Tbilisi, Yerevan (+4:30)Kabul (+5:0)Islamabad, Karachi, Tashkent, Ekaterinburg (+5:30)Chennai, Kolkata, Mumbai, New Delhi, Sri Jayawardenepura

			(+5:45)Kathmandu (+6:0)Astana, Dhaka, Almaty, Novosibirsk (+6:30)Rangoon (+7:0)Bangkok, Hanoi, Jakarta, Krasnoyarsk (+8:0)Beijing, Hong Kong, Singapore; Taipei (+8:0)Perth; Irkutsk, Ulaan Bataar (+9:0)Seoul, Osaka, Sapporo, Tokyo (+9:0)Yakutsk (+9:30)Darwin, Adelaide (+10:0)Brisbane, Guam, Port Moresby (+10:0)Canberra, Melbourne, Sydney, Hobart, Vladivostok (+11:0)Magadan, Solomon Is., New Caledonia (+12:0)Auckland, Wellington (+12:0)Fiji, Kamchatka, Marshall Is. (+13:0)Nuku'alofa

title	This is the job function of a person in their organizational context.	Optional	
userName	This is the username portion of the loginName field. It is an alphanumeric value that must comply with the userinfo related portion of a URI as described in rfc2396. However, it is further restricted as ASCII characters with only the "_" and "." special characters supported. This is the rfc2798 "uid" attribute.	Mandatory	
userPassword	This is the encrypted password for this user account. A null password is used when the user is authenticated by the enterprise such as with a separate source such as the enterprise LDAP.	Optional	Need not specified value for Enterprise User. If the value is not specified for the Basic user, the user will be disabled.
commPassword	This is the encrypted “subscriber” or communication password with which the user logs can use to authentication with on to any CommProfile SIP and non SIP. This attribute is shared across different communication profiles and thus different communication services.	Optional, if no SIP handle is defined.	
userType	This enumerates the possible primary user application types. A User can be associated with multiple user types.	Optional	Possible values are administrator, communication_user, agent, supervisor, resident_expert, service_technician, lobby_phone
roles	This is the text name of a role. This value must pre-exist in SMGR DB.	Optional	
Address	This is the address of the user.	Optional	
securityIdentity	This is the SecurityIdentity is used to hold any additional identities for a user that can be used for authentication such as loginName, Kerberos account name, or X509 certificate name.	Optional	

ownedContactLists	It is a collection of internal or external contacts. ContactList is owned by a specific user and has a name that a unique name within the context of its owner.	Optional	The system creates a default contactlist per user.
ownedContacts	It represents a non Avaya application user (external) contact. Contacts can be collected together along with User entities into a contact list. Contacts can be created by an administrator or an end user.	Optional	
presenceUserACL	These are personal rules defined by presentities themselves on who can monitor their presence information. There might be several entries in the list for a given presentity, each entry corresponding to one watcher.	Optional	
commProfileSet	A user has a default Commprofile set. A commprofile set can exist without any handles or commprofiles referencing it. That is, you can create a commprofile set without creating a handle or a commprofile. A commprofile set can contain multiple commprofiles, but only one of each specific type. This is enforced by having the CommProfile uniqueness constraint include type, commprofile_set_id.	Optional	A user has a default commprofile set.
commProfileList	List of communication profile	Optional	
commProfile	A communication profile is an entity that supportscommunication interactions established through Avaya Communication Services. A communication profile is used to represent a user's subscription to a product specific communication subsystem and contains its specific configuration needs for the user.	Optional	
handleList	List of handles	Optional	
handle	A user's address of record (AOR) is represented by a combination of a handle (userpart) and domain (domainpart). The entity that contains the userinfo part of an address that can be used to establish an interaction with a user. A user can have multiple handles.	Optional	
HandleName	This is the name given to the user to allow communication to be established with the user. It is an alphanumeric value that must	Optional	

	comply with the userinfo related portion of a URI as described in rfc2396. However, it is further restricted as ASCII characters with only the prefix to signify this is an E.164 handle and "_" and "." special characters supported.Note, the handle plus domain can be used to construct a user's Address of Record.		
handleType	The value reflecting the type of handle this is. Possible values are sip, smtp, ibm, and xmpp.	Optional	
handleSubType	This is an additional qualify on the handle type to help specify which private subsystem this handle belongs to.Possible values are e164,username,msrtc,googletalk,jabber, ibmsametime,lotousnotes,msexchage.	Optional	
domainName	The text name of the domain.	Optional	
employeeNo	Employee number of the user.	Optional	
department	Department which the employee belongs to.	Optional	
organization	Organization which the employee belongs to.	Optional	
localizedNames	Localized name of the user.	Optional	
NEW ATTRIBUTES INTRODUCED IN XSD IN SYSTEM MANAGER 6.3.4			
FOR SUPPORTING MULTI_TENANCY			
Tenant	Name of the tenant the user belongs	Optional	
organizationUnitLevelOne	Name of the organizationUnitLevelOne.For e.g. Site name	Optional	
organizationUnitLevelTwo	Name of the organizationUnitLevelTwo.For e.g. Department name	Optional	
organizationUnitLevelThree	Name of the organizationUnitLevelThree.For e.g. Team name	Optional	
createTenantIfNotAlreadyPresent	If the flag is set as true, the Tenant will be created if the same does not already existing in the system	Optional	Possible values: True/False
FOR SUPPORTING USER PROVISIONING RULE			
UserProvisionRuleName	Specify the rule name.This will facilitate administrators while provisioning users. When user will be created using this rule, most of the user attributes will get populated	Optional	

	based on the rule and minimal user attribute value will be required to be entered.		
FOR SUPPORTING LATIN TRANSLATION OF FIRST NAME AND SURNAME			
givenNameAscii	Displays the Latin translation of the first name. The user preferred first name that must be displayed on end points. Typically, the name is the written or spoken language of the user.	Optional	Auto-populated
surnameAscii	Displays the Latin translation of the Surname. The user preferred last name that must be displayed on end points. Typically, the name is the written or spoken language of the user.	Optional	Auto-populated
NEW ATTRIBUTES INTRODUCED IN XSD IN SYSTEM MANAGER 6.3.8			
newLoginName	New login name of the user	Optional	

Note: **For givenNameAscii and surnameAscii Attributes -**
When you enter the first name and the last name of the user associated with an endpoint, the Latin translation of the first name and the last name is auto populated in the Name field.The Auto Transliteration flag should be set to true in System Manager Configuration page.

XSD attribute definition for deleting User(s)

Attribute	Attribute description	Mandatory /Optional	Validation constraints
deleteType	Defines the delete type of the user. If the user selects “soft”, the system does not delete the user record permanently. You can recover the user record. If the user selects “delete”, the system permanently deletes all attributes associated with the user and the links to public contacts and shared addresses.	Mandatory	Possible values: soft,permanent
loginName	A unique system login name assigned to the user in the format of username@domain or username.	Mandatory	
Id	A unique identifier for a user record. The id attribute is included in the XSD for future enhancement.	Optional	

XSD attribute definition for retrieving, creating and updating and User(s): Messaging

Attribute	Attribute Description	Mandatory/ Optional	Validation Constraints
Messaging System Name messagingName	Name of Messaging System	Mandatory	
Messaging Pref Handle ID msgprefHandleId		Optional	
Use Existing Mailbox number useExisting	'true' if already created mailbox number is to be used. 'false' if available mailbox number is to be used.	Optional	
Messaging Template messagingTemplate	Specifies the messaging template of a subscriber.	Optional	
Mailbox Number mailboxNumber		Mandatory	The Mailbox number can be from one digit in length to a maximum of 50 digits.
Password	Specifies the default password the subscriber must use to log in to his or her mailbox.	Mandatory	The password can be from one digit in length to a maximum of 15 digits.
deleteOnUnassign		Optional	

Class of service	The class of service for this subscriber. The COS controls subscriber access to many features and provides general settings, such as mailbox size.	Optional	Valid ranges from 0 to 995
Community ID	Specifies the default community ID for the subscriber. Community IDs are used to control message sending and receiving among groups of subscribers.	Optional	The default value is 1.
Email Handle	Specifies the name that appears before the machine name and domain in the subscriber's e-mail address. The machine name and domain are automatically added to the handle you enter when the subscriber sends or receives an e-mail.	Optional	
Common Name	Specifies the display name of the subscriber in address book listings, such as those for e-mail client applications.	Optional	The name you enter can be 1 to 64 characters in length.
secondaryExtension	Specifies one or more alternate number to reach a subscriber. You can use secondary extensions to specify a telephone number for direct reception of faxes, to allow callers to use an existing Caller Application, or to identify each line appearance on the subscriber's telephone set if they have different telephone numbers.	Optional	Valid values 0 to 9 number values of length 10
mmSpecific	This is complex type for Modular Messaging specific fields data.	Optional	
communityID	This is field of MM data. Specifies the default community ID for the subscriber. Community IDs are used to control message sending and receiving among groups of subscribers.	Optional	The default value is 1.
numericAddress	This is field of MM data. Specifies a	Optional	The numeric

	unique address in the voice mail network.		address can be from 1 to 50 digits and can contain the Mailbox Number.
pbxExtension	This is field of MM data. The primary telephone extension of the subscriber.	Optional	
telephoneNumber	This is field of MM data. The telephone number of the subscriber as displayed in address book listings and client applications.	Optional	The entry can be a maximum of 50 characters in length and can contain any combination of digits (0-9), period (.), hyphen (-), plus sign (+), and left and right parentheses ([) and (]).
asciiVersionOfName	This is field of MM data. If the subscriber name is entered in multibyte character format, then this field specifies the ASCII translation of the subscriber name.	Optional	
expirePassword	This is field of MM data. Specifies whether your password expires or not.	Optional	You can choose one of the following:yes: for password to expire no: if you do not want your password to expire
mailBoxLocked	This is field of MM data. Specifies whether you want your mailbox to be locked. A subscriber mailbox can become locked after two unsuccessful login attempts.This is field of Messaging specific data.	Optional	You can choose one of the following:no: to unlock your mailbox yes: to lock your mailbox and prevent access to it
personalOperatorMailbox	This is field of MM data. Specifies the mailbox number or transfer dial string of	Optional	

	the subscriber's personal operator or assistant. This field also indicates the transfer target when a caller to this		
personalOperatorSchedule	This is field of MM data. Specifies when to route calls to the backup operator mailbox. The default value for this field is Always Active.	Optional	
tuiMessageOrderAdmin	This is field of MM data.	Optional	You can choose one of the following: urgent first then newest; oldest messages first; newest messages first; urgent first then oldest
tuiMessageOrderDeleted	This is field of MM data.	Optional	You can choose one of the following: urgent first then newest; oldest messages first; newest messages first; urgent first then oldest
tuiMessageOrderNew	This is field of MM data.	Optional	You can choose one of the following: urgent first then newest; oldest messages first; newest messages first; urgent first then oldest
tuiMessageOrderSaved	This is field of MM data.	Optional	You can choose one of the following: urgent first then newest;

			oldest messages first; newest messages first; urgent first then oldest
intercomPaging	This is field of MM data. Specifies the intercom paging settings for a subscriber.	Optional	You can choose one of the following: paging is off: to disable intercom paging for this subscriber.paging is manual: if the subscriber can modify, with Subscriber Options or the TUI, the setting that allows callers to page the subscriber.paging is automatic: if the TUI automatically allows callers to page the subscriber.
voiceMailEnabled	This is field of MM data. Specifies whether a subscriber can receive messages, e-mail messages and callanswer messages from other subscribers.	Optional	You can choose one of the following: - yes: to allow the subscriber to create, forward, and receive messages. - no: to prevent the subscriber from receiving call-answer messages and to hide the subscriber from the telephone user interface (TUI). The subscriber cannot

			use the TUI to access the mailbox, and other TUI users cannot address messages to the subscriber.
miscellaneous1	This is field of MM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 51
miscellaneous2	This is field of MM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 51
miscellaneous3	This is field of MM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 51
miscellaneous4	This is field of MM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 51
secondaryExtension	This is field of MM data. Specifies one or more alternate number to reach a subscriber. You can use secondary extensions to specify a telephone number for direct reception of faxes, to allow callers to use an existing Caller Application, or to identify each line appearance on the subscriber's telephone set if they have	Optional	

	different telephone numbers.		
cmmSpecific	This is complex type for Communication Manager Messaging specific fields data.	Optional	
switchNumber	This is field of CMM data. Specifies the number of the switch on which this subscriber's extension is administered.	Optional	You can enter "0" through "99", or leave this field blank.Leave this field blank if the host switch number should be used.Enter a "0" if no message waiting indicators should be sent for this subscriber. You should enter 0 when the subscriber does not have a phone on any switch in the network.
Community ID	This is field of CMM data. Specifies the default community ID for the subscriber. Community IDs are used to control message sending and receiving among groups of subscribers.	Optional	The default value is 1.
accountCode	This is field of CMM data. Specifies the Subscriber Account Code. The Subscriber Account Code is used to create Call Detail Records on the switch for calls placed by the voice ports. The value you enter in this field can contain any combination of digits from 0 to 9. If an account code is not specified, the system will use the subscriber's mailbox extension as the account code.	Optional	
mwiEnabled	This is field of CMM data.	Optional	Set this field to **No** if no messaging waiting indicators

			should be sent for this subscriber or if the subscriber does not have a phone on any switch in the network.
coveringExtension	This is field of CMM data. Specifies the number to be used as the default destination for the Transfer Out of Messaging feature.	Optional	You can enter 3 to 50 digits in this field depending on the length of the system's extension, or leave this field blank.
miscellaneous1	This is field of CMM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 11
miscellaneous2	This is field of CMM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 11
miscellaneous3	This is field of CMM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 11
miscellaneous4	This is field of CMM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 11
secondaryExtension	This is field of CMM data. Specifies alternate number to reach a subscriber. You can use secondary extension to specify a telephone number for direct reception of faxes, to allow callers to use an existing Caller Application, or to identify each line appearance on the subscriber's telephone set if they have different telephone numbers.	Optional	Valid Entries are blank or 3-50 digits (0-9), depending on the length of the system's extension.
asciiVersionOfName	This is field of CMM data. If the subscriber name is entered in multi-byte character format, then this field specifies the ASCII translation of the subscriber name.	Optional	

frwdDestAddress	This is field of CMM data. Specifies an email address to be used as the Forwarding Destination Address where messages should be forwarded to when an incoming message is received.	Optional	This field can include alphanumeric characters, periods (.), hyphens (-), underscores (_), pluses (+), colons (:) and should also include the commercial at (@). Note: we allow maxLength 76 for frwdDestAddress in order to avoid issues with System Manager data base varchar type limitation.
deleteAfterForward	This is field of CMM data. Specifies that an incoming message should be deleted once it is forwarded on to the Forwarding Destination Address.	Optional	
aamSpecific	This is field of Avaya Aura Messaging specific data.	Optional	
siteId	This is field of AAM data. The Id of Site	Optional	
numericAddress	This is field of AAM data. Specifies a unique address in the voice mail network.	Optional	The numeric address can be from 1 to 52 digits and can contain the Mailbox Number.
pbxExtension	This is field of AAM data. The primary telephone extension of the subscriber.	Optional	It can be from 3 to 50 digits.
storageDestination	This is field of AAM data. The option to specify the storage server.	Optional	It can be Avaya Message Store or Microsoft Exchange
storageDestinationAccountId	This is field of AAM data. The email address of Exchange Server.	Mandatory if storageDestination is Microsoft Exchange	
storageDestinationServer	This is field of AAM data. The FQDN of Exchange Server	Mandatory if storageDestination is Microsoft	

		Exchange	
telephoneNumber	This is field of AAM data. The telephone number of the subscriber as displayed in address book listings and client applications.	Optional	The entry can be a maximum of 32 characters in length and can contain any combination of digits (0-9), period (.), hyphen (-), plus sign (+), and left and right parentheses ([) and (]).
asciiVersionOfName	This is field of AAM data. If the subscriber name is entered in multi-byte character format, then this field specifies the ASCII translation of the subscriber name.	Optional	
expirePassword	This is field of AAM data. Specifies whether your password expires or not.	Optional	You can choose one of the following:yes: for password to expire no: if you do not want your password to expire
mailBoxLocked	This is field of AAM data. Specifies whether you want your mailbox to be locked. A subscriber mailbox can become locked after three unsuccessful login attempts.	Optional	You can choose one of the following: - no: to unlock your mailbox - yes: to lock your mailbox and prevent access to it
mustChangePassword	This is field of AAM data. The option to enable the user to change the password when users call the voice mailboxes the next time.	Optional	It can be true or false
personalAttendant	This is field of AAM data. Specifies subscriber's personal attendant.	Optional	
intercomPaging	This is field of AAM data. Specifies the intercom paging settings for a subscriber. **Note: intercomPaging is no longer used since SMGR_10.1.0.0.**	Optional	You can choose one of the following: - paging is off: to disable intercom paging for this

			subscriber. - paging is manual: if the subscriber can modify, with Subscriber Options or the TUI, the setting that allows callers to page the subscriber. - paging is automatic: if the TUI automatically allows callers to page the subscriber.
tuiMessageOrder	This is field of AAM data. Specifies the order in which the subscriber hears the voice messages. **Note: tuiMessageOrder field is no longer used since SMGR 7.1.2.** You should use playback order related fields for Unread, Read and Saved messages instead of tuiMessageOrder: tuiMessageOrderUnread, playUnreadUrgentFirst, arrangeBySenderUnread, tuiMessageOrderRead, playReadUrgentFirst, arrangeBySenderRead, tuiMessageOrderSaved, playSavedUrgentFirst, arrangeBySenderSaved.	Optional	You can choose one of the following: - urgent first then newest: to direct the system to play any messages marked as urgent prior to playing non-urgent messages. Both the urgent and non-urgent messages are played in the reverse order of how they were received. - oldest messages first: to direct the system to play messages in the order they were received. - urgent first then oldest: to direct the system to play any messages marked as urgent prior to playing non-urgent messages. Both

			the urgent and non-urgent messages are played in the order of how they were received. - newest messages first: to direct the system to play messages in the reverse order of how they were received.
voiceMailEnabled	This is field of AAM data. Specifies whether a subscriber can receive messages, e-mail messages and call-answer messages from other subscribers. **Note: voiceMailEnabled is no longer used since SMGR_10.1.0.0.**	Optional	You can choose one of the following: - yes: to allow the subscriber to create, forward, and receive messages. - no: to prevent the subscriber from receiving call-answer messages and to hide the subscriber from the telephone user interface (TUI). The subscriber cannot use the TUI to access the mailbox, and other TUI users cannot address messages to the subscriber.
miscellaneous1	This is field of AAM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 51
miscellaneous2	This is field of AAM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system.	Optional	Max length 51

miscellaneous3	This is field of AAM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system. **Note: miscellaneous3 is no longer used since SMGR_10.1.0.0.**	Optional	Max length 51
miscellaneous4	This is field of AAM data. Specifies additional, useful information about a subscriber. Entries in this field are for convenience and are not used by the messaging system. **Note: miscellaneous4 is no longer used since SMGR_10.1.0.0.**	Optional	Max length 51
secondaryExtension	This is field of AAM data. Specifies one or more alternate number to reach a subscriber. You can use secondary extensions to specify a telephone number for direct reception of faxes, to allow callers to use an existing Caller Application, or to identify each line appearance on the subscriber's telephone set if they have different telephone numbers.	Optional	
TimeZone	This is field of AAM data. This is time zone for Avaya Aura Messaging time subscribers. This attribute must be in 'Standardized Name' format. Example: 'America/Phoenix' and administrator must be sure that entering time zone exists on AAM Server; otherwise the System Manager Server time zone will be set for AAM subscriber.	Optional	This attribute must be in 'Standardized Name' format. Example: 'America/Phoenix'
tuiMessageOrderUnread	This is field of AAM data. The option defines the order for unread messages in Aria or AUDIX TUI case and for all messages in CallPilot TUI case.	Optional	You can choose one of the following: Play newest first : The TUI plays the unread messages starting with the newest message first. Play oldest first: The TUI plays the unread messages starting with the

			oldest message first. Default value: The TUI plays the unread messages in order defined by selected Class Of Service (i.e. Play newest first or Play oldest first). This value is only for AAM 7.0 and above.
playUnreadUrgentFirst	This is field of AAM data. Play important messages before others: the TUI plays the unread urgent messages first.	Optional	For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are: Yes, No, Default value.
arrangeBySenderUnread	This is field of AAM data. The TUI plays the unread messages that Messaging sorts by sender name.	Optional	For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are: Yes, No, Default value.
tuiMessageOrderRead	This is field of AAM data. The option defines the order for read messages in Aria or AUDIX TUI case.	Optional	You can choose one of the following: Play newest first : The TUI plays the read messages starting with the newest message first. Play oldest first: The TUI plays the read messages starting with the oldest message first. Default value: The TUI plays the read messages in order defined by selected Class Of Service (i.e. Play newest first or Play oldest first). This value is only

			for AAM 7.0 and above.
playReadUrgentFirst	This is field of AAM data. Play important messages before others: the TUI plays the read urgent messages first.	Optional	For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are: Yes, No, Default value.
arrangeBySenderRead	This is field of AAM data. The TUI plays the read messages that Messaging sorts by sender name.	Optional	For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are: Yes, No, Default value.
tuiMessageOrderSaved	This is field of AAM data. The option defines the order for saved messages in Aria TUI case.	Optional	You can choose one of the following: Play newest first : The TUI plays the saved messages starting with the newest message first. Play oldest first: The TUI plays the saved messages starting with the oldest message first. Default value: The TUI plays the saved messages in order defined by selected Class Of Service (i.e. Play newest first or Play oldest first). This value is only for AAM 7.0 and above.
playSavedUrgentFirst	This is field of AAM data. Play important messages before others: the TUI plays the saved urgent messages first.	Optional	For AAM 6.3 and below the options are: Yes, No. For AAM 7.0 and above the options are: Yes, No, Default value.
arrangeBySenderSave	This is field of AAM data. The TUI plays	Optional	For AAM 6.3 and

d	the saved messages that Messaging sorts by sender name.		below the options are: Yes, No. For AAM 7.0 and above the options are: Yes, No, Default value.

Note: System Manager 7.0.1 introduces new fields in Messaging XSD: 'TimeZone', 'frwdDestAddress' and 'deleteAfterForward'

Note: System Manager 7.1 introduces new fields in Messaging XSD: 'tuiMessageOrderUnread', 'playUnreadUrgentFirst', 'arrangeBySenderUnread', 'tuiMessageOrderRead', 'playReadUrgentFirst', 'arrangeBySenderRead', 'tuiMessageOrderSaved', 'playSavedUrgentFirst', 'arrangeBySenderSaved'

XSD attribute definition for retrieving, creating and updating and User(s): Session Manager

Attribute	Attribute Description	Mandatory/Optional	Validation Constraints
Policy policyType	Specify the name of policy for the Session Manager communication profile.		-
Primary Session Manager primarySM	Specify the name of the Session Manager instance that must be used as the home server for a Communication Profile. As a home server, the primary Session Manager instance is used as the default access point for connecting devices associated with the Communication Profile to the Aura network.	Mandatory	-
Secondary Session Manager secondarySM	If a secondary Session Manager instance is specified, this Session Manager provides continued service to SIP devices associated with this communication profile when the primary Session Manager is not available.	Optional	-

Third Session Manager thirdSM	If a third Session Manager instance is specified, this Session manager provides continued service to SIP devices associated with this communication profile when the primary or the secondary Session Manager server becomes unavailable.	Optional	-
Fourth Session Manager fourthSM	If a fourth Session Manager instance is specified, this Session manager provides continued service to SIP devices associated with this communication profile when the primary, secondary, or third Session Manager server becomes unavailable.	Optional	-
Primary Region primaryFixedRegion	Specify the name of the primary region of the primary Session Manager server.	Mandatory	-
Secondary Region secondaryFixedRegion	Specify the name of the secondary region, if secondary Session Manager server specified.	Optional	-
Third Region thirdFixedRegion	Specify the name of the third region, if third Session Manager server specified.	Optional	-
Fourth Region fourthFixedRegion	Specify the name of the fourth region, if fourth Session Manager server specified.	Optional	-
Survivability Server	For local survivability, you can specify the name of a Survivability Server (a SIP Entity) to provide survivability communication services for devices associated with a communication profile in the event that local connectivity to Session Manager instances in the Aura Core is lost. If you specify a Branch Session Manager, and the termination and origination application sequences contain a Communication Manager application, sequencing to this application continues, locally, to the Communication Manager remote survivability server resident with the Branch Session Manager. A selection is optional. If a termination or origination application sequence contains a Communication Manager	Optional	-

	application, the Communication Manager associated with the application must be the main Communication Manager for the Communication Manager remote survivability server resident with the Branch Session Manager.		
Max. Simultaneous Devices maxSimultaneousDevices	The maximum number of endpoints that you can register at a time by using this communication profile. If you register more than one endpoint, all the endpoints receive calls simultaneously.	The default value is 1.	
Block New Registration When Maximum Registrations Active blockNewRegistrationWhenMaxActive	Set the value to true or false. If you do not set the attribute, by default, the system sets the attribute to false. If you set to true and if an endpoint tries to register using this communication profile when the maximum number of allowed simultaneous registrations reaches, the endpoint cannot register with Session Manager The endpoint does not have the SIP service. If the value is set to false, the default, the endpoint can register only after the system cancels the registration of the oldest endpoint. The stopped endpoint does not have the SIP service.		-
Origination Application Sequence originationAppSequence	Specify an Application Sequence that will be invoked when calls are routed from this user. A selection is optional. If an origination and a termination application sequences are specified, and if each sequence contains a Communication Manager application, Communication Manager must be the same in both the sequences.	Optional	-

Termination Application Sequence	Specify an Application Sequence that will be invoked when calls are routed to this user. A selection is optional. If an origination and a termination application sequences are specified, and if each sequence contains a Communication Manager application, Communication Manager must be the same in both the sequences.	Optional	-
Enable Centralized Call History enabledisablecalllog	The option to enable the call history feature for SIP users.	Optional	-
Home Location	Specify a Home Location (the name of a Location – navigate to Routing > Locations) to support mobility for a user. When this user calls numbers that are not associated with an administered user, dial-plan rules (Routing > Dial Patterns) will be applied to complete the call based on this “home” location regardless of the physical location of the SIP device used to make the call. A selection is mandatory.	Mandatory	-
Conference Factory Set confFactorySet	The Conference Factory Set contains a set of Well Known Conference URIs that you can use as conference server to initiate conference calls. If the system has Conference Factory Set administered, you can get the list of Well Known Conference URIs. You require the URIs to initiate conference calls.	Optional	-
Emergency Calling Origination Sequence emergencyOriginationAppSequence	Name of the Emergency Origination Application Sequence. The list of application sequences invoked when the system routes emergency calls from this user.	Optional	-
Emergency Calling Termination Sequence emergencyTerminationAppSequenceApplication	Name of the Emergency Termination Application Sequence. The list of application sequences invoked when the system routes emergency calls to this user.	Optional	-

XSD attribute definition for retrieving, creating and updating and User(s): Endpoint

Attribute	Attribute Description	Mandatory /Optional	Validation Constraints
CM Name	Name of the Communication Manager system as it appears in 'Applications/Application Management/Entities	Mandatory	
Use Existing Extension	'true' if already created extension is to be used. 'false' if available extension is to be used.	Optional	
Extension	Station extension number that need to be assigned to the user	Mandatory	
extensionRange	Extension Range which will be used to create Station using available extension within given range.	Optional	
SecurityCode	A Station Security Code (SSC) provides security to a station user by preventing other users from accessing functions associated with the user's station. Each station user can change their own SSC if they know the station's current settings.	Optional	Value can be digit only.
Template Name template	Template name to be used to create endpoint. Values defined in Template will be used if not provided.	Optional	
Set Type setType	Specifies the set type of the endpoint.	Optional	

Port port	Valid values for port.	Optional	01 to 64 First and second numbers are the cabinet number A to E Third character is the carrier 01 to 20 Fourth and fifth characters are the slot number 01 to 32 Sixth and seventh characters are the circuit number x or X Indicates that there is no hardware associated with the port assignment since the switch was set up, and the administrator expects that the extension has a non-IP set. Or, the extension had a non-IP set, and it dissociated. Use x for Administered WithOut Hardware (AWOH) and Computer Telephony (CTI) endpoints, as well as for SBS Extensions. IP Indicates that there is no hardware associated with the port assignment since the switch was set up, and the administrator expects that the extension has an IP set. This is automatically entered for certain IP endpoint set

			types, but you can enter for a DCP set with softphone permissions. This changes to the s00000 type when the set registers.
Delete endpoint is unassigned deleteOnUnassign	Whether the endpoint must be deleted if it unassigned from the user.	Optional	
Lock messages feature. lockMessages	Enable/ Disable lock messages feature.	Optional	true/false to enable/disable lock messages feature.
Coverage Path 1 coveragePath1	A coverage path is a prioritized sequence of extensions to which your voice system will route an unanswered call.	Optional	Valid values: Path Number between 1-9999, time of day table, t1-t999, or blank.
Coverage Path 2	A coverage path is a prioritized sequence of extensions to which your voice system will route an unanswered call.	Optional	Valid values: Path Number between 1-9999, time of day table, t1-t999, or blank.
Hunt To Station huntToStation	The extension the system must hunt to for this telephone when the telephone is busy. A endpoint hunting chain can be created by assigning a hunt-to endpoint to a series of telephones.	Optional	

Tenant Number tn	Provides for partitioning of attendant groups and/or endpoints and trunk groups. Typically this is used for multiple tenants in a building or multiple departments within a company or organization.	Mandatory	Valid values: 1 to 100
Class of Restriction cor	This is used for multiple tenants in a building or multiple departments within a company or organization. This is used for multiple tenants in a building or multiple departments within a company or organization.	Mandatory	Valid values: 0 to 995
Class of Service cos	Class of Service lets you define groups of users and control those groups' access to features.	Mandatory	Valid values: 0 to 15
speakerphone	Controls the behavior of speakerphones.	Optional	Valid values : none, 1-way, 2-way
Display Language displayLanguage	The language that displays on endpoints.	Optional	Time of day is displayed in 24-hour format (00:00 - 23:59) for all languages except English, which is displayed in 12-hour format (12:00 a.m. to 11:59 p.m.). unicode: Displays English messages in a 24-hour format . If no Unicode file is installed, displays messages in English by default.

Personalized Ringing Pattern personalizedRinging Pattern	Defines the personalized ringing pattern for the endpoint.Personalized Ringing allows users of some telephones to have one of 8 ringing patterns for incoming calls. For virtual endpoints, this field dictates the ringing pattern on its mapped to physical telephone.		L = 530 Hz, M = 750 Hz, and H = 1060 Hz Valid Entries Usage: 1. MMM (standard ringing) 2. HHH 3. LLL 4. LHH 5. HHL 6. HLL 7. HLH 8. LHL
Message Lamp Extension messageLampExt	The Message Lamp Extension associated with the current extension.	Mandatory	
muteButtonEnabled	Enables or disables the mute button on the endpoint.		
Media Complex Extension mediaComplexExt	When used with Multi-media Call Handling, indicates which extension is assigned to the data module of the multimedia complex. Users can dial this extension to place either a voice or a data call, and voice conversion, coverage, and forwarding apply as if the call were made to the 1-number.	Optional	Valid Entry Usage A valid BRI data extension For MMCH, enter the extension of the data module that is part of this multimedia complex.H.323 endpoint extension For 4600 series IP Telephones, enter the corresponding H.323 endpoint. For IP Softphone, enter the corresponding H.323 endpoint. If you enter a value in this field, you

			can register this endpoint for either a road-warrior or elecommuter/Avaya IP Agent application. blank Leave this field blank for single-connect IP applications.
IP Softphone ipSoftphone	Whether this is IP soft phone.	Optional	
Servivable GK Node Name survivableGkNodeName	Survivable GK Node Name identifies the existence of other H.323 gatekeepers located within gateway products that offer survivable call features. For example, the MultiTech MVPxxx-AV H.323 gateway family and the SLS function within the H.248 gateways. When you enter a valid IP node name in this field, Communication Manager adds the IP address of this gateway to the bottom of the Alternate Gatekeeper List for this IP network region. As H.323 IP endpoints register with Communication Manager, this list is sent down in the registration confirm message. The IP endpoint can use the IP address of this Survivable Gatekeeper as the call controller of last resort to register with. Available only if the endpoint type is an H.323 endpoint (46xxor 96xx models).	Optional	Valid Entry Usage Valid IP node name Any valid previously-administered IP node name.
Survivable class of restriction survivableCOR	Sets a level of restriction for endpoints to be used with the survivable dial plan to limit certain users to only to certain types of calls. You can list the restriction levels in order from the most restrictive to least restrictive. Each level assumes the calling ability of the ones above it. This field is used by PIM module of the Integrated	Optional	Valid Entries Usage emergency - This endpoint can only be used to place emergency calls.unrestricted - This endpoint can place a call to any number defined in

	Management to communicate with the Communication Manager administration tables and obtain the class of service information. PIM module builds a managed database to send for Standard Local Survivability (SLS) on the H.248 gateways. Available for all analog and IP endpoint types.		the Survivable Gateway Call Controller's routing tables. Those strings marked as deny are also denied to these users. Internal - This endpoint can only make intra-switch calls. This is the default. local - This endpoint can only make calls that are defined as locl, op, svc, or hnpa in the Survivable Gateway Call Controller's routing tables. toll - This endpoint can place any national toll calls that are defined as fnpa or natl on the Survivable Gateway Call Controller's routing tables.
Survivable Trunk Destination survivableTrunkDest	Designates certain telephones as not being allowed to receive incoming trunk calls when the Media Gateway is in survivable mode. This field is used by the PIM module of the Integrated Management to successfully interrogate the Communication Manager administration tables and obtain the class of service information. PIM module builds a managed database to send for SLS on the H.248 gateways. Available for all analog and IP endpoint types.	Optional	Valid Entry Usage: true - Allows this endpoint to be an incoming trunk destination while the Media Gateway is running in survivability mode. This is the default. false - Prevents this endpoint from receiving incoming trunk calls when in survivable mode.
Voice Mail Number voiceMailNumber	Enter the complete Voice Mail Dial Up number.	Optional	String

offPremisesStation	Analog telephones only.	Optional	Valid entries Usage:true - Enter true if this telephone is not located in the same building with the system. If you enter true, you must complete R Balance Network. false - Enter false if the telephone is located in the same building with the system.
dataOption	If a second line on the telephone is administered on the I-2 channel, enter analog. Otherwise, enter data module if applicable or none.	Optional	Valid entries analog, none.
Message Waiting Indicator messageWaitingIndicator	If led or neon, then messageLampExt must be enable otherwise its blank.	Optional	Valid entries: led, neon, none.
remoteOfficePhone	Enter true to use this endpoint as an endpoint in a remote office configuration.	Optional	Valid entries:audix - If LWC is attempted, the messages are stored in AUDIX.spe - If LWC is attempted, the messages are stored in the system processing element (spe).none - If LWC is attempted, the messages are not stored.
lwcActivation	Enter true to allow internal telephone users to leave short LWC messages for this extension. If the system has hospitality, enter true for guest-room telephones if the extension designated to receive failed wakeup messages must receive LWC messages that indicate the wakeup calls failed. Enter true if LWC Reception is audix.	Optional	Boolean

activeStationRinging	Active endpoint Ringing	Optional	Valid entries:single continuous if-busy-single silent
idleActiveRinging	Defines how call rings to the telephone when it is on-hook.	Optional	Valid entries:continuous - Enter continuous to cause all calls to this telephone to ring continuously. if-busy-single - Enter if-busysingle to cause calls to this telephone to ring continuously when the telephone is off-hook and idle and calls to this telephone to receive one ring cycle and then ring silently when the telephone is off-hook and active. silent-if-busy - Enter silent-if-busy to cause calls to ring silently when this endpoint is busy.
switchhookFlash	Must be set to true when the Type field is set to H.323	Optional	Boolean
ignoreRotaryDigits	If this field is true, the short switch-hook flash (50 to 150) from a 2500-type set is ignored.	Optional	Boolean
h320Conversion	H.320 Conversion — Valid entries are true and false (default). This field is optional for non-multimedia complex voice endpoints and for Basic multimedia complex voice endpoints. It is mandatory for Enhanced multimedia complex voice endpoints. Because the system can only handle a limited number of conversion calls, you might need to limit the number of	Optional	Boolean

	telephones with H.320 conversion. Enhanced multimedia complexes must have this flag set to true.		
serviceLinkMode	The service link is the combined hardware and software multimedia connection between an Enhanced mode complex's H.320 DVC system and the Avaya DEFINITY Server which ends the H.320 protocol. A service link is never used by a Basic mode complex H.320 DVC system. Connecting a service link will take several seconds. When the service link is connected, it uses MMI, VC and system timeslot resources. When the service link is disconnected it does not tie up any resources. The Service Link Mode can be administered as either 'as-needed' or 'permanent' as described below: - As- Needed - Most non-call center multimedia users will be administered with this service link mode. The as-needed mode provides the Enhanced multimedia complex with a connected service link whenever a multimedia call is answered by the endpoint and for a period of 10 seconds after the last multimedia call on the endpoint has been disconnected. Having the service link stay connected for 10 seconds allows a user to disconnect a multimedia call and then make another multimedia call without having to wait for the service link to disconnect and re-establish. - Permanent – Multimedia call center agents and other users who are constantly making or receiving multimedia calls might want to be administered with this service link mode.The permanent mode service link will be connected during the endpoint's first multimedia call and will remain in a connected state until the user disconnects from their PC's multimedia application or the Avaya	Optional	Valid entries as-needed permenant

	DEFINITY Server restarts. This provides a multimedia user with a much quicker video cut-through when answering a multimedia call from another permanent mode endpoint or a multimedia call that has been early answered.		
multimediaMode	There are two multimedia modes, Basic and Enhanced,	Optional	Basic - A Basic multimedia complex consists of a BRIconnected multimedia-equipped PC and a non-BRI-connected multifunction telephone set. Enhanced - An Enhanced multimedia complex consists of a BRI-connected multimediaequipped PC and a non-

			BRIconnected multifunction telephone.
mwiServedUserType	Controls the auditing or interrogation of a served user's message waiting indicator (MWI).	Optional	Valid entries:1. fp-mwi - Use if the endpoint is a served user of an fp-mwi message center. 2. qsig-mwi - Use if the endpoint is a served user of a qsig-mwi message center. 3. sip adjuncts - Use if the endpoint is a served user of a sip adjunct message center. 4. blank - Leave blank if you do not want to audit the served user's MWI or if the user is not a served user of either an fp-mwi or qsigmwi message center.
audixName	The AUDIX associated with the endpoint. Must contain a user-defined adjunct name that was previously administered.	Optional	String

automaticMoves	Automatic Moves allows a DCP telephone to be unplugged from one location and moved to a new location without additional Communication Manager administration. Communication Manager automatically associates the extension to the new port.	Optional	Valid entries:1. always - Enter always and the DCP telephone can be moved anytime without additional administration by unplugging from one location and plugging into a new location. 2. once - Enter once and the DCP telephone can be unplugged and plugged into a new location once. After a move, the field is set to done the next time that routine maintenance runs on the DCP telephone. Use once when moving a large number of DCP telephones so each extension is removed from the move list. Use once to prevent automatic maintenance replacement. 3. no - Enter no to require administration in order to move the DCP telephone. 4. done - Done is a display-only value. Communication Manager sets the field to done after the telephone is moved and routine maintenance runs on the DCP telephone. 5. Error - Error is a

			display-only value. Communication Manager sets the field to error, after routine maintenance runs on the DCP telephone, when a non-serialized telephone is set as a movable telephone.

remoteSoftphoneEmergencyCalls	An Avaya IP endpoint can dial emergency calls (for example, 911 calls in the U.S.). It only reaches the local emergency service in the Public Safety Answering Point area where the telephone system has local trunks.	Optional	Valid entries:1. As-on-local - as-on-local sends the extension entered in the Emergency Location Extension field on the Endpoint screen to the Public Safety Answering Point (PSAP) 2. Block - Enter block to prevent the completion of emergency calls. 3. Cesid - Enter cesid to allow Communication Manager to send the CESID information supplied by the IP Softphone to the PSAP. 4. Option - Enter option to allow the user to select the option (extension, block, or cesid) that the user selected during registration and the IP Softphone reported.

emergencyLocation Ext	This field allows the system to properly identify the location of a caller who dials a 911 emergency call from this endpoint. An entry in this field must be of an extension type included in the dial plan, but does not have to be an extension on the local system. It can be a UDP extension. The entry defaults to blank. A blank entry typically is used for an IP softphone dialing in through PPP from somewhere outside your network. If you populate the IP Address Mapping screen with emergency numbers, the feature functions as follows: If the Emergency Location Extension field in the Endpoint screen is the same as the Emergency Location Extension field in the IP Address Mapping screen, the feature sends the extension to the Public Safety Answering Point (PSAP). If the Emergency Location Extension field in the Endpoint screen is different from the Emergency Location Extension field in the IP Address Mapping screen, the feature sends the extension in the IP Address Mapping screen to the Public Safety Answering Point (PSAP).	Optional	
alwaysUse	A softphone can register no matter what emergency call handling settings the user has entered into the softphone. If a softphone dials 911, the administered Emergency Location Extension is used. The softphone's user-entered settings are ignored. If an IP telephone dials 911, the administered Emergency Location Extension is used. If a call center agent dials 911, the physical endpoint extension is displayed, overriding the administered LoginID for ISDN Display . Does not apply to SCCAN wireless telephones, or to extensions administered as type h.323.	Optional	Boolean

precedenceCallWaiting	Activates or deactivates Precedence Call Waiting for this endpoint.	Optional	
autoSelectAnyIdleAppearance	Enables or disables automatic selection of any idle appearance for transferred or conferenced calls. Communication Manager first attempts to find an idle appearance that has the same extension number as the call being transferred or conferenced has. If that attempt fails, Optional Boolean Communication Manager selects the first idle appearance. coverageMsgRetrieval	Optional	Boolean
coverageMsgRetrieval	Allows or denies users in the telephone's Coverage Path to retrieve Leave Word Calling (LWC) messages for this telephone. Applies only if the telephone is enabled for LWC Reception.	Optional	Boolean
autoAnswer	In EAS environments, the auto answer setting for the Agent LoginID can override a endpoint's setting when an agent logs in.	Optional	Valid entries:1. all: All ACD and non-ACD calls ended to an idle endpoint cut through immediately. Does not allow automatic hands-free answer for intercom calls. With non-ACD calls, the set is also rung while the call is cut through. The ring can be prevented by activating the ringer-off feature button when the Allow Ringer-off with Auto-Answer is enabled for the system.2. acd: Only ACD split /skill calls and direct agent calls to auto answer. Non-ACD calls ended to a

			endpoint ring audibly. For analog endpoints, the endpoint is off-hook and idle, only the ACD split/skill calls and direct agent calls auto answer; non-ACD calls receive busy treatment. If the endpoint is active on an ACD call and a non-ACD call arrives, the Agent receives call-waiting tone.3. none: All calls ended to this endpoint receive an audible ringing treatment. 4. icom: Allows a telephone user to answer an intercom call from the same intercom group without pressing the intercom button.
dataRestriction	Enables or disables data restriction that is used to prevent tones, such as call-waiting tones, from interrupting data calls. Data restriction provides permanent protection and cannot be changed by the telephone user. Cannot be assigned if Auto Answer is administered as all or acd. If enabled, whisper page to this endpoint is denied.	Optional	
idleAppearancePreference	Indicates which call appearance is selected when the user lifts the handset and there is an incoming call.	Optional	true - The user connects to an idle call appearance instead of the ringing call. false - The Alerting Appearance Preference is set and the user connects to the

			ringing call appearance.
callWaitingIndication	enable/disable call waiting for this endpoint	Optional	
attCallWaitingIndication	Attendant call waiting allows attendantoriginated or attendant-extended calls to a busy single-line telephone to wait and sends distinctive call-waiting tone to the single-line user. Enable/disable attendant call waiting	Optional	Boolean
distinctiveAudibleAlert	Enter true so the telephone can receive the 3 different types of ringing patterns which identify the type of incoming calls. Distinctive ringing might not work properly for off-premises telephones.	Optional	
restrictLastAppearance		Optional	Valid entries:1. true: Restricts the last idle call appearance used for incoming priority calls and outgoing call originations only. 2. false: Last idle call appearance is used for incoming priority calls and outgoing call originations.
adjunctSupervision	Enable / Disable adjunct Supervision.	Optional	Valid entries:1. true: Analog disconnect signal is sent automatically to the port after a call ends. Analog devices (such as answering machines and speakerphones) use this signal to turn the devices off after a call ends. 2. false: Hunt

			group agents are alerted to incoming calls. In a hunt group environment, the disconnect signal blocks the reception of zip tone and incoming call notification by an auto-answer endpoint when a call is queued for the endpoint.
perStationCpnSendCallingNumber	Send Calling Number.	Optional	Valid entries:1. y: All outgoing calls from the endpoint will deliver the Calling Party Number (CPN) information as "Presentation Allowed." 2. n: No CPN information is sent for the call 3. r: Outgoing non-DCS network calls from the endpoint will deliver the Calling Party Number information as "Presentation Restricted."
busyAutoCallbackWithoutFlash	Appears on the Endpoint screen for analog telephones, only if the Without Flash field in the ANALOG BUSY AUTO CALLBACK section of the Feature-Related System Parameters screen is set to true. The Busy Auto Callback without Flash field then defaults to true for all analog telephones that allow Analog Automatic Callback. Set true to provide automatic callback for a calling analog endpoint without flashing the hook.	Optional	

audibleMessageWaiting	Provides audible message waiting	Optional	Boolean
displayClientRedirection	Only administrable if Hospitality is enabled on the System Parameters Customer- Options (Optional Features) screen. This field affects the telephone display on calls that originated from a endpoint with Client Room Class of Service. Note: For endpoints with an audix endpoint type, AUDIX Voice Power ports, or ports for any other type of messaging that needs display information, Display Client Redirection must be enabled. Set true to redirect information for a call originating from a Client Room and ending to this endpoint displays.	Optional	Boolean
selectLastUsedAppearance		Optional	Valid entries: 1. True: Indicates that a endpoint's line selection is not to be moved from the currently selected line button to a different, non-alerting line button. If you enter true, the line selection on an on-hook endpoint only moves from the last used line button to a line button with an audibly alerting call. If there are no alerting calls, the line selection remains on the button last used

			for a call. 2. false: The line selection on an on-hook endpoint with no alerting calls can be moved to a different line button, which might be serving a different extension.
coverageAfterForwarding	Whether an unanswered forwarded call is provided coverage treatment.	Optional	
directIpIpAudioConnections	Allow/disallow direct audio connections between IP endpoints.	Optional	
ipAudioHairpinning	Allows IP endpoints to be connected through the server's IP circuit pack.	Optional	
primeAppearancePreference	Set prime appearance preference.	Optional	
endpointSiteData	This is complex type for Site Data fields		
room	This is field of Site Data	Optional	Max length 10
jack	This is field of Site Data	Optional	Max length 5
cable	This is field of Site Data	Optional	Max length 5
floor	This is field of Site Data	Optional	
building	This is field of Site Data	Optional	
headset	This is field of Site Data	Optional	
speaker	This is field of Site Data	Optional	
mounting	This is field of Site Data	Optional	Valid values d, w.
cordLength	This is field of Site Data	Optional	Valid range from 0 to 99.
setColor	This is field of Site Data	Optional	
abbrList	This is complex type for Station Abbreviated Dialing Data fields.	Optional	
listType	This is field of Station Abbreviated Dialing Data.	Mandatory	Valid values enhanced, group, personal, system.
number	This is field of Station Abbreviated Dialing Data.	Mandatory	A number.

buttons	This is complex type for button data	Optional	
Number	This is field of button data.	Mandatory	
Type	This is field of button data.	Optional	
data1	This is field of button data.	Optional	
data2	This is field of button data.	Optional	
data3	This is field of button data.	Optional	
data4	This is field of button data.	Optional	
data5	This is field of button data.	Optional	
data6	This is field of button data.	Optional	
endpointDataModule	This is complex type for Station Data module.	Optional	
dataExtension	This is field of Station Data module.	Mandatory	
name	This is field of Station Data module.	Optional	Max length 29
Class of restriction cor	This is field of Station Data module.	Mandatory	Valid range from 0 to 995.
Class of Service cos	This is field of Station Data module.	Mandatory	Valid range from 0 to 15.
itc	This is field of Station Data module.	Mandatory	Valid values:1. restricted 2. unrestricted
Tenant Number	This is field of Station Data module.	Mandatory	Valid range from 1 to 100.
listType	This is field of Station Data module.	Optional	Valid values:1. enhanced 2. group 3. personal 4. system
listId	This is field of Station Data module.	Optional	
specialDialingOption	This is field of Station Data module.	Optional	Valid values:1. default 2. hot-line
specialDialingAbbrDialCode	This is field of Station Data module.	Optional	
hotLineDestAbbrevList	This is field of Station Hot Line Data.	Optional	Valid range 1 to 3

hotLineAbbrevDialCode	This is field of Station Hot Line Data.	Optional	Numeric string
nativeName	This is complex type of Native Name Data.	Optional	
locale	This is field of Native Name Data. Note: If the displayName, givenName, or surname contains characters of multiple scripts then the locale tag should be present. The locale for the multiscript languages are: Japanese: ja Simplified Chinese: zh-cn Traditional Chinese: zh-tw Korean: ko-kr Vietnamese: vi-vn The locale tag is case sensitive.	Optional	
Name	This is field of Native Name Data.	Mandatory	Max length 27
Dual Registration	This field will provide the ability to automatically generate an OPS entry for an H.323 set type.	Optional	Default Value: False
Calculate Route Pattern	This field will allow SMGR to automatically select the Route Pattern for a SIP endpoint based on the Primary and Secondary Session Manager.	Optional	Default Value: False
Enable Reachability for Station Domain Control	This is the new field supported in station object.	Optional	System/Yes/No Default Value: System
displayCallerId	This is the new field supported to display Caller Id	Optional	True/False Default Value: True
callerIdMsgWaitingIndication	This is the new field supported to caller id message waiting indication	Optional	True/False Default Value: False
recallRotaryDigit	This is the new field supported for recall rotary digit	Optional	True/False Default Value: False

Note: System Manager 7.0 introduces Dual Registration, Calculate Route Pattern, and Enable Reachability for Station Domain Control.

Note: System Manager 7.0.1 has updated Endpoint xsd, added following fields for supporting CallrId and K2500 Set types: displayCallerId; callerIdMsgWaitingIndication and recallRotaryDigit

XSD attribute definition for retrieving, creating and updating and User(s): Work Assignment

Attribute	Attribute Description	Mandatory /Optional	Validation Constraints
associatedHandleName	Name of the Handle	Mandatory	
accountName		Mandatory	
accountAddress		Mandatory	
sourceName		Mandatory	
sourceAddress		Optional	

XSD attribute definition for retrieving, creating and updating and User(s): Officelinx

Attribute	Attribute Description	Mandatory/ Optional	Validation Constraints
officelinxName	Name of the Officelinx system	Mandatory	String
mailBoxNumber	The mailbox number of the subscriber.	Mandatory	Long value
numericPassword	The numeric password that is used to log in to the Officelinx system.	Optional	String
applicationUserPassword	The password that is used to gain access to non-telephone applications, such as Web Client, iLink Pro, iLink Pro Mobile, and iLink Pro Desktop	Optional	String

Company	The name of the company to which the user belongs	Optional	String
Department	The department to which the user belongs	Optional	String
featureGroup	The feature group name that determines the rules for the mailboxes associated with it	Optional	String
Capability	The type of functionality that the user contains.	Optional	String
domainAccountName	The mailbox NT account name for the Officelinx profile	Optional	String
synchronizationUserName	The account name that is used to gain access to the email server, for example, Microsoft Exchange and Google Gmail.	Optional	String

XSD attribute definition for retrieving, creating and updating and User(s): Equinox / Scopia

Attribute	Attribute Description	Mandatory/Optional	Validation Constraints
scopiaUserId	User ID that is returned from Equinox Management when a User is created on SMGR and synchronized from SMGR to Equinox Management. Internal use.	Optional	string
password	The password that is used to log in to the Avaya Equinox Management.	Mandatory	string
vrNumber	The number of a virtual room that is used to create a conference. Be default Virtual Room Number serves as Meeting ID when a conference is created.	Optional	string
needVR	For future use.	Optional	Boolean
virtualRoomId	Virtual Room ID that is returned from Equinox	Optional	string

	Management when a virtual room is created on Equinox Management because of synchronization of users. Internal use.		

Glossary

UPM	User Profile Manager
CUD	Create Update Delete
CUP	Common User Provisioning
RBAC	Role Based Access Control
CRUD	Create Read Update Delete
UPM	User Profile Management
WADL	Web Application Description Language
UM WS	User Management Web Services
JAR	Java Archive
XML	Extensible Markup Language
HTTP	Hypertext Transfer Protocol
SM	Session Manager